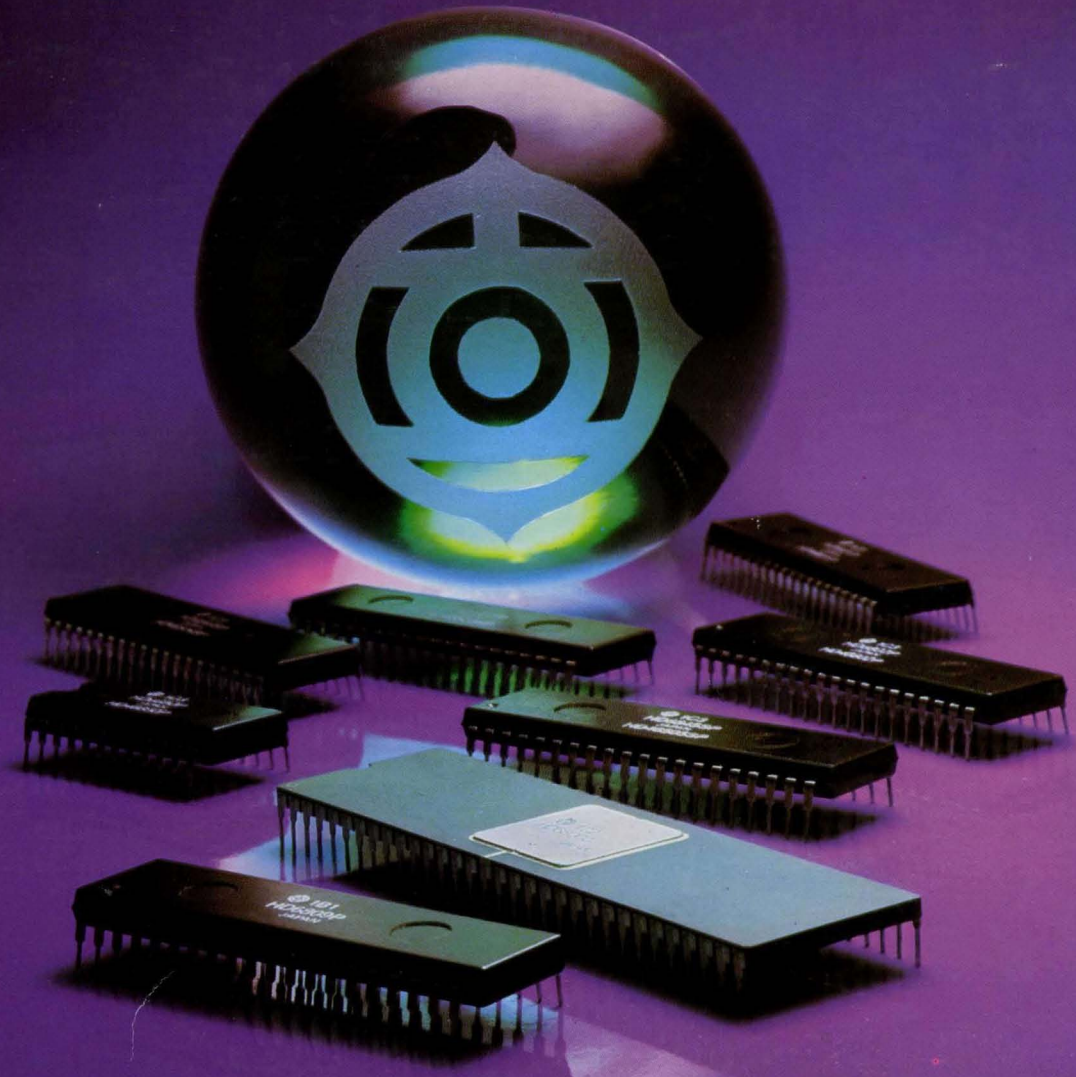
HLN062

Microcomputer Data Book



HITACHI

A World Leader in Technology

MICROCOMPUTER DATA BOOK

INDEX

GENERAL INFORMATION

- Type and Package Information
- Reliability and Ouality Assurance
- Design Procedure and Support Tools for 8-bit Single-chip Microcomputer
- HMCS6800 Family Instruction Set
- 8-bit Microcomputers for Industrial Application

TYPE AND PACKAGE INFORMATION

The Hitachi Microcomputer LSI are classified into 3 types; plastic mold type, side-brazed ceramic type and flat package, according to the type of material and outline used for the package. Therefore, after taking into consideration the operating environment and other conditions, please select the optimum package. In regard to the types which have two package materials, please define clearly when ordering the package material code (C or P) (See the following table). As List of 8/16-bit Microcomputer LSI Package shows, old type numbers are changed from HD468XX to HD68XX (standard number). But as for original products of Hitachi, the type numbers are not changed.

Division of Package Material

Types which have side-brazed ceramic and plastic mold package.

Side-brazed Ceramic		Plastic Mold
Single-chip LSI	HD68XXC	HD68XXP
Multi-chip LSI	HD68XX ∈ No indication	

■ **List of 8/16-bit Microcomputer LSI Package**

Chip Division	Type No.	Old type No.	Function	Package*			
				Pin No.	C	P	F
8-bit Single-chip	HD6801S0	–	Microcomputer Unit	40	●	●	
	HD6801S5	–		40		●	
	HD6801V0	–		40		●	
	HD6801V5	–		40		●	
	HD6803	–	Micro Processing Unit	40	●	●	
	HD6803-1	–		40		●	
	HD6805S1	–	Microcomputer Unit	28		●	
	HD6805U1	–		40		●	
	HD6805V1	–		40		●	
	HD6805W0	–		40		●	
	HD6805X0	–		64		●	
	HD6301V0	–		40		●	
	HD63A01V0	–		40		●	
	HD63B01V0	–		40		●	
	HD6303	–	Micro Processing Unit	40		●	
	HD63A03	–		40		●	
	HD63B03	–		40		●	
	HD6333	–		40		●	
	HD63A33	–		40		●	
	HD63B33	–		40		●	
	HD63L05	–	Microcomputer Unit	60			●
	HD68P01S0	–		40**	●		
	HD68P01V05	–		40**	●		
	HD68P01V07	–		40**	●		
	HD68P05V05	–		40**	●		
	HD68P05V07	–		40**	●		

(to be continued)

* The package codes of C, P, and F are applied to the package materials as follows.
C: Side-brazed Ceramic DIP, P: Plastic DIP, F: Flat Package.

** EPROM on the package.

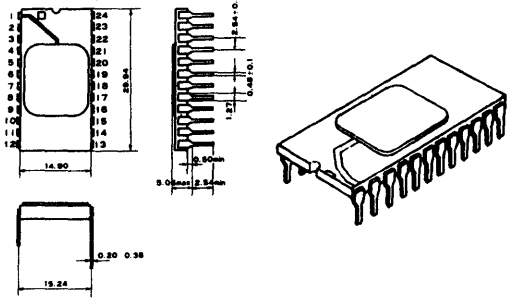
TYPE AND PACKAGE INFORMATION

Chip Division	Type No.	Old type No.	Function	Package*			
				Pin No.	C	P	F
8-bit Multi-chip	HD6800	HD46800D	Micro Processing Unit	40	●	●	
	HD68A00	HD468A00		40	●	●	
	HD68B00	HD468B00		40	●	●	
	HD6802	HD46802	Microprocessor with Clock and RAM	40	●	●	
	HD6802W	–		40		●	
	HD6809	–	Micro Processing Unit	40	●	●	
	HD68A09	–		40	●	●	
	HD68B09	–		40	●	●	
	HD6809E	–		40	●	●	
	HD68A09E	–		40	●	●	
	HD68B09E	–		40	●	●	
	HD6821	HD46821	Peripheral Interface Adapter	40	●		
	HD68A21	HD468A21		40	●	●	
	HD68B21	HD468B21		40	●	●	
	HD6840	–	Programmable Timer Module	28	●	●	
	HD68A40	–		28	●	●	
	HD68B40	–		28	●	●	
	HD6843	HD46503S	Floppy Disk Controller	40	●	●	
	HD68A43	HD46503S-1		40	●	●	
	HD6844	HD46504	Direct Memory Access Controller	40	●	●	
	HD68A44	HD46504-1		40	●	●	
	HD68B44	HD46504-2		40	●	●	
	HD6845S	HD46505S	CRT Controller	40	●	●	
	HD68A45S	HD46505S-1		40	●	●	
	HD68B45S	HD46505S-2		40	●	●	
	HD6846	HD46846	Combination ROM I/O Timer	40	●	●	
	HD6850	HD46850	Asynchronous Communications Interface Adapter	24	●	●	
	HD68A50	HD468A50		24	●	●	
	HD6852	HD46852	Synchronous Serial Data Adapter	24	●	●	
	HD68A52	HD468A52		24	●	●	
	HD46508	–	Analog Data Acquisition Unit	40		●	
	HD46508-1	–		40		●	
	HD46508A	–		40		●	
	HD46508A-1	–		40		●	
	HD146818	–	Real Time Clock Plus RAM	24		●	
16-bit Multi-chip	HD68000-4	–	Micro Processing Unit	64	●		
	HD68000-6	–		64	●		
	HD68000-8	–		64	●		
	HD68000-10	–		64	●		
	HD68450	–	Direct Memory Access Controller	64	●		

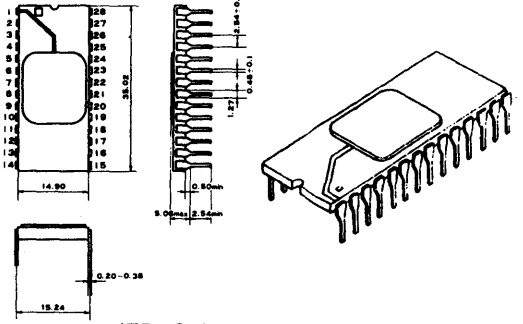
* The package codes of C, P, and F are applied to the package materials as follows.
C: Side-brazed Ceramic DIP, P: Plastic DIP, F: Flat Package.

- **Package Information (Dimensions in mm)**
- **Side-brazed Ceramic DIP**

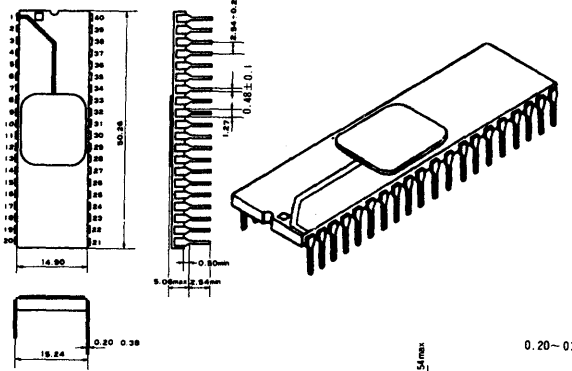
● DC-24



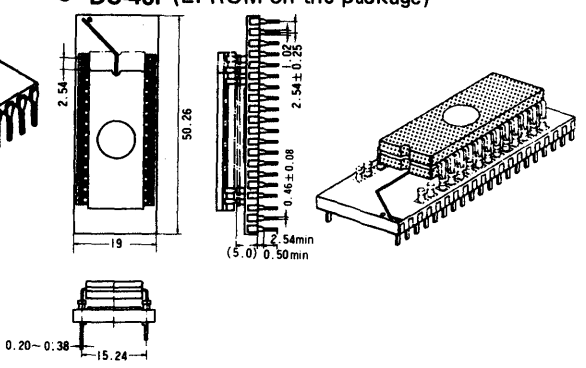
● DC-28



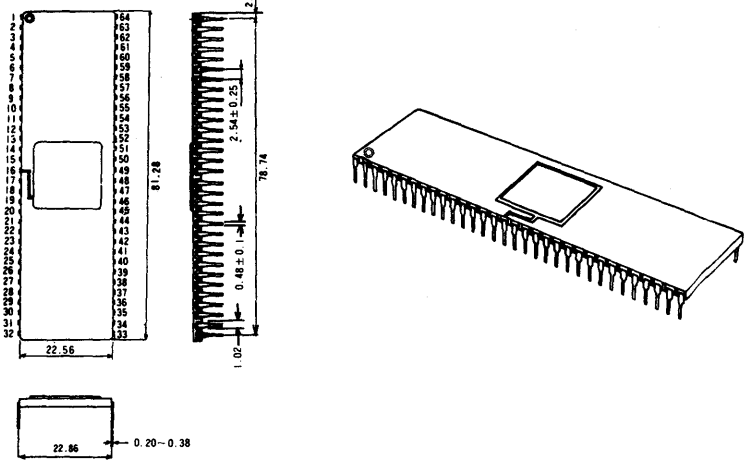
● DC-40



● DC-40P (EPROM on the package)



● DC-64



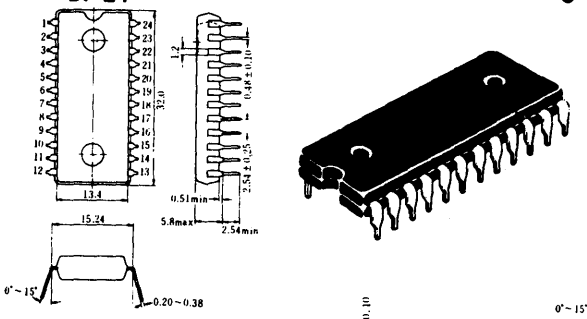
Applicable LSIs

DC-24	HD6850, HD68A50, HD6852, HD68A52
DC-28	HD6840, HD68A40, HD68B40
DC-40	HD6801S0C, HD6803C, HD6800, HD68A00, HD68B00, HD6802, HD6809, HD68A09, HD68B09, HD6809E, HD68A09E, HD68B09E, HD6821, HD68A21, HD68B21, HD6843, HD68A43, HD6844, HD68A44, HD68B44, HD6845S, HD68A45S, HD68B45, HD6846
DC-40P	HD68P01S0, HD68P01V05, HD68P01V07, HD68P05V05, HD68P05V07
DC-64	HD68000-4, HD68000-6, HD68000-8, HD68000-10, HD68450

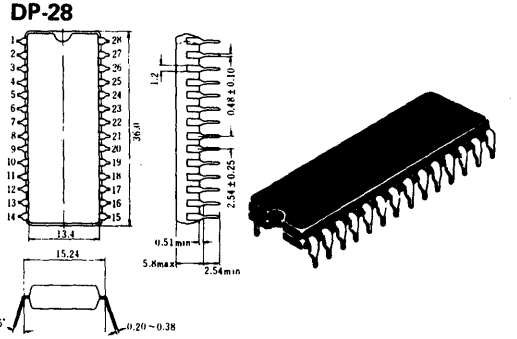
TYPE AND PACKAGE INFORMATION

- **Plastic DIP**

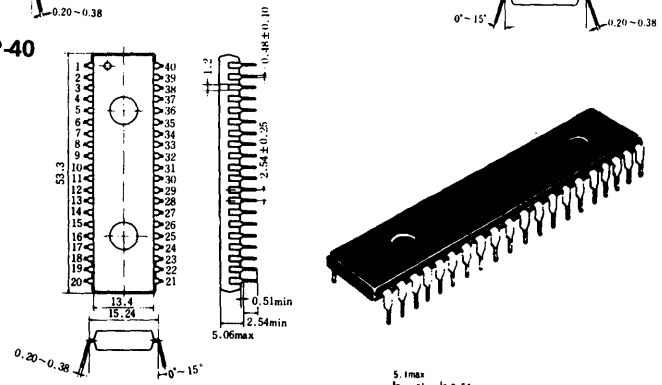
- **DP-24**



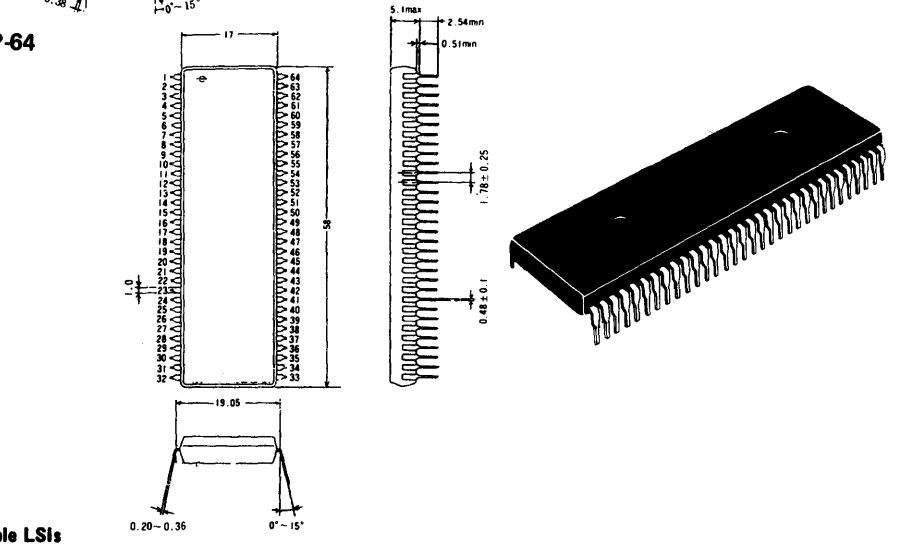
- **DP-28**



- **DP-40**



- **DP-64**

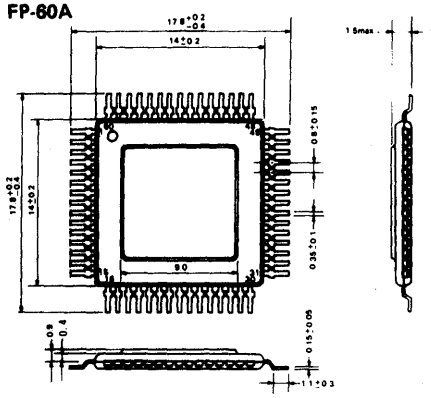


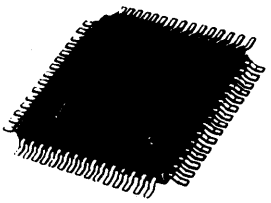
Applicable LSIs

DP-24	HD6850P, HD68A50P, HD6852P, HD68A52P, HD146818P
DP-28	HD6805S1P, HD6840P, HD68A40P, HD68B40P
DP-40	HD6801S0P, HD6801S5P, HD6801V0P, HD6801V5P, HD6803P, HD6803P-1, HD6805U1P, HD6805V1P, HD6805W0P, HD6301V0P, HD63A01V0P, HD63B01V0P, HD6303P, HD63A03P, HD63B03P, HD6333P, HD63A33P, HD63B33P, HD6800P, HD68A00P, HD68B00P, HD6802P, HD6802WP, HD6809P, HD68A09P, HD68B09P, HD6809EP, HD68A09EP, HD68B09EP, HD6821P, HD68A21P, HD68B21P, HD6843P, HD68A43P, HD6844P, HD68A44P, HD68B44P, HD6845SP, HD68A45SP, HD68B45SP, HD6846P, HD46508P, HD46508P-1, HD46508PA, HD46508PA-1
DP-64	HD6805X0P

- **Flat Package**

- **FP-60A**



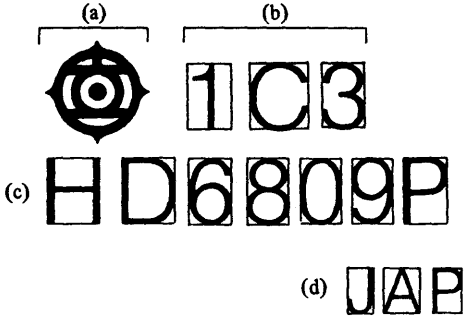


Applicable LSI	
FP-60A	HD63L05

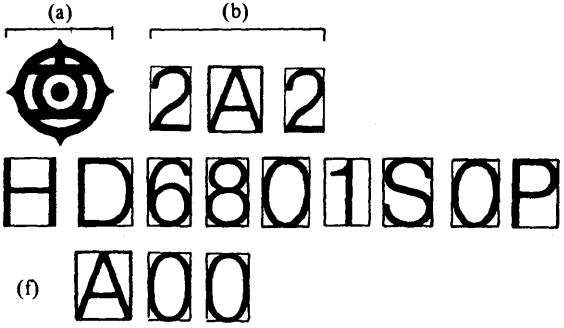
■ Marking

There are two kinds of marking. One has a new ordering No. (Case I) and the other has both new and old ordering No. (Case II). They are listed on the List of 8/16-bit Microcomputer LSI Package. Case I is applied to the LSI which has only new ordering No. listed in the List of 8/16-bit Microcomputer LSI Package and Case II is applied to the LSI which has both Ordering No.

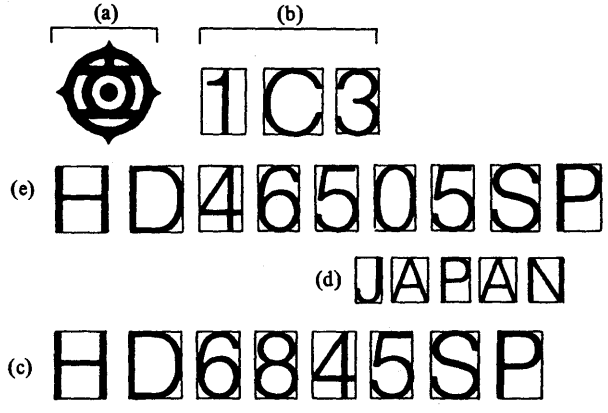
Case I (Indicated an ordering No.)



Case III (Example of marking on Single-chip)



Case II (Indicated a New & Old ordering No.)



Meaning of each mark	
(a)	Hitachi mark
(b)	Lot Code
(c)	New type No.
(d)	Japan mark
(e)	Old type No.
(f)	ROM Code

RELIABILITY AND QUALITY ASSURANCE

1. INTRODUCTION

Microcomputer is required to provide higher reliability and quality with increasing function, enlarging scale and widening application. To meet this demand, Hitachi is improving the quality by evaluating reliability, building up quality in process, strengthening inspection and analyzing field data etc..

This chapter describes reliability and quality assurance data for Hitachi 8-bit and 16-bit microcomputer based test and failure analysis results. More detail data and new information will be reported in another reliability data sheet.

2. PACKAGE AND CHIP STRUCTURE

2.1 Package

Packages are generally classified into 2 types; one is the hermetic sealed type using metal or glass and the other is the plastic molded type. Hitachi 8-bit microcomputer are produced in plastic package or side-braze package.

Selection of package type should be done considering the application, environment, reliability, cost and other factors of the system.

The reliability of plastic molded type has been greatly improved, recently their applications have been expanded to automobiles measuring and control systems, and computer terminal equipment operated under relatively severe conditions.

Actually, field application data has revealed that their failure rates in a commercial environment are equivalent to those of the hermetic sealed type.

However, in a view of reliability guarantee, the hermetic sealed type passes a leak test 100%. Due to poor screening technology, the plastic type may exhibit moisture absorption or permeation inherent to their plastic materials.

Therefore, Hitachi recommends users employ the hermetic sealed type for certain systems which require high durability against severe conditions, long service life and high reliability.

On the other hand, production output and application of plastic molded type continue to increase.

To meet such requirements, Hitachi has considerably improved moisture resistance, operation stability, and chip and plastic manufacturing process.

Plastic and side-braze package type structure are shown in Figure 1 and Table 1.

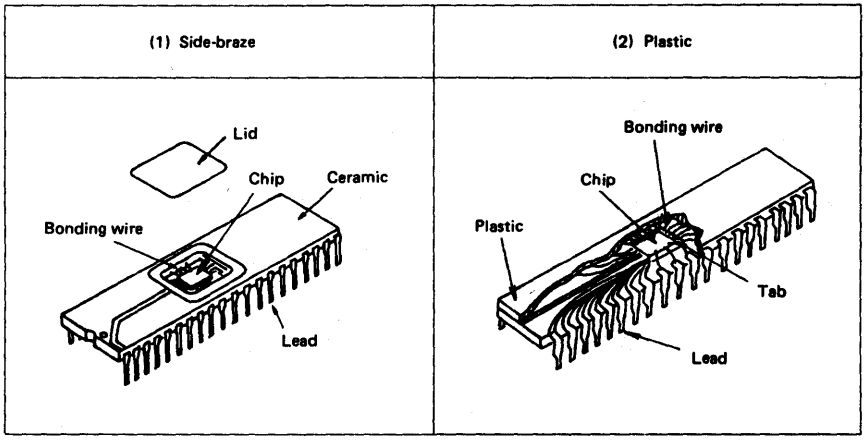


Figure 1 Package Structure

Table 1 Package Material and Properties

Item	Side-braze	Plastic
Package	Alumina	epoxy
Lead	Tin plating Brazed Alloy 42	Solder dipping Alloy 42
Seal	Au-Sn Alloy	N.A
Die bond	Au-Si	Au-Si
Wire bond	Ultrasonic	Thermo compression
Wire	Al	Au

2.2 Chip Structure

HMCS6800 family are produced in NMOS E/D technology or low power CMOS technology. Si-gate process is used in both types because of high reliability and high density.

Chip structure and basic circuit are shown in Figure 2.

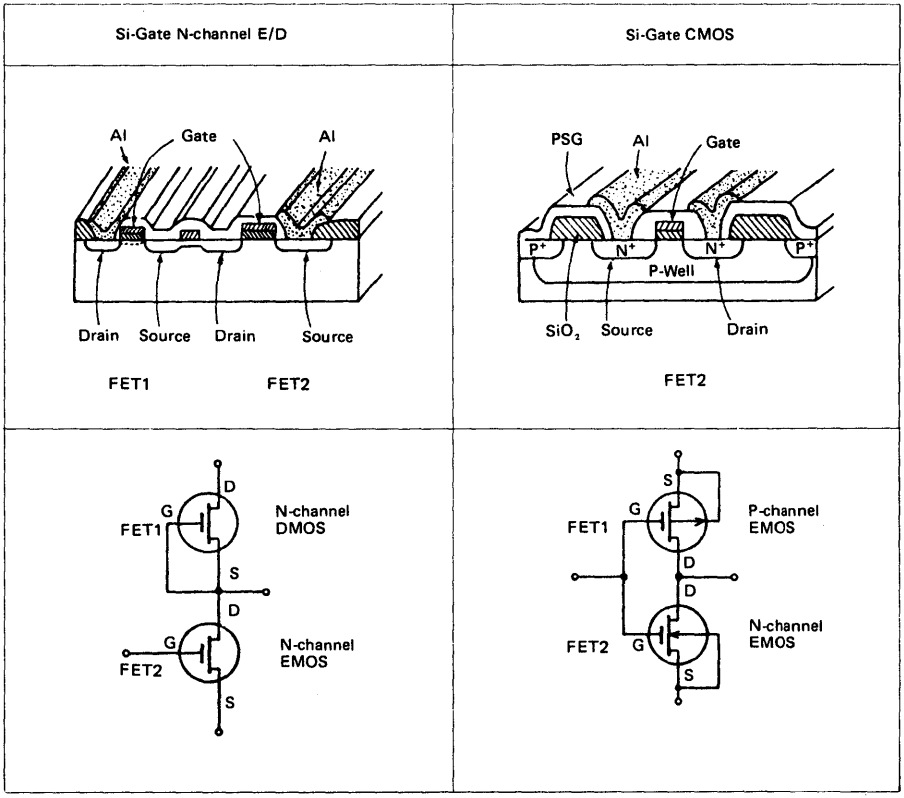


Figure 2 Chip Structure and Basic Circuit

3. QUALITY QUALIFICATION AND EVALUATION

3.1 Reliability Test Methods

Reliability test methods shown in Table 2 are used to qualify and evaluate the new products and new process.

Table 2 Reliability Test Methods

Test Items	Test Condition	MIL-STD-883B Method No.
Operating Life Test	125°C, 1000hr	1005,2
High Temp, Storage	Tstg max, 1000hr	1008,1
Low Temp, Storage	Tstg min, 1000hr	
Steady State Humidity	65°C 95%RH, 1000hr	
Steady State Humidity Biased	85°C 85%RH, 1000hr	
Temperature Cycling	-55°C ~ 150°C, 10 cycles	1010,4
Temperature Cycling	-20°C ~ 125°C, 200 cycles	
Thermal Shock	0°C ~ 100°C, 100 cycles	1011,3
Soldering Heat	260°C, 10 sec	
Mechanical Shock	1500G 0.5 sec, 3 times/X, Y, Z	2002,2
Vibration Fatigue	60Hz 20G, 32hrs/X, Y, Z	2005,1
Valiable Frequency	20~2000Hz 20G, 4 min/X, Y, Z	2007,1
Constant Acceleration	20000G, 1 min/X, Y, Z	2001,2
Lead Integrity	225gr, 90° 3 times	2004,3

3.2 Reliability Test Results

3.2.1 Dynamic Life Test

The reliability for chip design is evaluated by dynamic life test. The test results of HMCS6800 microcomputer family are shown in Table 3. Depend on these test results, the 70°C failure rate is determined 0.007%/1000hrs (confidence level 60%, activation energy 0.7eV)

Table 3 Dynamic Life Test Result

Device	Sample Size	Component Hour	Failure
HD6800	248	248000	0
HD6802	452	153712	1
HD6809	85	85000	0
HD6801	146	146000	0
HD6803	45	45000	0
HD6805	114	114000	0
MPU Total	1090	791712	1
HD6821	399	266368	1
HD6850	158	158000	0
HD6852	170	125816	0
HD6846	69	69000	0
HD6843	66	55000	0
HD6844	80	69000	0
HD6845S	88	55000	0
HD6840	64	64000	0
HD46508	140	140000	0
Peripheral Total	1234	1002184	1
Total	2324	1793896	2

3.2.2 Temperature-Humidity Bias Test

The moisture resistance of plastic package is evaluated by temperature-humidity bias test, 85°C/85% RH biased condition. The result of this test is shown in Table 4.

Table 4 85°C/85%RH Bias Test Result

Device	Sample Size	Component Hore	Failure
MPU	132	132000	0
Peripheral	226	204000	0
Total	358	336000	0

3.2.3 Storage Life Test

These tests evaluate the effect of storage at high temperature, low temperature or high humidity without bias.

(1) Plastic Package

Table 5 Storage Life Test on Plastic Package

Test Items	Condition	Sample Size	Failure
High Temp, High Humidity	65°C/95%RH, 1000hrs	288	0
High Temp, High Humidity	80°C/90%RH, 1000hrs	88	0
Presser Cooker	2atm 121°C, 100hrs	266	0
High Temp, Storage	Ta = 150°C, 1000hrs	85	0
Low Temp, Storage	Ta = -55°C, 1000hrs	34	0

(2) Side-braze

Table 6 Storage Life Test on Side-braze Package

Test Items	Condition	Sample Size	Failure
High Temp, High Humidity	65°C/95%RH, 1000hrs	90	0
High Temp, Storage	Ta = 150°C, 1000hrs	313	0
Low Temp, Storage	Ta = -55°C, 1000hrs	86	0

3.2.4 Mechanical & Environmental Test

Table 7 Mechanical & Environmental Test Results

Test Item	Condition	Plastic		Side-braze	
		Sample Size	Failure	Sample Size	Failure
Temperature Cycling	-55°C ~ 150°C 10 cycles	4159	0	4920	1
	-55°C ~ 150°C 200 cycles	826	1	359	0
Thermal Shock	0°C ~ 100°C 10 cycles	110	0	175	0
Soldering Heat	260°C, 10sec	180	0	177	0
Mechanical Shock	1500G 0.5 ms 3 times/X, Y, Z	110	0	189	0
Vibration Fatigue	60Hz, 20G 32hrs/X, Y, Z	110	0	167	0
Vibration Variable Freq.	20 ~ 2000Hz 20G 4 min/X, Y, Z	110	0	167	0
Lead Integrity	Bending Tention Fatigue	65 pins	0	102 pins	0

3.3 Reliability Test Results on 16-bit MPU

Table 8 Reliability Test Results on 16-bit MPU HD68000 (Side-braze)

Test Items	Condition	Sample Size	Failures
Operation Life Test (1)	Ta = 125°C, V_{CC} = 5.5V 1000hrs	30	0
Operation Life Test (2)	Ta = 150°C, V_{CC} = 5.5V 1000hrs	20	0
High Temperature Storage	Ta = 295°C, 1000hrs	20	0
Temperature Cycling (1)	-55°C ~ 150°C, 10 cycles	105	0
Temperature Cycling (2)	-20°C ~ 125°C, 500 cycles	45	0
Thermal Shock	-55°C ~ 125°C, 15 cycles	22	0
Soldering Heat	260°C, 10 sec	22	0
Mechanical Shock	1500G, 5 msec 3 times/X, Y, Z	22	0
Vibration Variable Freq.	20G, 100 ~ 2000Hz 3 times/X, Y, Z	22	0
Constant Acceleration	20000G, 1 min/X, Y, Z	22	0
Solderability	230°C, 5 sec	22	0

4. QUALITY CONTROL IN PROCESS

Process quality control plays an extremely important role in quality assurance for semiconductor devices. Examples of control items are shown in Figure 3 and Figure 4.

Flow	Process	Inspection & QC Item
	Wafer process	
	Probe test	100% Electrical test
	Dicing/Break	
	Chip visual	100% visual inspection
	Die bonding	Patroll inspection once/day/machine visual
	Wire bonding	once/day/machine visual once/week/machine bond dimension bond strength
	Internal visual	100% visual inspection
	Molding	
	Temperature cycling	100%, Cond. -55°C ~ 150°C, 10 cycles
	Lead trim & form	
	Solder dipping	
	External visual	100% visual inspection
	Mark	
	Burn-in	100% Burn-in, 125°C static
	Electrical test	100% DC, AC & Functional test
	External visual	100% visual inspection
	Lot acceptance	
	Warehouse	

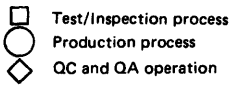


Figure 3 Quality Plan in Plastic Package Process

Flow	Process	Inspection & QC Item
	Wafer process	
	Probe test	100% Electrical test
	Dicing/Break	
	Chip visual	100% visual inspection
	Die bonding	Patrol inspection once/day/machine Visual
	Wire bonding	Patrol inspection once/day/machine visual once/week/machine bond dimention bond strength
	Internal visual	100% visual inspection
	Seal	
	Temperature cycling	100%, Cond. -55°C ~ 150°C, 10 cycles
	Leak test	100% Fine leak (He) 100% Gross leak (Bubble)
	Plating	
	Lead cutting	
	External visual	100% visual inspection
	Mark	
	Burn-in	100% Burn-in, 125°C static
	Electrical test	100% DC, AC & Functional test
	External visual	100% visual inspection
	Lot acceptance	
	Warehouse	

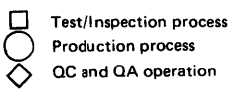


Figure 4 Quality Plan in Side-braze Package Process

5. QUALITY DATA FROM FIELD USE

Field failure rate is estimated in advance through production process evaluation and reliability tests. Past field data on similar devices provides the basis for this estimation. Quality information from the users are indispensable to the improvement of products quality. Therefore, field data on products delivered to the users are followed up carefully. On the basis of information furnished by the user, failure analysis is conducted and the results are quickly fed back to the design and production divisions.

Failure analysis result on 8-bit microcomputer returned to Hitachi from April '80 to March '81 is shown in Figure 5.

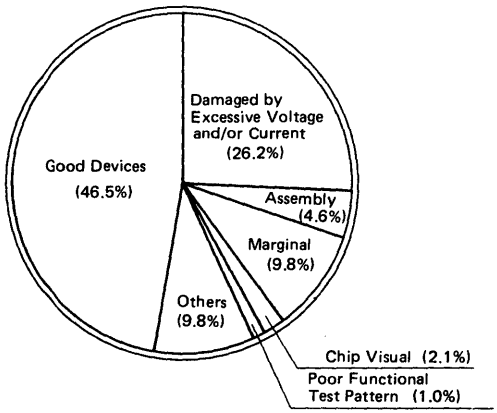


Figure 5 Failure Analysis Result

6. PRECAUTION

6.1 Storage

It is preferable to store semiconductor devices in the following ways to prevent detrioration in their electrical characteristics, solderability, and appearance, or breakage.

(1) Store in an ambient temperature of 5 to 30°C, and in a relative humidity of 40 to 60%.
(2) Store in a clean air environment, free from dust and active gas.
(3) Store in a container which does not induce static electricity.
(4) Store without any physical load.
(5) If semiconductor devices are stored for a long time, store them in the unfabricated form. If their lead wires are formed beforehand, bent parts may corrode during storage.
(6) If the chips are unsealed, store them in a cool, dry, dark, and dustless place. Assemble them within 5 days after unpacking. Storage in nitrogen gas is desirable. They can be stored for 20 days or less in dry nitrogen gas with a dew point at -30°C or lower. Unpacked devices must not be stored for over 3 months.
(7) Take care not to allow condensation during storage due to rapid temperature changes.

6.2 Transportation

As with storage methods, general precautions for other electronic component parts are applicable to the transportation of semiconductors, semiconductor-incorporating units and other similar systems. In addition, the following considerations must be given, too:

(1) Use containers or jigs which will not induce static electricity as the result of vibration during transportation. It is desirable to use an electrically conductive container or aluminium foil.
(2) In order to prevent device breakage from clothes-induced static electricity, workers should be properly grounded with a resistor while handling devices. The resistor of about 1 M ohm must be provided near the worker to protect from electric shock.
(3) When transporting the printed circuit boards on which semiconductor devices are mounted, suitable preventive measures against static electricity induction must be taken; for example, voltage built-up is prevented by shorting terminal circuit. When a belt conveyor is used, prevent the conveyor belt from being electrically charged by applying some surface treatment.
(4) When transporting semiconductor devices or printed circuit boards, minimize mechanical vibration and shock.

6.3 Handling for Measurement

Avoid static electricity, noise and surge-voltage when semiconductor devices are measured. It is possible to prevent breakage by shorting their terminal circuits to equalize electrical potential during transportation. However, when the devices are to be measured or mounted, their terminals are left open to provide the possibility that they may be accidentally touched by a worker, measuring instrument, work bench, soldering iron, belt conveyor, etc. The device will fail if it touches something which leaks current or has a static charge. Take care not to allow curve tracers, synchroscopes, pulse generators, D.C. stabilizing power supply units etc. to leak current through their terminals or housings.

Especially, while the devices are being tested, take care not to apply surge voltage from the tester, to attach a clamping circuit to the tester, or not to apply any abnormal voltage through a bad contact from a current source.

During measurement, avoid miswiring and short-circuiting. When inspecting a printed circuit board, make sure that no soldering bridge or foreign matter exists before turning on the power switch.

Since these precautions depend upon the types of semiconductor devices, contact Hitachi for further details.

6.4 Soldering

Semiconductor devices should not be left at high temperatures for a long time. Regardless of the soldering method, soldering must be done in a short time and at the lowest possible temperature. Soldering work must meet soldering heat test conditions, namely, 260°C for 10 seconds and 350°C for 3 seconds at a point 1 to 1.5 mm away from the end of the device package.

Use of a strong alkali or acid flux may corrode the leads, deteriorating device characteristics. The recommended soldering iron is the type that is operated with a secondary voltage supplied by a transformer and grounded to protect from lead current. Solder the leads at the farthest point from the device package.

6.5 Removing Residual Flux

To insure the reliability of electronic systems, residual flux must be removed from circuit boards. Detergent or ultrasonic cleaning is usually applied. If chloric detergent is used for the plastic molded devices, package corrosion may occur. Since cleaning over extended periods or at high temperatures will

cause swollen chip coating due to solvent permeation, select the type of detergent and cleaning condition carefully. Lotus Solvent and Dyfron Solvent are recommended as a detergent. Do not use any trichloroethylene solvent. For ultrasonic cleaning, the following conditions are advisable:

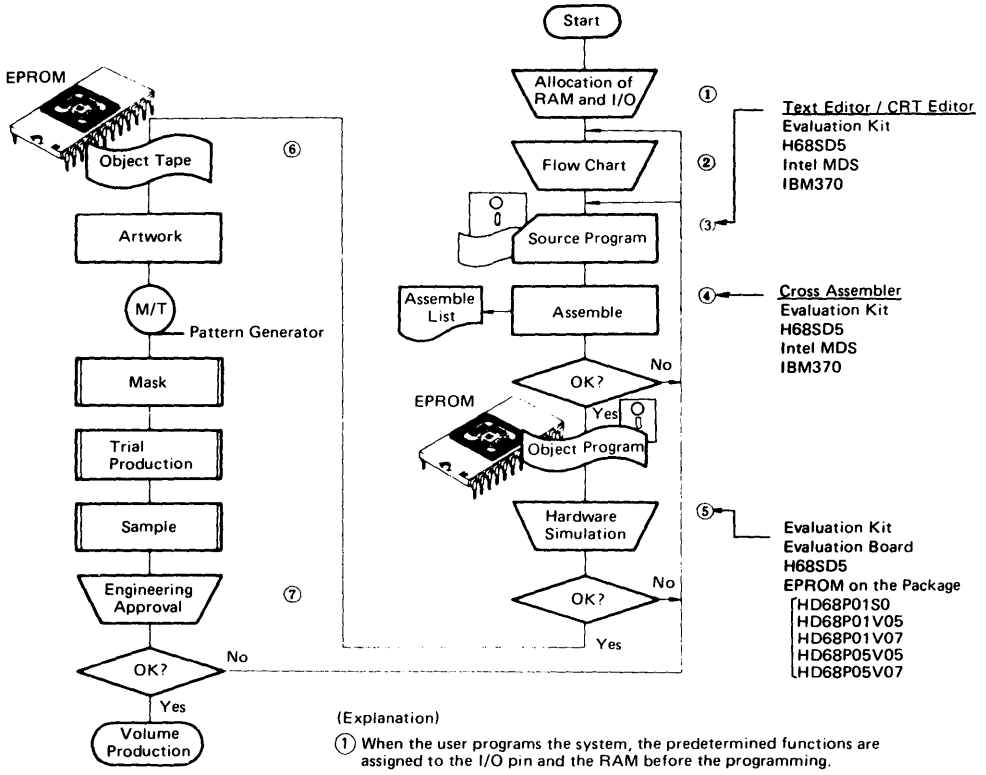
- Frequency: 28 to 29 kHz (to avoid device resonation)
- Ultrasonic output: 15W/ℓ
- Keep the devices out of direct contact with the power generator.
- Cleaning time: Less than 30 seconds

DESIGN PROCEDURE AND SUPPORT TOOLS FOR 8-BIT SINGLE-CHIP MICROCOMPUTER

The cross assembler and the hardware simulator using various types of computer are prepared by the company as supporting systems to develop user's programs.

User's programs are mask programed into the ROM and delivered as the LSI by the company.

Fig. 1 shows the typical program design procedure and Table 1 shows the system development support tools for the HD6801 and HD6805 Family which are used in these processes.



(Explanation)

① When the user programs the system, the predetermined functions are assigned to the I/O pin and the RAM before the programming.

② A flow chart is designed to achieve the predetermined functions and the flow chart is coded by using the prenumeric code.

③ The coded flow chart is punched into the card or the paper tape or written into the floppy disk, to generate a source program.

④ The source program is assembled by the resident system (evaluation kit) or the cross system, to generate the object program. In this case, errors during the assembling are also detected.

⑤ Hardware simulation is performed to confirm the program. The company provides three kinds of hardware, H68SD5, the evaluation kit and the evaluation board. The consumers are able to choose the best suitable tool.

⑥ The completed program is sent to the company in the form of EPROM or the object tape.

⑦ Options such as ROM is masked by the company, LSI is testatively produced and the sample is handed in to the user. After the user has evaluated the sample and confirmed that the program is correct, mass production is started.

Figure 1 Program Design Procedure

Table 1 System Development Support Tools

Type No.	Resident System				Cross System	
	Evaluation Kit	Evaluation Board	EPROM on the Package	H68SD5 + Emulator Set (Hardware + Software)	HITAC M / IBM370	Intel MDS220/230
HD6801S0 HD6801S5	H61EVT2* (Hardware) + S31MIX-R (Software)	H61EV00*	HD68P01S0	H68SD5 + H61MIX1*	○	○
HD6801V0 HD6801V5	H61EVT2* (Hardware) + S31MIX1-R (Software)	H61EV01*	HD68P01V05 HD68P01V07	H68SD5 + H61MIX1*	○	○
HD6805S1	H65EVT2 (Hardware) + S65MIX1-R (Software)	H65EV00*	HD68P05V05 HD68P05V07	H68SD5 + H65MIX1		○
HD6805U1 HD6805V1	H65EVT2 (Hardware) + S65MIX1-R (Software)	H65EV01*	HD68P05V05 HD68P05V07	H68SD5 + H65MIX1		○
HD6805W0	H65EVT3* (Hardware) + S65MIX1-R (Software)	H65EV02*	–	H68SD5 + H65MIX2		○
HD6301V0 HD63A01V0 HD63B01V0	H31EVT1* (Hardware) + S31MIX1-R (Software)	H31EV00*	–	H68SD5 + H31MIX1*	○	○
HD63L05	H3L5EVT1* (Hardware) + S65MIX1-R (Software)	H3L5EV00*	–	H68SD5 + H3L5MIX1*		○

* Under development

DESIGN PROCEDURE AND SUPPORT TOOLS FOR 8-BIT SINGLE-CHIP MICROCOMPUTER

■ Single-Chip Microcomputer Development System

The H68SD5 is a development system for Hitachi 4-bit and 8-bit single-chip microcomputers.

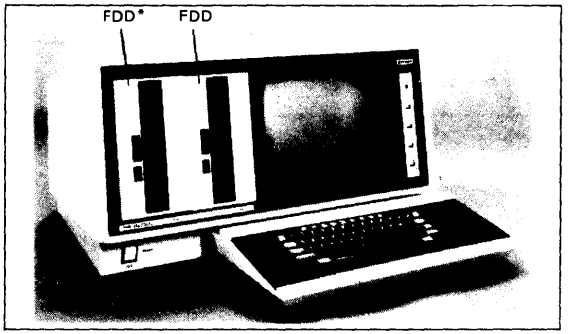
It is an all-in-one type compact HD6800 based CRT/Key board microcomputer terminal with one Floppy disk driver and has standard interface for the TTY (RS-232C or TTL level) and printer (Centronics parallel interface). The EPROM Writer and the second Floppy disk driver are optionally available.

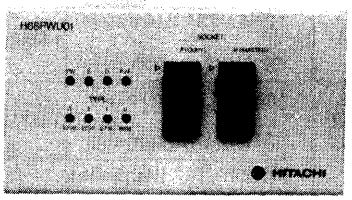
Features

- Supports the system development for 8-bit and 4-bit single chip microcomputers – HD6801/6805 family and HMCS40 series.
- Disk based low cost system
- Provides the CRT Editor, Assembler, Emulator and EPROM Writer controlled by FDOS-III
- 56k-byte RAMs
- Allows linking between the H68SD5 and the I/O devices (TTY and Printer)
- Easy to debug user's prototype system using the Emulator Module

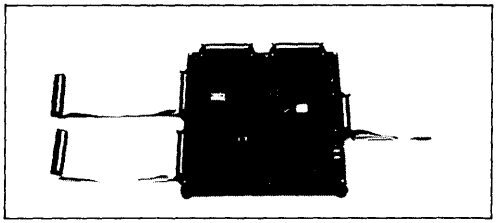
System Configuration

H68SD5





EPROM Writer*



Emulator Module* { HD6801/6805 family / HMCS40 series }

* Option

HMCS6800 FAMILY INSTRUCTION SET

● Accumulator and Memory Operations

Operations	Mnemonic	Boolean/ Arithmetic Operation	HD6801S HD6801V HD6803						HD6805S HD6805U HD6805V HD6805W						HD6301V						HD63L05						HD6800 HD6802 HD6802W						HD6809 HD6809E							
			IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	EXT INDIRECT	IMPL	RELATIVE
Add	ADDA	A+M→A	○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○	○		
	ADDB	B+M→B	○	○	○	○									○	○	○	○									○	○	○	○			○	○	○	○	○		
Add Double	ADDD	A:B+M:M+1 →A:B	○	○	○	○									○	○	○	○															○	○	○	○	○		
Add Accumulators	ABA	A+B→A					○												○												○								
Add with Carry	ADCA	A+M+C→A	○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○	○		
	ADCB	B+M+C→B	○	○	○	○									○	○	○	○									○	○	○	○			○	○	○	○	○		
Subtract	SUBA	A−M→A	○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○	○		
	SUBB	B−M→B	○	○	○	○									○	○	○	○									○	○	○	○			○	○	○	○	○		
Subtract Double	SUBD	A:B−M:M+1 →A:B	○	○	○	○									○	○	○	○															○	○	○	○	○		
Subtract Accumulator	SBA	A−B→A					○												○												○								
Subtract with Carry	SBCA	A−M−C→A	○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○	○		
	SBCB	B−M−C→B	○	○	○	○									○	○	○	○									○	○	○	○			○	○	○	○	○		
Multiply	MUL	A×B→A:B					○												○																			○	
Decimal Adjust A	DAA	Decimal Adjust Accumulator					○												○												○							○	
Increment	INC	M+1→M			○	○				○	○						○	○				○	○						○	○				○	○	○	○			
	INCA	A+1→A					○						○						○						○						○							○	
	INCB	B+1→B					○												○												○							○	
Decrement	DEC	M−1→M			○	○				○	○						○	○				○	○						○	○				○	○	○	○			
	DECA	A−1→A					○						○						○						○						○							○	
	DECB	B−1→B					○												○												○							○	
Clear	CLR	0→M			○	○				○	○						○	○				○	○						○	○				○	○	○	○			
	CLRA	0→A					○						○						○						○						○							○	
	CLRB	0→B					○												○												○							○	
Compare	CMPA	A−M	○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○	○		
	CMPB	B−M	○	○	○	○									○	○	○	○									○	○	○	○			○	○	○	○	○		
Compare Double	CMPD	A:B−M:M+1																															○	○	○	○	○		
Compare Accumulators	CBA	A−B					○												○												○								
Test Zero or Minus	TST	M−00			○	○				○	○						○	○				○	○						○	○				○	○	○	○			
	TSTA	A−00					○						○						○						○						○							○	
	TSTB	B−00					○												○												○							○	
And	ANDA	A · M→A	○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○	○		
	ANDB	B · M→B	○	○	○	○									○	○	○	○									○	○	○	○			○	○	○	○	○		
Or	ORAA ORA	A+M→A	○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○	○		
	ORAB ORB	B+M→B	○	○	○	○									○	○	○	○									○	○	○	○			○	○	○	○	○		
Exclusive Or	EORA	A⊕M→A	○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○	○		
	EORB	B⊕M→B	○	○	○	○									○	○	○	○									○	○	○	○			○	○	○	○	○		
Complement 1's	COM	$\overline{M}$→M			○	○				○	○						○	○				○	○						○	○				○	○	○	○			
	COMA	$\overline{A}$→A					○						○						○						○						○							○	
	COMB	$\overline{B}$→B					○												○												○							○	
Complement 2's (Negate)	NEG	00−M→M			○	○				○	○						○	○				○	○						○	○				○	○	○	○			
	NEGA	00−A→A					○						○						○						○						○							○	
	NEGB	00−B→B					○												○												○							○	
Bit Test	BITA	A · M	○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○	○		
	BITB	B · M	○	○	○	○									○	○	○	○									○	○	○	○			○	○	○	○	○		

(to be continued)

HMCS6800 FAMILY INSTRUCTION SET

Operation	Mnemonic	Boolean/ Arithmetic Operation	HD6801S HD6801V HD6803						HD6805S HD6805U HD6805V HD6805W						HD6301V						HD63L05						HD6800 HD6802 HD6802W						HD6809 HD6809E							
			IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	EXT INDIRECT	IMPL	RELATIVE
Bit Clear	BCLR n	0→M·Bit(n)								○												○																	
Bit Set	BSET n	1→M·Bit (n)								○												○																	
	SEX	Sign Extend B into A																																					○	
Load	LDAA LDA	M→A	○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○	○			
	LDAB LDB	M→B	○	○	○	○									○	○	○	○									○	○	○	○			○	○	○	○	○			
	LDD	M:M+1→A:B	○	○	○	○									○	○	○	○															○	○	○	○	○			
Store	STAA STA	A→M		○	○	○				○	○	○				○	○	○				○	○	○				○	○	○				○	○	○	○			
	STAB STB	B→M		○	○	○										○	○	○										○	○	○				○	○	○	○			
	STD	A:B→M:M+1		○	○	○										○	○	○																○	○	○	○			
Transfer	TFR	Register 1 →Register 2																																					○	
	TAB	A→B					○												○												○									
	TBA	B→A					○												○												○									
Exchange	EXG	Register 1 →Register 2																																					○	
Push Data	PSHA	A→Ms S−1→S					○												○												○									
	PSHB	B→Ms S−1→S					○												○												○									
	PSHS	Registers→Ms S−n→S																																					○	
	PSHU	Registers→Mu U−n→U																																					○	
Pull Data	PULA	Ms→A S+1→S					○												○												○									
	PULB	Ms→B S+1→S					○												○												○									
	PULS	Ms→Registers S+n→S																																					○	
	PULU	Mu→Registers U+n→U																																					○	
Shift Left Arithmetic	ASL	M			○	○				○	○						○	○				○	○						○	○					○	○	○	○		
	ASLA	A					○					○							○					○							○								○	
	ASLB	B					○												○												○								○	
	ASLD	A:B					○												○																					
Shift Right Arithmetic	ASR	M			○	○				○	○						○	○				○	○						○	○					○	○	○	○		
	ASRA	A					○					○							○					○							○								○	
	ASRB	B					○												○												○								○	
Shift Left Logical	LSL	M								○	○						○	○				○	○						○	○					○	○	○	○		
	LSLA	A										○							○					○							○								○	
	LSLB	B																	○												○								○	
Shift Right Logical	LSR	M			○	○				○	○						○	○				○	○						○	○					○	○	○	○		
	LSRA	A					○					○							○					○							○								○	
	LSRB	B					○												○												○								○	
	LSRD	A:B					○												○																					
Rotate Left	ROL	M			○	○				○	○						○	○				○	○						○	○					○	○	○	○		
	ROLA	A					○					○							○					○							○								○	
	ROLB	B					○												○												○								○	
Rotate Right	ROR	M			○	○				○	○						○	○				○	○						○	○					○	○	○	○		
	RORA	A					○					○							○					○							○								○	
	RORB	B					○												○												○								○	

(to be continued)

Operations	Mnemonic	Boolean / Arithmetic Operation	HD6801S HD6801V HD6803						HD6805S HD6805U HD6805V HD6805W						HD6301V						HD63L05						HD6800 HD6802 HD6802W						HD6809 HD6809E						
			IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	EXT INDIRECT	IMPL	RELATIVE
And Immediate	AIM	M·IMMED→M														○	○																					
OR Immediate	OIM	M+MMED→M														○	○																					
EOR Immediate	EIM	M⊕IMMED→M														○	○																					
Test Immediate	TIM	M·IMMED														○	○																					

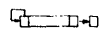
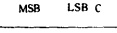
●Jump and Branch Instructions

Operations	Mnemonic	Branch Test	HD6801S HD6801V HD6803						HD6805S HD6805U HD6805V HD6805W						HD6301V						HD63L05						HD6800 HD6802 HD6802W						HD6809 HD6809E						
			IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	EXT INDIRECT	IMPL	RELATIVE
Branch if Carry Clear	BCC	C=0						○						○						○						○						○						○
	LBCC																																					○
Branch if Carry Set	BCS	C=1						○						○						○						○						○						○
	LBCS																																					○
Branch if=Zero	BEQ	Z=1						○						○						○						○						○						○
	LBEQ																																					○
Branch if Not Equal Zero	BNE	Z=0						○						○						○						○						○						○
	LBNE																																					○
Branch if≧Zero	BGE	N⊕V=0						○												○												○						○
	LBGE																																					○
Branch if>Zero	BGT	Z+(N⊕V)=0						○												○												○						○
	LBGT																																					○
Branch if<Zero	BLT	N⊕V=1						○												○												○						○
	LBLT																																					○
Branch if≦Zero	BLE	Z+(N⊕V)=1						○												○												○						○
	LBLE																																					○
Branch if Higher	BHI	C+Z=0						○						○						○						○						○						○
	LBHI																																					○
Branch if Higher or same(unsigned)	BHS	C=0						○						○												○						○						○
	LBHS																																					○
Branch if Lower or Same	BLS	C+Z=1						○						○						○						○						○						○
	LBLS																																					○
Branch if Lower (unsigned)	BLO	C=1						○						○												○						○						○
	LBLO																																					○
Branch if Plus	BPL	N=0						○						○						○						○						○						○
	LBPL																																					○
Branch if Minus	BMI	N=1						○						○						○						○						○						○
	LBMI																																					○
Branch if Overflow Clear	BVC	V=0						○												○												○						○
	LBVC																																					○
Branch if Overflow Set	BVS	V=1						○												○												○						○
	LBVS																																					○
Branch if Half Carry Clear	BHCC	H=0						○						○												○						○						
Branch if Half Carry Set	BHCS	H=1						○						○												○						○						
Branch if Interrupt Mask Clear	BMC	I=0						○						○												○						○						
Branch if Interrupt Mask Set	BMS	I=1						○						○												○						○						
Branch if Interrupt Line High	BIH	INT=1						○						○												○						○						

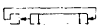
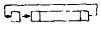
(to be continued)

Operations	Mnemonic	Branch Test	HD6801S HD6801V HD6803						HD6805S HD6805U HD6805V HD6805W						HD6301V						HD63L05						HD6800 HD6802 HD6802W						HD6809 HD6809E							
			IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	EXT INDIRECT	IMPL	RELATIVE
Branch if Interrupt Line Low	BIL	INT=0												○												○													
Branch if Bit n of M Clear	BRCLR n	M(n)=0												○												○													
Branch if Bit n of M Set	BRSET n	M(n)=1												○												○													
Branch Always	BRA							○						○						○						○						○							○
	LBRA																																						○	○
Branch Never	BRN							○						○						○						○													○
	LBRN																																						○	○
Branch to Subroutine	BSR							○						○						○						○						○							○
	LBSR																																						○	○
Jump	JMP				○	○				○	○	○					○	○				○	○	○					○	○				○	○	○	○	○		
Jump to Subroutine	JSR			○	○	○				○	○	○				○	○	○				○	○	○					○	○				○	○	○	○	○		
Return from Subroutine	RTS						○						○						○						○						○						○		
No Operation	NOP						○						○						○						○						○						○		
Software Interrupt	SWI						○						○						○						○						○						○	○	
	SWI2																																						○	○
	SWI3																																						○	○
Return from Interrupt	RTI						○						○						○						○						○						○	○	
Wait	WAI						○												○						○						○								
	CWAI																																						○	
Synchronize to Interrupt	SYNC																																						○	
Sleep	SLP																		○																				

●Index Register and Stack Pointer Instructions

Operations	Mnemonic	Boolean/ Arithmetic Operation	HD6801S HD6801V HD6803						HD6805S HD6805U HD6805V HD6805W						HD6301V						HD63L05						HD6800 HD6802 HD6802W						HD6809 HD6809E							
			IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	EXT INDIRECT	IMPL	RELATIVE
Increment	INX	X+1→X					○						○						○						○						○								
Decrement	DEX	X−1→X					○						○						○						○						○								
ADD with B	ABX	B+X→X					○												○																			○	
Clear	CLRX	0→X											○												○														
Negate	NEGX	00−X→X											○												○														
Complement 1's	COMX	FF−X→X											○												○														
Shift Left Arithmetic	ASLX	C MSB LSB											○												○														
Shift Right Arithmetic	ASRX	 MSB LSB C											○												○														
Shift Right Logical	LSRX	 MSB LSB C											○												○														

(to be continued)

Operations	Mnemonic	Boolean/ Arithmetic Operation	HD6801S HD6801V HD6803						HD6805S HD6805U HD6805V HD6805W						HD6301V						HD63L05						HD6800 HD6802 HD6802W						HD6809 HD6809E						
			IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	EXT INDIRECT	IMPL	RELATIVE
Shift Left Logical	LSLX	 C MSB LSB											○						○						○						○								
Rotate Left	ROLX	 C MSB LSB											○						○						○						○								
Rotate Right	RORX	 C MSB LSB											○						○						○						○								
Test	TSTX	X−00											○						○						○						○								
Compare	CPX CMPX	X−M:M+1	○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○	○		
	CMPY	Y−M:M+1																															○	○	○	○	○		
Load	LDX	M:M+1→X	○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○			○	○	○	○	○		
	LDY	M:M+1→Y																															○	○	○	○	○		
Store	STX	X→M:M+1		○	○	○				○	○	○				○	○	○				○	○	○				○	○	○				○	○	○	○		
	STY	Y−M:M+1																																○	○	○	○		
Load effective Address	LEAX	Effective Address→X																																	○				
	LEAY	Effective Address→Y																																	○				
Push	PSHX	X→Ms					○											○																					
Pull	PULX	Ms→X					○											○																					
Transfer X,A	TAX	A→X											○																		○								
	TXA	X→A											○																		○								
Transfer X,S	TSX	S→X					○											○														○							
	TXS	X→S					○											○														○							
Increment	INS	S+1→S					○											○														○							
Decrement	DES	S−1→S					○											○														○							
Reset	RSP	$7F→S											○						○						○														
Compare	CMPS	S−M:M+1																															○	○	○	○	○		
	CMPU	U−M:M+1																															○	○	○	○	○		
Load	LDS	M:M+1→S	○	○	○	○									○	○	○	○									○	○	○	○			○	○	○	○	○		
	LDU	M:M+1→U																															○	○	○	○	○		
Store	STS	S→M:M+1		○	○	○										○	○	○										○	○	○				○	○	○	○		
	STU	U→M:M+1																																○	○	○	○		
Load effective Address	LEAS	Effective Address→S																																	○				
	LEAU	Effective Address→U																																	○				
Exchange	XGDX	ACCD↔X																○																					

●Condition Code Register Instructions

Operations	Mnemonic	Boolean Operation	HD6801S HD6801V HD6803						HD6805S HD6805U HD6805V HD6805W						HD6301V						HD63L05						HD6800 HD6802 HD6802W						HD6809 HD6809E						
			IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	EXT INDIRECT	IMPL	RELATIVE
Clear Carry	CLC	0→C					○						○						○						○						○								
Clear Interrupt Mask	CLI	0→I					○						○						○						○						○								

(to be continued)

HMCS6800 FAMILY INSTRUCTION SET

Operations	Mnemonic	Boolean Operation	HD6801S HD6801V HD6803						HD6805S HD6805U HD6805V HD6805W						HD6301V						HD63L05						HD6800 HD6802 HD6802W						HD6809 HD6809E						
			IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	IMPL	RELATIVE	IMMED	DIRECT	INDEX	EXTND	EXT INDIRECT	IMPL	RELATIVE
Clear Overflow	CLV	0→V					○						○						○						○						○								
Set Carry	SEC	1→C					○						○						○						○						○								
Set Interrupt Mask	SEI	1→I					○						○						○						○						○								
Set Overflow	SEV	1→V					○												○												○								
Transfer A, CC	TAP	A→CC					○												○												○								
	TPA	CC→A					○												○												○								
And CC	AND CC	CC·imm→CC																															○						
Or CC	ORCC	CC+imm→CC																															○						

LEGEND

A	Accumulator A	H	Half Carry from 3 bit
B	Accumulator B	I	Interrupt Mask
X	Index Register	N	Negative
Y	Index Register (6809 only)	Z	Zero
S	Stack Pointer	V	Overflow
U	User Stack Pointer (6809 only)	C	Carry Bit from 7 bit
→	Transfer into	MSB	Most Signification bit
+	Arithmetic Plus	LSB	Least Signification bit
−	Arithmetic Minus	IMMED	Immediate
·	Boolean AND	DIRECT	Direct
+	Boolean Inclusive OR	INDEX	Indexed
⊕	Boolean Exclusive OR	EXTND	Extended
:	Connection Area	EXT INDIRECT	Extended indirect
$\overline{M}$	Complement of M	IMPL	Implied (Inherent, Accumulator)
Ms	Stack area pointed by S.	RELATIVE	Relative
Mu	Stack area pointed by U.		
CC	Condition Code Register		

8-BIT MICROCOMPUTERS FOR INDUSTRIAL APPLICATION

For industrial application which needs wider operating temperature range (from -40°C to +85°C), Hitachi has the following devices for both the 8-bit single-chip microcomputers and the 8-bit multi-chip microcomputers.

"J" indicates industrial grade devices (Example HD6801S0PJ).

• 8-BIT Single-chip Microcomputer Characteristics

Type No.		HD6801S0PJ HD6801S0CJ	HD6801V0PJ	HD6803PJ HD6803CJ	HD6805S1PJ	HD6805U1PJ	HD6805V1PJ	HD6805W0PJ*	HD6301V0PJ* HD63A01V0PJ*
Process		NMOS	NMOS	NMOS	NMOS	NMOS	NMOS	NMOS	CMOS
Package		DP-40 DC-40	DP-40	DP-40 DC-40	DP-28	DP-40	DP-40	DP-40	DP-40
Electrical** Characteristics	V_{CC}	5V ± 0.25V	5V ± 0.25V	5V ± 0.25V	5.25V ± 0.5V	5.25V ± 0.5V	5.25V ± 0.5V	5.25V ± 0.5V	5V ± 0.5V
	Topr	-40~+85°C	-40~+85°C	-40~+85°C	-40~+85°C	-40~+85°C	-40~+85°C	-40~+85°C	-40~+85°C
	P_D				800mW	800mW	800mW		
	V_{IH} (EXTAL)	2.2V	2.2V	2.2V					
	$\|I_{in}\|$ (EXTAL)	1.2mA	1.2mA	1.2mA					
	I_{SBB}	10mA	10mA	10mA					
Function: Memory	ROM (k Byte)	2	4	–	1.1	2	4	4	4
	RAM (Byte)	128	128	128	64	96	96	96	128
I/O		29	29	13	20	32	32	29	29
Timer (bit)		16	16	16	8***	8***	8***	8***x2	16
SCI		Yes	Yes	Yes	No	No	No	No	Yes
Other Features		• Data Retention Capability	• Data Retention Capability	• Multiplexed Address and Data	• Vectored Interrupts • Self-check Mode • Master Reset	• Voltage Comparator • Vectored Interrupts • Self-check Mode • Master Reset	• Voltage Comparator • Vectored Interrupts • Self-check Mode • Master Reset	• Voltage Comparator • Vectored Interrupts • Self-check Mode • Master Reset	• Sleep Operation • Low power Consumption

* Preliminaly

** Electrical Characteristics shown here is for the industrial grade which is different from standard specification. So refer to each data sheet for details.

*** Timer; 8-Bit programmable timer with 7-Bit programmable pre-scaler.

• 8-BIT Multi-chip Microcomputer Characteristics

Type No.	Function	Electrical Characteristics**			Package
		V_{CC}	Topr	P_D	
HD6800PJ	Micro Processing Unit	5V ± 0.25V	-40 ~ +85°C		DP-40
HD6802PJ	Micro Processor with Clock and RAM (128 byte)	5V ± 0.25V	-40 ~ +85°C		DP-40
HD6802WPJ	Micro Processor with Clock and RAM (256 byte)	5V ± 0.25V	-40 ~ +85°C		DP-40
HD6809PJ* HD68A09PJ* HD68B09PJ*	Advanced Micro Processing Unit	5V ± 0.25V	-40 ~ +85°C		DP-40
HD6809EPJ* HD68A09EPJ* HD68B09EPJ*	Advanced Micro Processing Unit	5V ± 0.25V	-40 ~ +85°C		DP-40
HD6821PJ	Peripheral Interface Adapter	5V ± 0.25V	-40 ~ +85°C		DP-40
HD6840PJ HD68A40PJ HD68B40PJ	Programmable Timer Module	5V ± 0.25V	-40 ~ +85°C		DP-28
HD6846PJ	Combination ROM I/O Timer	5V ± 0.25V	-40 ~ +85°C	1000mW	DP-40
HD46508PJ HD46508PJ-1	Analog Data Acquisition Unit	5V ± 0.25V (Analog Input 0 ~ 5.0V)	-40 ~ +85°C		DP-40
HD46508PAJ HD46508PAJ-1	Analog Data Acquisition Unit	5V ± 0.25V (Analog Input 0 ~ 5.0V)	-40 ~ +85°C		DP-40

* Under development

** Electrical Characteristics shown here is for the industrial grade which is different from standard specification.

DATA SHEETS

8-BIT MICROCOMPUTER
HMCS6800 SINGLE-CHIP
SERIES

Preliminary data sheets herein contain information on new products. Specifications and information are subject to change without notice.

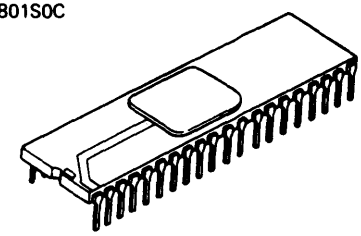
Advance Information data sheets herein contain information on a product under development. Hitachi reserves the right to change or discontinue these products without notice.

HD6801S0, HD6801S5

MCU (Microcomputer Unit)

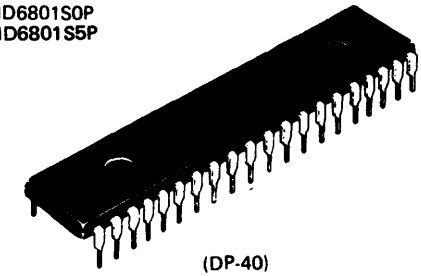
The HD6801S MCU is an 8-bit microcomputer system which is compatible with the HMCS6800 family of parts. The HD6801S MCU is object code compatible with the HD6800 with improved execution times of key instructions plus several new 16-bit and 8-bit instructions including an 8x8 unsigned multiply with 16-bit result. The HD6801S MCU can operate as a single-chip microcomputer or be expanded to 65k words. The HD6801S MCU is TTL compatible and requires one +5.0 volt power supply. The HD6801S MCU has 2k bytes of ROM and 128 bytes of RAM on chip. Serial Communications interface (S.C.I.), and parallel I/O as well as a three function 16-bit timer. Features and Block diagram of the HD6801S include the following:

HD6801S0C



(DC-40)

HD6801S0P
HD6801S5P

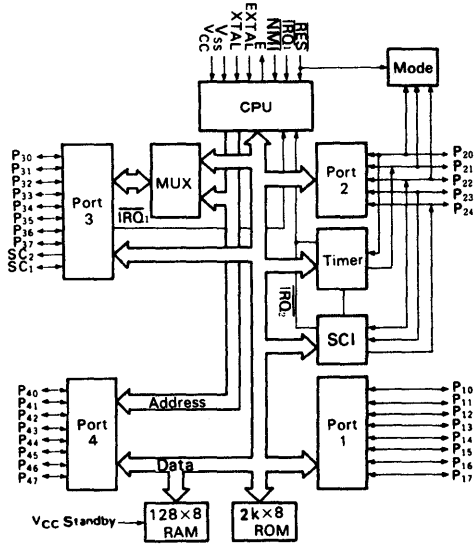


(DP-40)

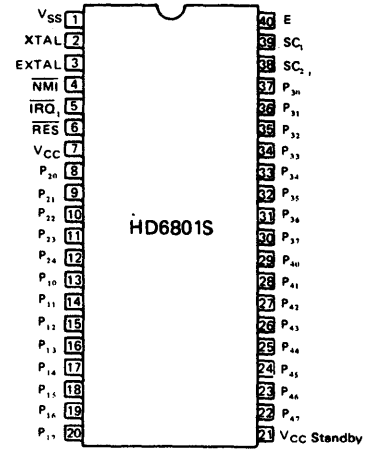
■ FEATURES

- Expanded HMCS6800 Instruction Set
- 8 x 8 Multiply
- On-Chip Serial Communications Interface (S.C.I.)
- Object Code Compatible With The HD6800 MPU
- 16-Bit Timer
- Single Chip Or Expandable To 65k Words
- 2k Bytes Of ROM
- 128 Bytes Of RAM (64 Bytes Retainable On Power Down)
- 29 Parallel I/O Lines And 2 Handshake Control Lines
- Internal Clock/Divided-By-Four Circuitry
- TTL Compatible Inputs And Outputs
- Interrupt Capability
- Compatible with MC6801

■ BLOCK DIAGRAM



■ PIN ARRANGEMENT



(Top View)

■ TYPE OF PRODUCTS

MCU	Bus Timing
HD6801S0	1 MHz
HD6801S5	1.25 MHz

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC} *	-0.3 ~ +7.0	V
Input Voltage	V_{in} *	-0.3 ~ +7.0	V
Operating Temperature	T_{opr}	0 ~ + 70	°C
Storage Temperature	T_{stg}	- 55 ~ +150	°C

* With respect to V_{SS} (SYSTEM GND)

[NOTE] Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ ELECTRICAL CHARACTERISTICS

• DC CHARACTERISTICS (V_{CC} =5.0V±5%, V_{SS} = 0V, Ta = 0 ~ +70°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Input "High" Voltage	$\overline{RES}$	V_{IH}		4.0	–	V_{CC}	V
	Other Inputs*			2.0	–	V_{CC}	
Input "Low" Voltage	All Inputs*	V_{IL}		-0.3	–	0.8	V
Input Load Current	P_{40} ~ P_{47}	$\|I_{in}\|$	V_{in} = 0 ~ 2.4V	–	–	0.5	mA
	SC_1			–	–	0.8	
	EXTAL		V_{in} = 0 ~ V_{CC}	–	–	0.8	
Input Leakage Current	$\overline{NMI}$, $\overline{IRQ_1}$, $\overline{RES}$	$\|I_{in}\|$	V_{in} = 0 ~ 5.25V	–	–	2.5	μA
Three State (Offset) Leakage Current	P_{10} ~ P_{17}, P_{30} ~ P_{37}	$\|I_{TSI}\|$	V_{in} = 0.5 ~ 2.4V	–	–	10	μA
	P_{20} ~ P_{24}			–	–	100	
Output "High" Voltage	P_{30} ~ P_{37}	V_{OH}	I_{LOAD} = -205 μA	2.4	–	–	V
	P_{40} ~ P_{47}, E, SC_1, SC_2		I_{LOAD} = -145 μA	2.4	–	–	
	Other Outputs		I_{LOAD} = -100 μA	2.4	–	–	
Output "Low" Voltage	All Outputs	V_{OL}	I_{LOAD} = 1.6 mA	–	–	0.5	V
Darlington Drive Current	P_{10} ~ P_{17}	$-I_{OH}$	V_{out} = 1.5V	1.0	–	10.0	mA
Power Dissipation		P_D		–	–	1200	mW
Input Capacitance	P_{30} ~ P_{37}, P_{40} ~ P_{47}, SC_1	C_{in}	V_{in} = 0V, Ta = 25°C, f = 1.0 MHz	–	–	12.5	pF
	Other Inputs			–	–	10.0	
V_{CC} Standby	Powerdown	V_{SBB}		4.0	–	5.25	V
	Operating	V_{SB}		4.75	–	5.25	
Standby Current	Powerdown	I_{SBB}	V_{SBB} = 4.0V	–	–	8.0	mA

*Except Mode Programming Levels.

• AC CHARACTERISTICS

BUS TIMING (V_{CC} = 5.0V±5%, V_{SS} = 0V, Ta = 0 ~ +70°C, unless otherwise noted.)

Item		Symbol	Test Condition	HD6801S0			HD6801S5			Unit
				min	typ	max	min	typ	max	
Cycle Time		t_{cyc}		1	–	10	0.8	–	10	μs
Address Strobe Pulse Width "High"		PW_{ASH}		200	–	–	150	–	–	ns
Address Strobe Rise Time		t_{ASr}		5	–	50	5	–	50	ns
Address Strobe Fall Time		t_{ASf}		5	–	50	5	–	50	ns
Address Strobe Delay Time		t_{ASD}		60	–	–	30	–	–	ns
Enable Rise Time		t_{Er}		5	–	50	5	–	50	ns
Enable Fall Time		t_{Ef}		5	–	50	5	–	50	ns
Enable Pulse Width "High" Time		PW_{EH}		450	–	–	340	–	–	ns
Enable Pulse Width "Low" Time		PW_{EL}		450	–	–	350	–	–	ns
Address Strobe to Enable Delay Time		t_{ASED}		60	–	–	30	–	–	ns
Address Delay Time		t_{AD}	Fig. 1	–	–	260	–	–	260	ns
Address Delay Time for Latch (f = 1.0MHz)		t_{ADL}	Fig. 2	–	–	270	–	–	260	ns
Data Set-up Write Time		t_{DSW}		225	–	–	115	–	–	ns
Data Set-up Read Time		t_{DSR}		80	–	–	70	–	–	ns
Data Hold Time	Read	t_{HR}		10	–	–	10	–	–	ns
	Write	t_{HW}		20	–	–	20	–	–	
Address Set-up Time for Latch		t_{ASL}		60	–	–	50	–	–	ns
Address Hold Time for Latch		t_{AHL}		20	–	–	20	–	–	ns
Address Hold Time		t_{AH}		20	–	–	20	–	–	ns
Peripheral Read Access Time	Non-Multiplexed Bus	(t_{ACCN})		–	–	(610)	–	–	(410)	ns
	Multiplexed Bus	(t_{ACCM})		–	–	(600)	–	–	(400)	
Oscillator stabilization Time		t_{RC}	Fig. 10	100	–	–	100	–	–	ms
Processor Control Set-up Time		t_{PCS}	Fig. 11	200	–	–	200	–	–	ns

PERIPHERAL PORT TIMING (V_{CC} = 5.0V ±5%, V_{SS} = 0V, Ta = 0 ~ +70°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Peripheral Data Setup Time	Port 1, 2, 3, 4	t_{PDSU}	Fig. 3	200	–	–	ns
Peripheral Data Hold Time	Port 1, 2, 3, 4	t_{PDH}	Fig. 3	200	–	–	ns
Delay Time, Enable Positive Transition to $\overline{OS3}$ Negative Transition		t_{OSD1}	Fig. 5	–	–	350	ns
Delay Time, Enable Positive Transition to $\overline{OS3}$ Positive Transition		t_{OSD2}	Fig. 5	–	–	350	ns
Delay Time, Enable Negative Transition to Peripheral Data Valid	Port 1, 2*, 3, 4	t_{PWD}	Fig. 4	–	–	400	ns
Delay Time, Enable Negative Transition to Peripheral CMOS Data Valid	Port 2**, 4	t_{CMOS}	Fig. 4	–	–	2.0	μs
Input Strobe Pulse Width		t_{PWIS}	Fig. 6	200	–	–	ns
Input Data Hold Time	port 3	t_{IH}	Fig. 6	50	–	–	ns
Input Data Set-up Time	Port 3	t_{IS}	Fig. 6	20	–	–	ns

*Except P_{21} **10kΩ pull up register required for Port 2

TIMER, SCI TIMING (V_{CC} = 5.0V ±5%, V_{SS} = 0V, Ta = 0 ~ +70°C, unless otherwise noted.)

Item	Symbol	Test Condition	min	typ	max	Unit
Timer Input Pulse Width	t_{PWT}		t_{cyc} +200	–	–	ns
Delay Time, Enable Positive Transition to Timer Out	t_{TOD}	Fig. 7	–	–	600	ns
SCI Input Clock Cycle	t_{Scyc}		1	–	–	t_{cyc}
SCI Input Clock Pulse Width	t_{PWSCK}		0.4	–	0.6	t_{Scyc}

MODE PROGRAMMING (V_{CC} = 5.0V ±5%, V_{SS} = 0V, Ta = 0 ~ +70°C, unless otherwise noted.)

Item	Symbol	Test Condition	min	typ	max	Unit
Mode Programming Input "Low" Voltage	V_{MPL}	Fig. 8	–	–	1.7	V
Mode Programming Input "High" Voltage	V_{MPH}		4.0	–	–	V
$\overline{RES}$ "Low" Pulse Width	PW_{RSTL}		3.0	–	–	t_{cyc}
Mode Programming Set-up Time	t_{MPS}		2.0	–	–	t_{cyc}
Mode Programming Hold Time	t_{MPH}		150	–	–	ns

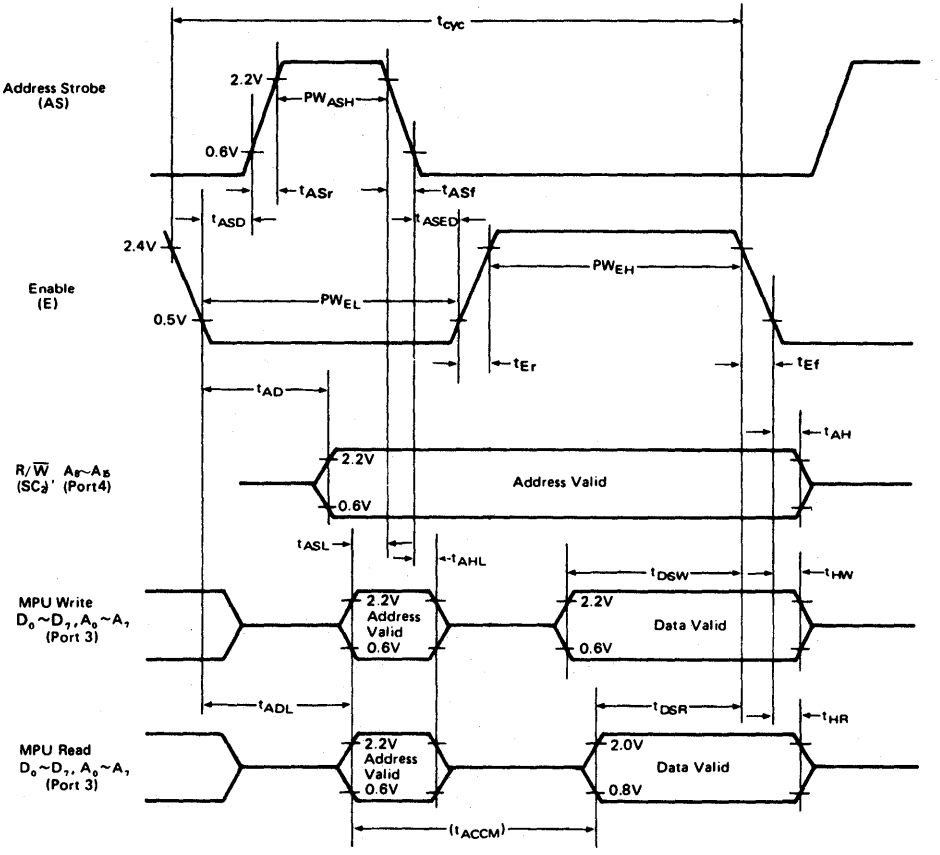


Figure 1 Expanded Multiplexed Bus Timing

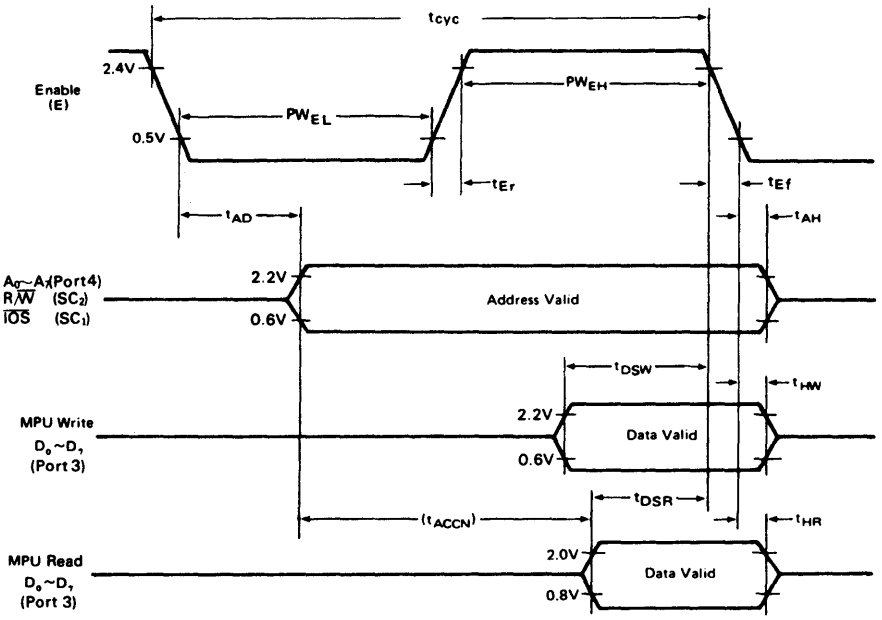
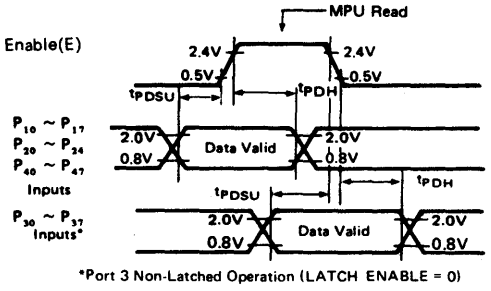
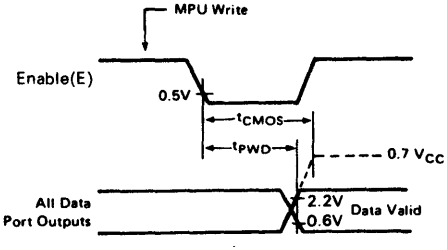


Figure 2 Expanded Non-Multiplexed Bus Timing



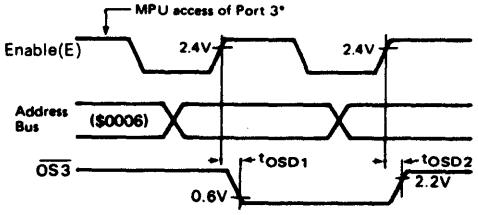
*Port 3 Non-Latched Operation (LATCH ENABLE = 0)

Figure 3 Data Set-up and Hold Times (MPU Read)



(NOTE)
1. 10 kΩ Pullup resistor required for Port 2 to reach 0.7 V_{CC}
2. Not applicable to P_{21}
3. Port 4 cannot be pulled above V_{CC}

Figure 4 Port Data Delay Timing (MPU Write)



*Access matches Output Strobe Select (OSS = 0, a read; OSS = 1, a write)

Figure 5 Port 3 Output Strobe Timing (Single Chip Mode)

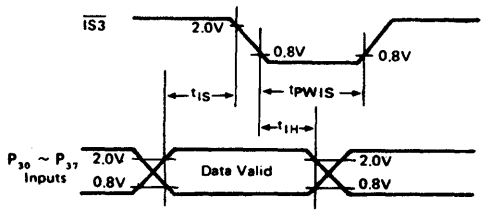


Figure 6 Port 3 Latch Timing (Single Chip Mode)

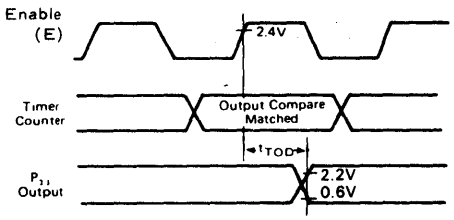


Figure 7 Timer Output Timing

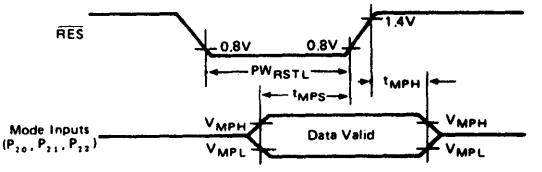
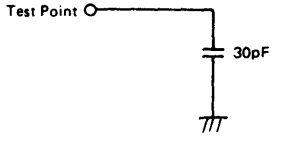
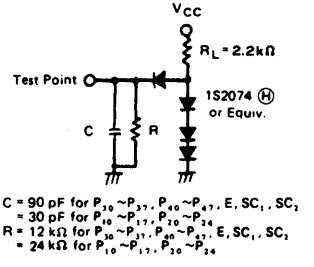


Figure 8 Mode Programming Timing



(a) CMOS Load



(b) TTL Load

Figure 9 Bus Timing Test Loads

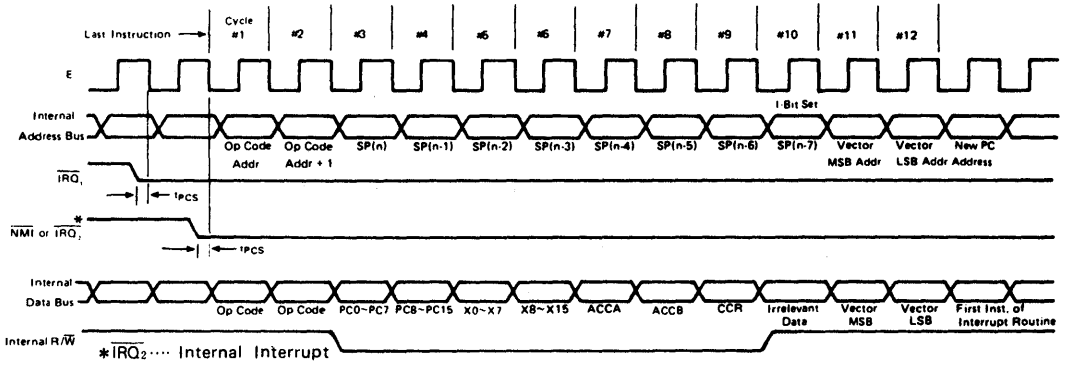


Figure 10 Interrupt Sequence

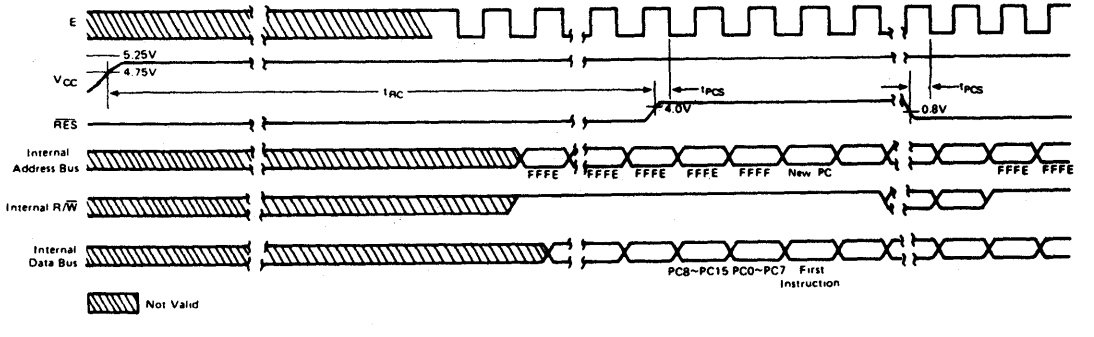


Figure 11 Reset Timing

■ SIGNAL DESCRIPTIONS

• Vcc and Vss

These two pins are used to supply power and ground to the chip. The voltage supplied will be +5 volts ±5%.

• XTAL and EXTAL

These connections are for a parallel resonant fundamental crystal, AT cut. Divide by 4 circuitry is included with the internal clock, so a 4 MHz crystal may be used to run the system at 1 MHz. The divide by 4 circuitry allows for use of the inexpensive 3.58 MHz Color TV crystal for non-time critical applications. Two 22pF capacitors are needed from the two crystal pins to ground to insure reliable operation. EXTAL may be driven by an external clock source at a 4 MHz rate to run at 1 MHz with a 40/60% duty cycle. It is not restricted to 4 MHz, as it will divide by 4 any frequency less than or equal to 4 MHz. XTAL must be grounded if an external clock is used. The following are the recommended crystal parameters:

Nominal Crystal Parameter

Item \ Crystal	4 MHz	5 MHz
Co	7 pF max.	4.7 pF max.
R_S	60Ω max.	30Ω typ.

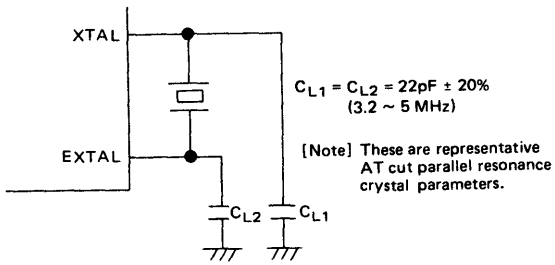


Figure 12 Crystal Interface

• Vcc Standby

This pin will supply +5 volts ±5% to the standby RAM on the chip. The first 64 bytes of RAM will be maintained in the power down mode with 8 mA current max. The circuit of figure 13 can be utilized to assure that V_{CC} Standby does not go below V_{SBB} during power down.

To retain information in the RAM during power down the following procedure is necessary:

1) Write "0" into the RAM enable bit, RAM E. RAM E is bit 6 of the RAM Control Register at location $0014. This disables the standby RAM, thereby protecting it at power down.
2) Keep V_{CC} Standby greater than V_{SBB}.

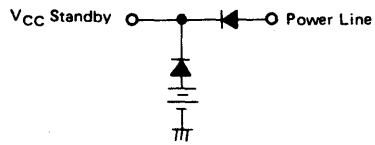


Figure 13 Battery Backup for V_{CC} Standby

• Reset ($\overline{RES}$)

This input is used to reset and start the MPU from a power down condition, resulting from a power failure or an initial startup of the processor. On power up, the reset must be held "Low" for at least 100 ms. During operation, $\overline{RES}$, when brought "Low", must be held "Low" at least 3 clock cycles.

When a "High" level is detected, the MPU does the following:

1) All the higher order address lines will be forced "High".
2) I/O Port 2 bits, 2, 1, and 0 are latched into programmed control bits PC2, PC1 and PC0.
3) The last two ($FFFE, $FFFF) locations in memory will be used to load the program addressed by the program counter.
4) The interrupt mask bit is set, must be cleared before the MPU can recognize maskable interrupts.

• Enable (E)

This supplies the external clock for the rest of the system when the internal oscillator is used. It is a single phase, TTL compatible clock, and will be the divide by 4 result of the crystal frequency. It will drive one TTL load and 90 pF.

• Non-Maskable Interrupt ($\overline{NMI}$)

A low-going edge on this input requests that a non-maskable-interrupt sequence be generated within the processor. As with interrupt Request signal, the processor will complete the current instruction that is being executed before it recognizes the $\overline{NMI}$ signal. The interrupt mask bit in the Condition Code Register has no effect on $\overline{NMI}$.

In response to an $\overline{NMI}$ interrupt, the Index Register, Program Counter, Accumulators, and Condition Code Register are stored on the stack. At the end of the sequence, a 16-bit address will be loaded that points to a vectoring address located in memory locations $FFFC and $FFFD. An address loaded at these locations causes the MPU to branch to a non-maskable interrupt service routine in memory.

A 3.3 kΩ external resistor to Vcc should be used for wire-OR and optimum control of interrupts.

Inputs $\overline{IRQ_1}$ and $\overline{NMI}$ are hardware interrupt lines that are sampled during E and will start the interrupt routine on the $\overline{E}$ following the completion of an instruction.

• Interrupt Request ($\overline{IRQ_1}$)

This level sensitive input requests that an interrupt sequence be generated within the machine. The processor will wait until it completes the current instruction that it being executed before it recognizes the request. At that time, if the interrupt mask bit in the Condition Code Register is not set, the machine will begin an interrupt sequence. The Index Register, Program Counter, Accumulators, and Condition Code Register are stored on the stack. Next the MPU will respond to the interrupt request by setting the interrupt mask bit "High" so that no further maskable interrupts may occur. At the end of the cycle, a 16-bit address will be loaded that points to a vectoring address which is located in memory locations $FFF8 and $FFF9. An address loaded at these locations causes the MPU to branch to an interrupt routine in memory.

The $\overline{IRQ_1}$ requires a 3.3 kΩ external resister to V_{CC} which should be used for wire-OR and optimum control of interrupts. Internal Interrupts will use an internal interrupt line ($\overline{IRQ_2}$). This interrupt will operate the same as $\overline{IRQ_1}$ except that it will use the vector address of $FFF0 through $FFF7. $\overline{IRQ_1}$ will have priority over $\overline{IRQ_2}$ if both occur at the same time. The Interrupt Mask Bit in the condition code register masks both interrupts (See Table 1).

Table 1 Interrupt Vector Location

	Vector MSB	Vector LSB	Interrupt
Highest Priority	FFFE	FFFF	$\overline{RES}$
	FFFC	FFFD	$\overline{NMI}$
	FFFA	FFFB	Software Interrupt (SWI)
	FFF8	FFF9	$\overline{IRQ_1}$ (or $\overline{IS3}$)
	FFF6	FFF7	ICF (Input Capture)
	FFF4	FFF5	OCF (Output Compare)
	FFF2	FFF3	TOF (Timer Overflow)
Lowest Priority	FFF0	FFF1	SC_1 (RDRF + ORFE + TDRE)

The following pins are available in the Single Chip Mode, and are associated with Port 3 only.

- **Input Strobe ($\overline{IS3}$) (SC_1)**

This sets an interrupt for the processor when the $\overline{IS3}$ Enable bit is set. As shown in Figure 6 Input Strobe Timing, $\overline{IS3}$ will fall t_{IS} minimum after data is valid on Port 3. If $\overline{IS3}$ Enable is set in the I/O Port 3 Control/Status Register, an interrupt will occur. If the latch enable bit in the I/O Port 3 Control/Status Register is set, this strobe will latch the input data from another device when that device has indicated that it has valid data.

- **Output Strobe ($\overline{OS3}$) (SC_2)**

This signal is used by the processor to strobe an external device, indicating valid data is on the I/O pins. The timing for the Output Strobe is shown in Figure 5 I/O Port 3 Control/Status Register is discussed in the following section.

The following pins are available in the Expanded Modes.

- **Read/Write (R/$\overline{W}$) (SC_2)**

This TTL compatible output signals the peripherals and memory devices whether the MPU is in a Read ("High") or a Write ("Low") state. The normal standby state of this signal is Read ("High"). This output is capable of driving one TTL load and 90 pF.

- **I/O Strobe ($\overline{IOS}$) (SC_1)**

In the expanded non-multiplexed mode of operation, $\overline{IOS}$ internally decodes A_9 through A_{15} as zero's and A_8 as a one. This allows external access of the 256 locations from $0100 to $01FF. The timing diagrams are shown as figure 2.

- **Address Strobe (AS) (SC_1)**

In the expanded multiplexed mode of operation address strobe is output on this pin. This signal is used to latch the 8 LSB's of address which are multiplexed with data on Port 3. An 8-bit latch is utilized in conjunction with Address Strobe, as shown in figure 19. Expanded Multiplexed Mode. Address Strobe signals the latch when it is time to latch the address lines so the lines can become data bus lines during the E pulse. The timing for this singal is shown in Figure 1 of Bus Timing. This signal is also used to disable the address from the multiplexed bus allowing a deselect time, t_{ASD} before the data is enabled to the bus.

■ PORTS

There are four I/O ports on the HD6801S MCU; three 8-bit ports and one 5-bit port. There are two control lines associated with one of the 8-bit ports. Each port has an associated write only Data Direction Register which allows each I/O line to be programmed to act as an input or an output*. A "1" in the corresponding Data Direction Register bit will cause that I/O line to be an output. A "0" in the corresponding Data Direction Register bit will cause that I/O line to be an input. There are four ports: Port 1, Port 2, Port 3, and Port 4. Their addresses and the addresses of their Data Direction registers are given in Table 2.

* The only exception is bit 1 of Port 2, which can either be data input or Timer output.

Table 2 Port and Data Direction Register Addresses

Ports	Port Address	Data Direction Register Address
I/O Port 1	$0002	$0000
I/O Port 2	$0003	$0001
I/O Port 3	$0006	$0004
I/O Port 4	$0007	$0005

- **I/O Port 1**

This is an 8-bit port whose individual bits may be defined as inputs or outputs by the corresponding bit in its data direction register. The 8 output buffers have three-state capability, allowing them to enter a high impedance state when the peripheral data lines are used as inputs. In order to be read properly, the voltage on the input lines must be greater than 2.0 V for a logic "1" and less than 0.8 V for a logic "0". As outputs, these lines are TTL compatible and may also be used as a source of up to 1 mA at 1.5 V to directly drive a Darlington base. After Reset, the I/O lines are configured as inputs. In all three modes, Port 1 is always parallel I/O.

- **I/O Port 2**

This port has five lines that may be defined as inputs or outputs by its data direction register. The 5 output buffers have three-state capability, allowing them to enter a high impedance state when used as an input. In order to be read properly, the voltage on the input lines must be greater than 2.0 V for a logic "1" and less than 0.8 V for a logic "0". As outputs, this port has no internal pullup resistors but will drive TTL inputs directly. For driving CMOS inputs, external pullup resistors are required. After Reset, the I/O lines are configured as inputs. Three pins on Port 2 (pins 10, 9, and 8 of the chip) are used to program the mode of operation during reset. The values of these pins at reset are latched into the three MSB's (bits 7, 6, and 5) of Port 2 which are read only. This is explained in the Mode Selection Section.

In all three modes, Port 2 can be configured as I/O and provides access to the Serial Communications Interface and the Timer. Bit 1 is the only pin restricted to data input or Timer output.

- **I/O Port 3**

This is an 8-bit port that can be configured as I/O, a data bus, or an address bus multiplexed with the data bus – depending on the mode of operation hardware programmed by the user at reset. As a data bus, Port 3 is bi-directional. As an input for peripherals, it must be supplied regular TTL levels, that is, greater than 2.0 V for a logic "1" and less than 0.8 V for a logic "0"

Its TTL compatible three-state output buffers are capable of driving one TTL load and 90 pF. In the Expanded Modes, after reset, the data direction register is inhibited and data flow depends on the state of the R/$\overline{W}$ line. The input strobe ($\overline{IS3}$) and the output strobe ($\overline{OS3}$) used for handshaking are explained later.

In the three modes, Port 3 assumes the following characteristics:

Single Chip Mode: Parallel Inputs/Outputs as programmed by its associated Data Direction Register. There are two control lines associated with this port in this mode, an input strobe and an output strobe, that can be used for handshaking. They are controlled by the I/O Port 3 Control/Status Register explained at the end of this section. Three options of Port 3 operations are sumarized as follows: (1) Port 3 input data can be latched using $\overline{IS3}$ (SC_1) as a control signal, (2) $\overline{OS3}$ can be generated by either an MPU read or write to Port 3's Data Register, and (3) and $\overline{IRQ_1}$ interrept can be enabled by an IS3 nagative edge. Port 3 latch and strobe timing is shown in Fig. 5 and Fig. 6.

Expanded Non-Multiplexed Mode: In this mode, Port 3 becomes the data bus (D_0~D_7).

Expanded Multiplexed Mode: In this mode, Port 3 becomes both the data bus (D_0~D_7) and lower bits of the address bus (A_0~A_7). An address strobe output is true when the address is on the port.

I/O PORT 3 CONTROL/STATUS REGISTER

	7	6	5	4	3	2	1	0
$000F	IS3 FLAG	IS3 IRQ1 ENABLE	X	OSS	LATCH ENABLE	X	X	X

Bit 0; Not used.
Bit 1; Not used.
Bit 2; Not used.
Bit 3; LATCH ENABLE. This controls the input latch for I/O Port 3. If this bit is set "High" the input data will be latched with the falling edge of the Input Strobe, $\overline{IS3}$. This bit is cleared by reset, and the latch is "re-opened" with MCU read Port 3.
Bit 4; OSS. (Output Strobe Select) This bit will select if the Output Strobe should be generated at $\overline{OS3}$ (SC_2) by a write to I/O Port 3 or a read of I/O Port 3. When this bit is cleared the strobe is generated by a read Port 3. When this bit is set the strobe is generated by a write Port 3.
Bit 5; Not used.
Bit 6; $\overline{IS3}$ $\overline{IRQ_1}$ ENABLE. When set, interrupt will be enabled whenever $\overline{IS3}$ FLAG is set; when clear, interrupt is inhibited. This bit is cleared by $\overline{RES}$.
Bit 7; $\overline{IS3}$ FLAG. This is a read only status bit that is set by the falling edge of the input strobe, $\overline{IS3}$ (SC_1). It is cleared by a read of the Control/Status Register followed by a read or write of I/O Port 3. Reset will clear this bit.

- **I/O Port 4**

This is an 8-bit port that can be configured as I/O or as address lines depending on the mode of operation. In order to be read properly, the voltage on the input lines must be greater than 2.0 V for a logic "1" and less than 0.8 V for a logic "0"

As outputs, each line is TTL compatible and can drive 1 TTL load and 90 pF. After reset, the lines are configured as inputs. To use the pins as addresses, therefore, they should be programmed as outputs. In the three modes, Port 4 assumes the following characteristics:

Single Chip Mode: Parallel Inputs/Outputs as programmed by its associated Data Direction Register.

Expanded Non-Multiplexed Mode: In this mode, Port 4 is configured as the lower order address lines (A_0~A_7) by writing one's to the data direction register. When all eight address lines are not needed, the remaining lines, starting with the most significant bit, may be used as I/O (inputs only).

Expanded Multiplexed Mode: In this mode, Port 4 is configured as the higher order address lines (A_8~A_{15}) by writing one's to the data direction register. When all eight address lines are not needed, the remaining lines, starting with the most significant bit, may be used as I/O (inputs only).

■ OPERATION MODES

The mode of operation that HD6801S will operate in after Reset is determined by hardware that the user must wire on pins 10, 9, and 8 of the chip. These pins are the three LSB's (I/O 2, I/O 1, and I/O 0 respectively) of Port 2. They are latched into programmed control bits PC2, PC1, and PC0 when reset goes high. I/O Port 2 Register is shown below.

PORT 2 DATA REGISTER

	7	6	5	4	3	2	1	0
$0003	PC2	PC1	PC0	I/O 4	I/O 3	I/O 2	I/O 1	I/O 0

An example of external hardware that could be used for Mode Selection is shown in Fig. 14. The HD14053B provides the isolation between the peripheral device and MCU during Reset, which is necessary if data conflict can occur between peripheral device and Mode generation circuit.

As bits 5, 6 and 7 of Port 2 are read only, the mode cannot be changed through software. The mode selections are shown in Table 3.

The HD6801S is capable of operating in three basic modes; (1) Single Chip Mode, (2) Expanded Multiplexed Mode (compatible with HMCS6800 peripheral family) (3) Expanded Non-Multiplexed Mode.

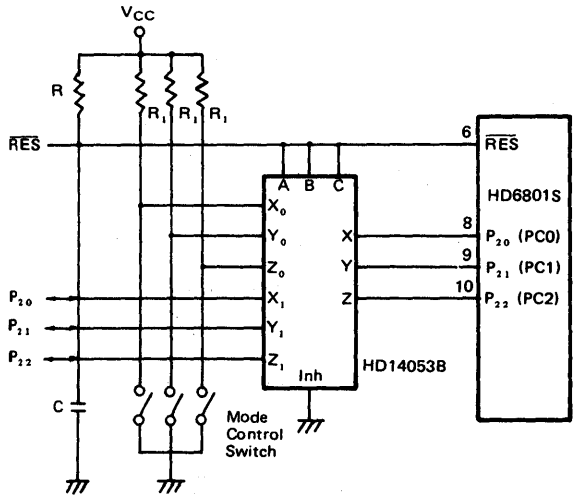
- **Single Chip Mode**

In the Single Chip Mode the Ports are configured for I/O.

This is shown in Figure 16 the single Chip Mode. In this mode, Port 3 will have two associated control lines, an input strobe and an output strobe for handshaking data.

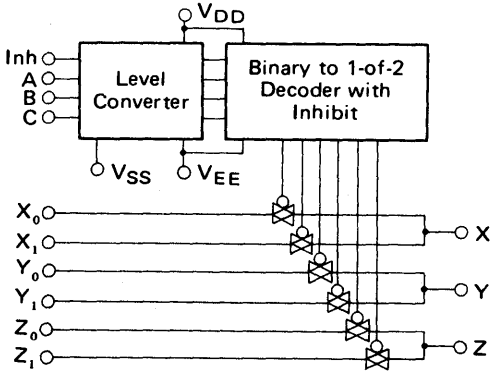
- **Expanded Non-Multiplexed Mode**

In this mode the HD6801S will directly address HMCS6800 peripherals with no external logic. In this mode Port 3 becomes the data bus. Port 4 becomes the A_0~A_7 address bus or partial address and I/O (inputs only). Port 2 can be parallel I/O, serial I/O, Timer, or any combination of them. Port 1 is parallel I/O only. In this mode the HD6801S is expandable to 256 locations. The eight address lines associated with Port 4 may be substituted for I/O (inputs only) if a fewer number of address lines will satisfy the application (See Figure 17).



[NOTES] 1) Mode 7 as shown
2) RC≈Reset time constant
3) R_1 =10kΩ

Figure 14 Recommended Circuit for Mode Selection



Truth Table

Control Input				On Switch
Inhibit	Select			
	C	B	A	HD14053B
0	0	0	0	Z_0 Y_0 X_0
0	0	0	1	Z_0 Y_0 X_1
0	0	1	0	Z_0 Y_1 X_0
0	0	1	1	Z_0 Y_1 X_1
0	1	0	0	Z_1 Y_0 X_0
0	1	0	1	Z_1 Y_0 X_1
0	1	1	0	Z_1 Y_1 X_0
0	1	1	1	Z_1 Y_1 X_1
1	X	X	X	–

Figure 15 HD14053B Multiplexers/Demultiplexers

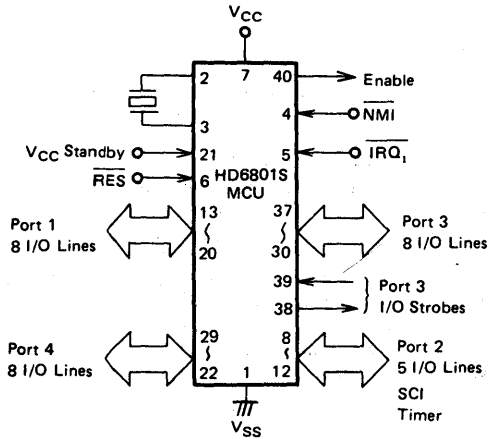


Figure 16 HD6801S MCU Single-Chip Mode

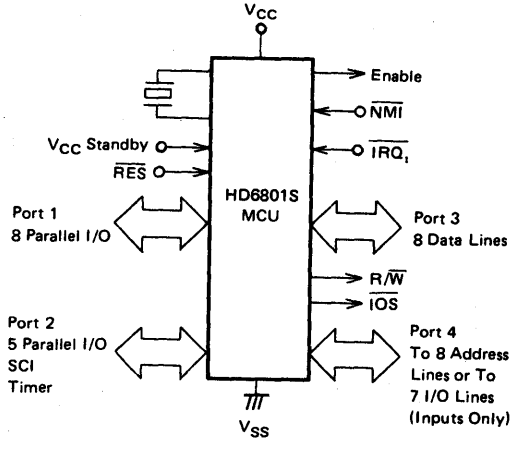


Figure 17 HD6801S MCU Expanded Non-Multiplexed Mode

- **Expanded Multiplexed Mode**

In this mode Port 4 becomes higher order address lines with an alternative of substituting some of the address lines for I/O (inputs only). Port 3 is the data bus multiplexed with the lower order address lines differentiated by an output called Address Strobe. Port 2 is 5 lines of Parallel I/O, SCI, Timer, or any combination of them. Port 1 is 8 Parallel I/O lines. In this mode it is expandable to 65k words. (See Figure 18).

- **Lower order Address Bus Latches**

Since the data bus is multiplexed with the lower order address bus in Port 3, latches are required to latch those address bits. The 74LS373 Transparent octal D-type latch can be used with the HD6801S to latch the least significant address byte. Figure 19 shows how to connect the latch to the HD6801S. The output control to the 74LS373 may be connected to ground.

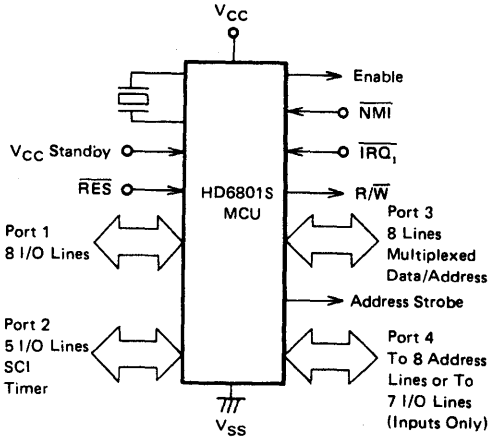
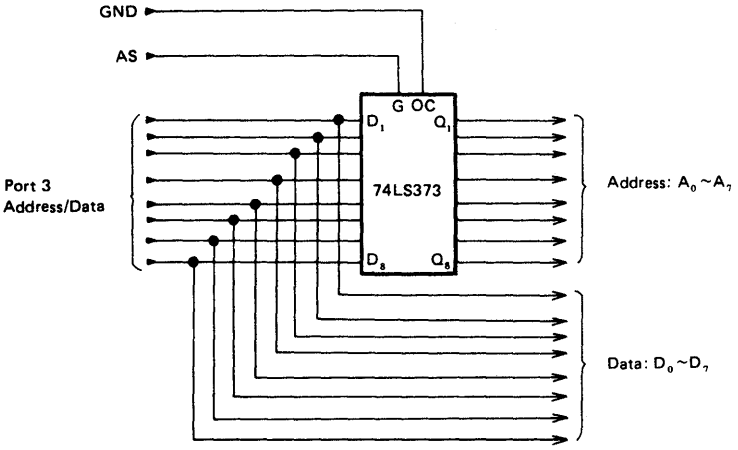


Figure 18 HD6801S MCU Expanded Multiplexed Mode



Function Table

Output Control	Enable G	Enable D	Output Q
L	H	H	H
L	H	L	L
L	L	X	Q_0
H	X	X	Z

Figure 19 Latch Connection

- **Mode and Port Summary MCU Signal Description**

This section gives a description of the MCU signals for the various modes. SC_1 and SC_2 are signals which vary with the mode that the chip is in.

MODE	PORT 1 Eight Lines	PORT 2 Five Lines	PORT 3 Eight Lines	PORT 4 Eight Lines	SC_1	SC_2
SINGLE CHIP	I/O	I/O	I/O	I/O	$\overline{IS3}$ (I)	$\overline{OS3}$ (O)
EXPANDED MUX	I/O	I/O	ADDRESS BUS (A_0~A_7) DATA BUS (D_0~D_7)	ADDRESS BUS* (A_8~A_{15})	AS(O)	R/$\overline{W}$(O)
EXPANDED NON-MUX	I/O	I/O	DATA BUS (D_0~D_7)	ADDRESS BUS* (A_0~A_7)	$\overline{IOS}$(O)	R/$\overline{W}$(O)

*These lines can be substituted for I/O (Input Only) starting with the most significant address line.
I = Input
O = Output
R/$\overline{W}$ = Read/$\overline{Write}$
$\overline{IS3}$ = Input Strobe
$\overline{OS3}$ = Output Strobe
$\overline{IOS}$ = I/O Select
SC = Strobe Control
AS = Address Strobe

Table 3 Mode Selection Summary

Mode	P_{22} ($PC2$)	P_{21} ($PC1$)	P_{20} ($PC0$)	ROM	RAM	Interrupt Vectors	Bus Mode	Operating Mode
7	H	H	H	I	I	I	I	Single Chip
6	H	H	L	I	I	I	MUX(6)	Multiplexed/Partial Decode
5	H	L	H	I	I	I	NMUX(6)	Non-Multiplexed/Partial Decode
4	H	L	L	I(2)	I(1)	I	I	Single Chip Test
3	L	H	H	E	E	E	MUX	Multiplexed/No RAM & ROM
2	L	H	L	E	I	E	MUX	Multiplexed/RAM
1	L	L	H	I	I	E	MUX	Multiplexed/RAM & ROM
0	L	L	L	I	I	I(3)	MUX	Multiplexed Test

LEGEND:
I – Internal
E – External
MUX – Multiplexed
NMUX – Non-Multiplexed
L – Logic "0"
H – Logic "1"

[NOTES]
1) Internal RAM is addressed at $XX80
2) Internal ROM is disabled
3) $\overline{RES}$ vector is external for 2 cycles after $\overline{RES}$ goes "High"
4) Addresses associated with Ports 3 and 4 are considered external in Modes 0, 1, 2, and 3
5) Addresses associated with Port 3 are considered external in Modes 5 and 6
6) Port 4 default is user data input; address output is optional by writing to Port 4 Data Direction Register

■ MEMORY MAPS

The MCU can provide up to 65k byte address space depending on the operating mode. A memory map for each operating mode is shown in Figure 20. The first 32 locations of each map are reserved for the MCU's internal register area, as shown in Table 4. With exceptions as indicated.

Table 4 Internal Register Area

Register	Address
Port 1 Data Direction Register***	00
Port 2 Data Direction Register***	01
Port 1 Data Register	02
Port 2 Data Register	03
Port 3 Data Direction Register***	04*
Port 4 Data Direction Register***	05**
Port 3 Data Register	06*
Port 4 Data Register	07**
Timer Control and Status Register	08
Counter (High Byte)	09
Counter (Low Byte)	0A
Output Compare Register (High Byte)	0B
Output Compare Register (Low Byte)	0C
Input Capture Register (High Byte)	0D
Input Capture Register (Low Byte)	0E
Port 3 Control and Status Register	0F*
Rate and Mode Control Register	10
Transmit/Receive Control and Status Register	11
Receive Data Register	12
Transmit Data Register	13
RAM Control Register	14
Reserved	15-1F

* External address in Modes 0, 1, 2, 3, 5, 6; cannot be accessed in Mode 5 (No. $\overline{IOS}$)
** External addresses in Modes 0, 1, 2, 3
*** 1=Output, 0=Input

■ INTERRUPT FLOWCHART

The Interrupt flow chart is depicted in Figure 24 and is common to every interrupt excluding reset.

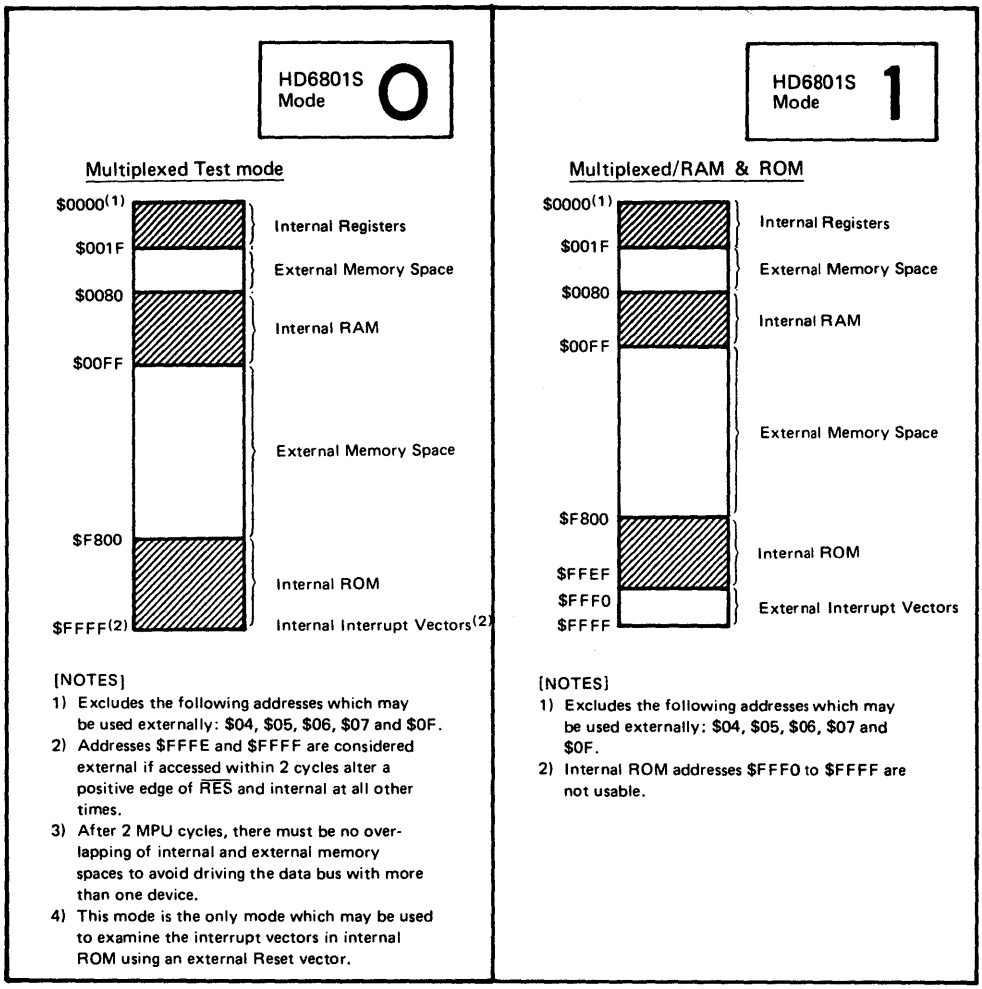


Figure 20 HD6801S Memory Maps

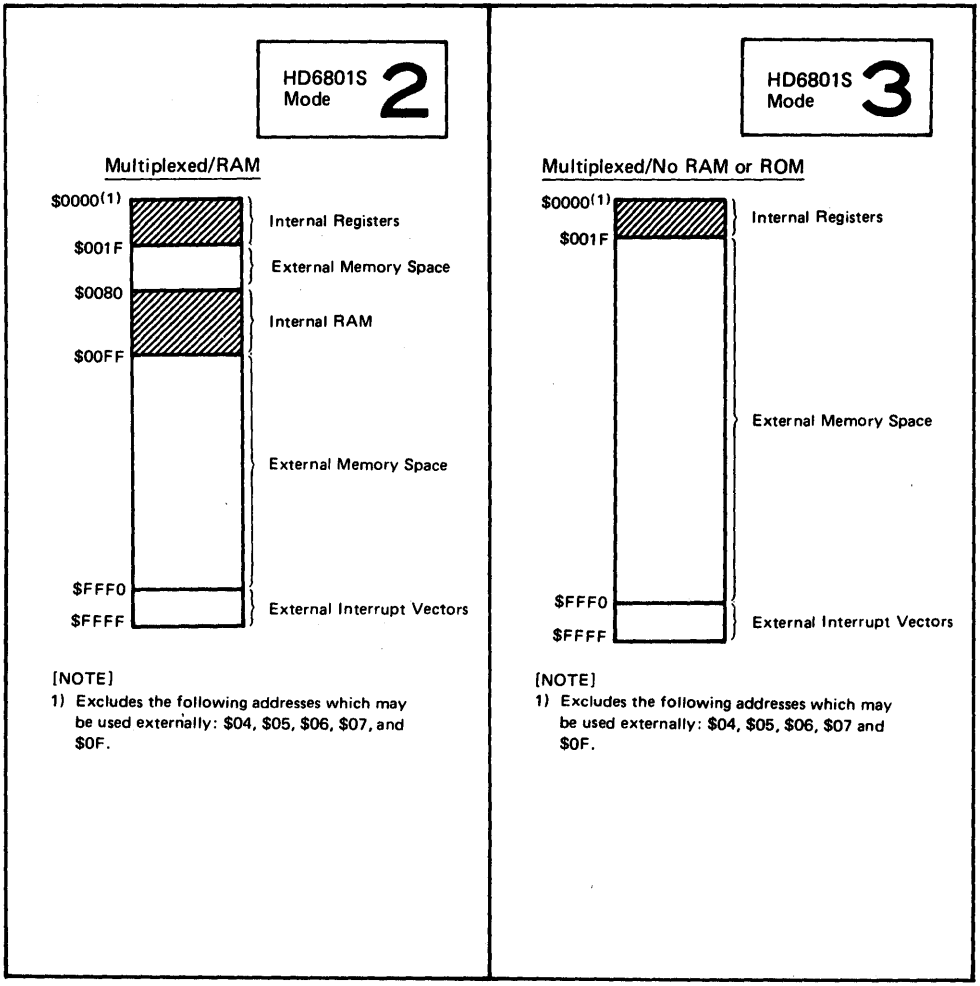


Figure 20 HD6801S Memory Maps (Continued)

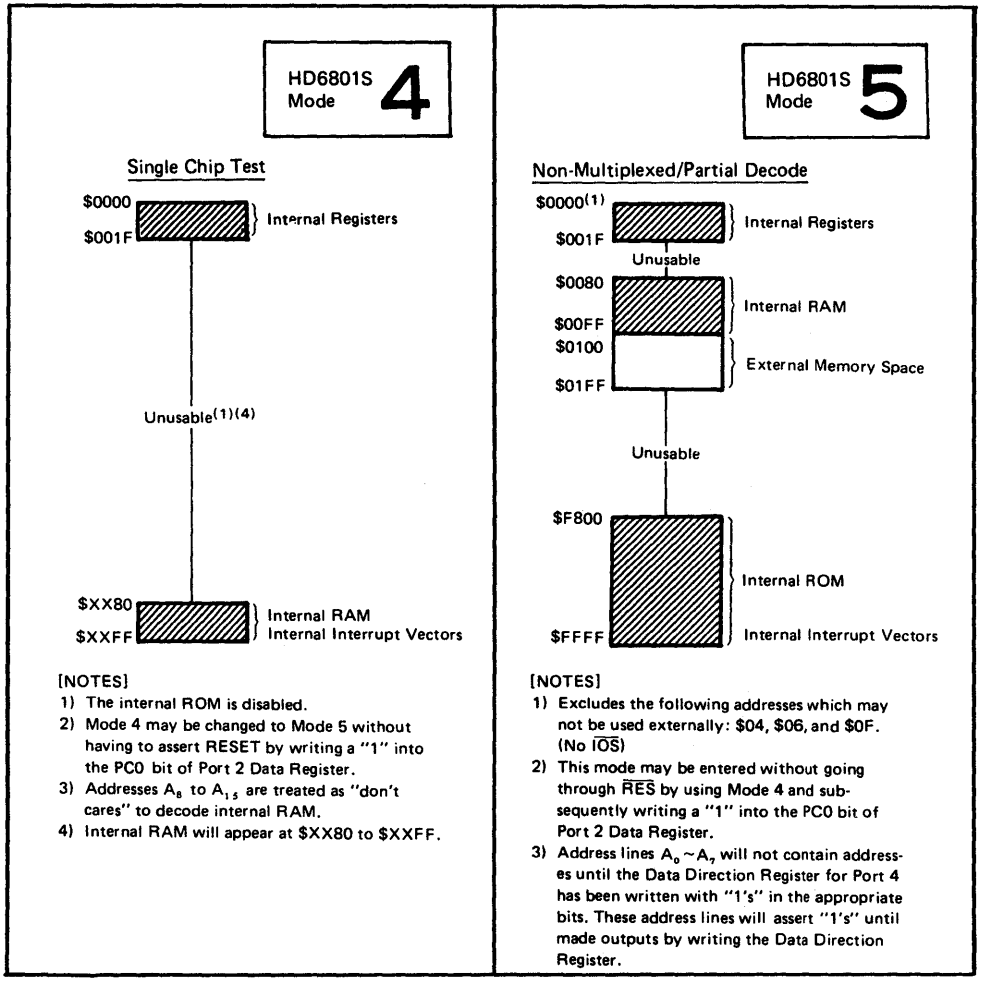
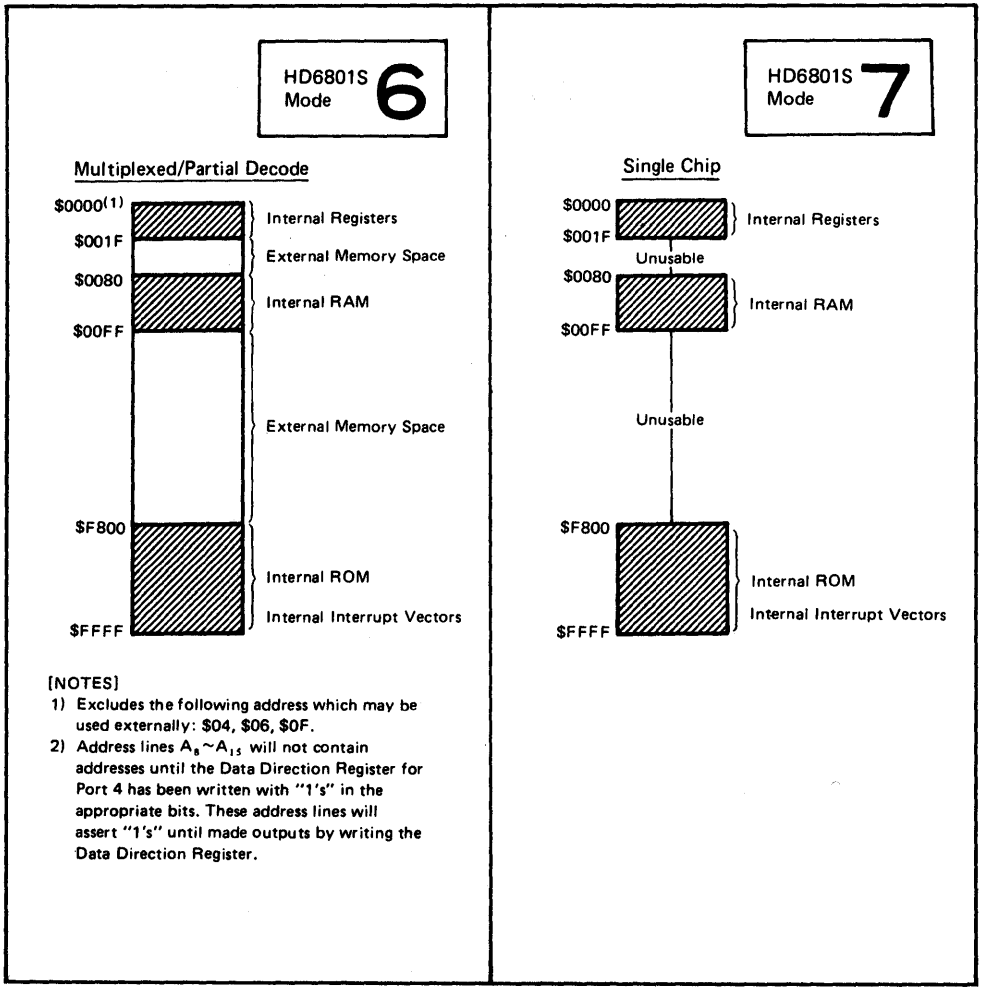


Figure 20 HD6801S Memory Maps (Continued)



[NOTES]
1) Excludes the following address which may be used externally: $04, $06, $0F.
2) Address lines $A_8 \sim A_{15}$ will not contain addresses until the Data Direction Register for Port 4 has been written with "1's" in the appropriate bits. These address lines will assert "1's" until made outputs by writing the Data Direction Register.

Figure 20 HD6801S Memory Maps (Continued)

■ PROGRAMMABLE TIMER

The HD6801S contains an on-chip 16-bit programmable timer which may be used to perform measurements on an input waveform while independently generating an output waveform. Pulse widths for both input and output signals may vary from a few microseconds to many seconds. The timer hardware consists of

- an 8-bit control and status register,
- a 16-bit free running counter,
- a 16-bit output compare register, and
- a 16-bit input capture register

A block diagram of the timer registers is shown in Figure 21.

- **Free Running Counter ($0009:000A)**

The key element in the programmable timer is a 16-bit free running counter which is driven to increasing values by E (Enable). The counter value may be read by the MPU software at any time. The counter is cleared to zero on $\overline{RES}$ and may be considered a read-only register with one exception. Any MPU write to the counter's address ($09) will always result in preset value of $FFF8 being loaded into the counter regardless of the value involved in the write. This preset figure is intended for testing operation of the part, but may be of value in some applications.

- **Output Compare Register ($000B:000C)**

The Output Compare Register is a 16-bit read/write register which is used to control an output waveform. The contents of this register are constantly compared with the current value of the free running counter. When a match is found, a flag is set (OCF) in the Timer Control and Status Register (TCSR) and the current value of the Output Level bit (OLVL) in the TCSR is clocked to the Output Level Register. Providing the Data Direction Register for Port 2, Bit 1 contains a "1" (Output), the output level register value will appear on the pin for Port 2 Bit 1. The values in the Output Compare Register and Output level bit may then be changed to control the output level on the next compare value. The Output Compare Register is set to $FFFF during $\overline{RES}$. The Compare function is inhibited for one cycle following a write to the high byte of the Output Compare Register to insure a valid 16-bit value is in the register before a compare is made.

- **Input Capture Register ($000D:000E)**

The Input Capture Register is a 16-bit read-only register used to store the current value of the free running counter when the proper transition of an external input signal occurs. The input transition change required to trigger the counter transfer is controlled by the input Edge bit (IEDG) in the TCSR. The Data Direction Register bit for Port 2 Bit 0, should* be clear (zero) in order to gate in the external input signal to the edge detect unit in the timer.

* With Port 2 Bit 0 configured as an output and set to "1", the external input will still be seen by the edge detect unit.

- **Timer Control and Status Register (TCSR) ($0008)**

The Timer Control and Status Register consists of an 8-bit register of which all 8 bits are readable but only the low order 5 bits may be written. The upper three bits contain read-only timer status information and indicate that:

- a proper transition has taken place on the input pin with a subsequent transfer of the current counter value to the input capture register.
- a match has been found between the value in the free running counter and the output compare register, and
- when $0000 is in the free running counter.

Each of the flags may be enabled onto the HD6801 internal bus ($\overline{IRQ_2}$) with an individual Enable bit in the TCSR. If the

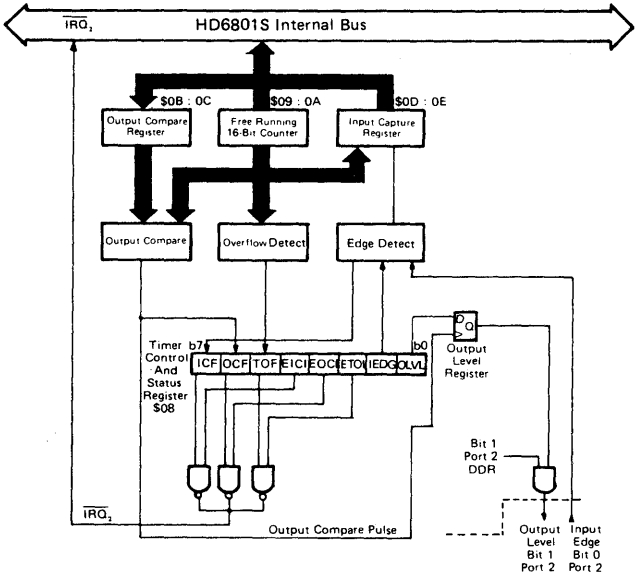


Figure 21 Block Diagram of Programable Timer

Timer Control and Status Register

7	6	5	4	3	2	1	0	
ICF	OCF	TOF	EICI	EOCI	ETOI	IEDG	OLVL	$0008

I-bit in the HD6801S Condition Code register has been cleared, a priority vectored interrupt will occur corresponding to the flag bit(s) set. A description for each bit follows:

Bit 0 **OLVL** Output Level – This value is clocked to the output level register on a successful output compare. If the DDR for Port 2 bit 1 is set, the value will appear on the output pin.

Bit 1 **IEDG** Input Edge – This bit controls which transition of an input will trigger a transfer of the counter to the input capture register. The DDR for Port 2 Bit 0 must be clear for this function to operate. IEDG = 0 Transfer takes place on a negative edge ("High"-to-"Low" transition).
IEDG = 1 Transfer takes place on a positive edge ("Low"-to-"High" transition).

Bit 2 **ETOI** Enable Timer Overflow Interrupt – When set, this bit enables $\overline{IRQ_2}$ to occur on the internal bus for a TOF interrupt; when clear the interrupt is inhibited.

Bit 3 **EOCI** Enable Output Compare Interrupt – When set, this bit enables $\overline{IRQ_2}$ to appear on the internal bus for an output compare interrupt; when clear the interrupt is inhibited.

Bit 4 **EICI** Enable input Capture Interrupt – When set, this bit enables $\overline{IRQ_2}$ to occur on the internal bus for an input capture interrupt; when clear the interrupt is inhibited.

Bit 5 **TOF** Timer Overflow Flag – This read-only bit is set when the counter contains $0000. It is cleared by a read of the TCSR (with TOF set) followed by an MPU read of the Counter ($09).

Bit 6 **OCF** Output Compare Flag – This read-only bit is set when a match is found between the output compare register and the free running counter. It is cleared by a read of the TCSR (with OCF set) followed by an MPU write to the output compare register ($0B or $0C).

Bit 7 **ICF** Input Capture Flag – This read-only status bit is set by a proper transition on the input; it is cleared by a read of the TCSR (with ICF set) followed by an MPU read of the Input Capture Register ($0D).

■ SERIAL COMMUNICATIONS INTERFACE

The HD6801S contains a full-duplex asynchronous serial communications interface (SCI) on chip. The controller comprises a transmitter and a receiver which operate independently or each other but in the same data format and at the same data rate. Both transmitter and receiver communicate with the MPU via the data bus and with the outside world via pins 2, 3, and 4 of Port 2. The hardware, software, and registers are explained in the following paragraphs.

• Wake-Up Feature

In a typical multi-processor application, the software protocol will usually contain a destination address in the initial byte(s) of the message. In order to permit non-selected MPU's to ignore the remainder of the message, a wake-up feature is included whereby all further interrupt processing may be optionally inhibited until the beginning of the next message. When the next message appears, the hardware re-enables (or "wakes-up") the for the next message. The "wake-up" is automatically triggered by a string of ten consecutive 1's which indicates an idle transmit line. The software protocol must provide for the short idle period between any two consecutive messages.

• Programmable Options

The following features of the HD6801S serial I/O section are programmable:

- format – standard mark/space (NRZ)
- Clock – external or internal
- baud rate – one of 4 per given MPU ϕ_2 clock frequency or external clock ×8 input
- wake-up feature – enabled or disabled
- Interrupt requests – enabled or masked individually for transmitter and receiver data registers
- clock output – internal clock enabled or disabled to Port 2 (Bit 2)
- Port 2 (bits 3 and 4) – dedicated or not dedicated to serial I/O individually for transmitter and receiver.

• Serial Communications Hardware

The serial communications hardware is controlled by 4 registers as shown in Figure 22. The registers include:

- an 8-bit control and status register
- a 4-bit rate and mode control register (write only)
- an 8-bit read only receive data register and
- an 8-bit write only transmit data register.

In addition to the four registers, the serial I/O section utilizes bit 3 (serial input) and bit 4 (serial output) of Port 2. Bit 2 of Port 2 is utilized if the internal-clock-out or external-clock-in options are selected.

Transmit/Receive Control and Status (TRCS) Register

The TRCS register consists of an 8-bit register of which all 8 bits may be read while only bits 0~4 may be written. The register is initialized to $20 on $\overline{RES}$. The bits in the TRCS register are defined as follows:

Transmit/Receive Control and Status Register

7	6	5	4	3	2	1	0	
RDRE	ORFE	TDRE	RIE	RE	TIE	TE	WU	ADDR : $0011

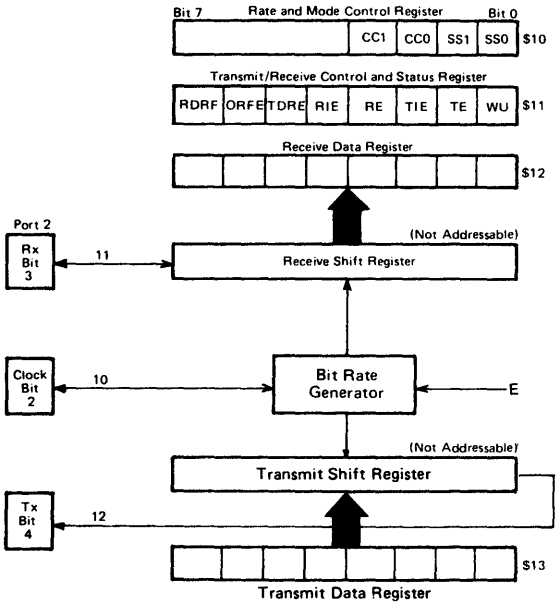


Figure 22 Serial I/O Registers

Bit 0 **WU** "Wake-up" on Next Message – set by HD6801S software and cleared by hardware on receipt of ten consecutive 1's or reset of RE flag. It should be noted that RE flag should be set in advance of MPU set of WU flag.

Bit 1 **TE** Transmit Enable – set by HD6801S to produce preamble of nine consecutive 1's and to enable gating of transmitter output to Port 2, bit 4 regardless of the DDR value corresponding to this bit; when clear, serial I/O has no effect on Port 2 bit 4.

TE set should be after at least one bit time of data transmit rate from the set-up of transmit data rate and mode.

Bit 2 **TIE** Transmit Interrupt Enable – when set, will permit an $\overline{IRQ_2}$ interrupt to occur when bit 5 (TDRE) is set; when clear, the TDRE value is masked from the bus.

Bit 3 **RE** Receiver Enable – when set, gates Port 2 bit 3 to input of receiver regardless of DDR value for this bit; when clear, serial I/O has no effect on Port 2 bit 3.

Bit 4 **RIE** Receiver Interrupt Enable – when set, will permit an $\overline{IRQ_2}$ interrupt to occur when bit 7 (RDRF) or bit 6 (ORFE) is set; when clear, the interrupt is masked.

Bit 5 **TDRE** Transmit Data Register Empty – set by hardware when a transfer is made from the transmit data register to the output shift register. The TDRE bit is cleared by reading the status register, then writing a new byte into the transmit data register, TDRE is initialized to 1 by $\overline{RES}$.

Bit 6 **ORFE** Over-Run-Framing Error – set by hardware when an overrun or framing error occurs (receive only). An overrun is defined as a new byte received with last byte still in Data Register/Buffer. A framing error has occurred when the byte boundaries in bit stream are not synchronized to bit counter. The ORFE bit is cleared by reading the status register, then reading the Receive Data Register, or by $\overline{RES}$.

Bit 7 **RDRF** Receiver Data Register Full – Set by hardware when a transfer from the input shift register to the receiver data register is made. The RDRF bit is cleared by reading the status register, then reading the Receive Data Register, or by $\overline{RES}$.

Rate and Mode Control Register

The Rate and Mode Control register controls the following serial I/O variables:

- Baud rate
- format
- clocking source, and
- Port 2 bit 2 configuration

The register consists of 4 bits all of which are write-only and cleared on $\overline{RES}$. The 4 bits in the register may be considered as a pair of 2-bit fields. The two low order bits control the bit rate for internal clocking and the remaining two bits control the format and clock select logic. The register definition is as follows:

Rate and Mode Control Register

7	6	5	4	3	2	1	0	
X	X	X	X	CC1	CC0	SS1	SS0	ADDR : $0010

Bit 0 **SS0** Speed Select – These bits select the Baud rate for
Bit 1 **SS1** the internal clock. The four rates which may be selected are a function of the MPU ϕ_2 clock frequency. Table 5 lists the available Baud rates.

Bit 2 **CC0** Clock Control and Format Select – this 2-bit field
Bit 3 **CC1** controls the format and clock select logic. Table 6 defines the bit field.

Table 5 SCI Bit Times and Rates

	XTAL	2.4576 MHz	4.0 MHz	4.9152 MHz*
SS1 : SS0	E	614.4 kHz	1.0 MHz	1.2288 MHz
0 0	E ÷ 16	26 μs/38,400 Baud	16 μs/62,500 Baud	13 μs/76,800 Baud
0 1	E ÷ 128	208 μs/4,800 Baud	128 μs/7812.5 Baud	104.2 μs/9,600 Baud
1 0	E ÷ 1024	1.67 ms/600 Baud	1.024 ms/976.6 Baud	33.3 μs/1,200 Baud
1 1	E ÷ 4096	6.67 ms/150 Baud	4.096 ms/244.1 Baud	3.33 ms/300 Baud

* HD6801S5 Only

Table 6 SCI Format and Clock Source Control

CC1, CC0	Format	Clock Source	Port 2 Bit 2	Port 2 Bit 3	Port 2 Bit 4
00	–	–	–	**	**
01	NRZ	Internal	Not Used	**	**
10	NRZ	Internal	Output*	**	**
11	NRZ	External	Input	**	**

* Clock output is available regardless of values for bits RE and TE.
** Bit 3 is used for serial input if RE = "1" in TRCS; bit 4 is used for serial output if TE = "1" in TRCS.

Internally Generated Clock

If the user wishes for the serial I/O to furnish a clock, the following requirements are applicable:

- the values of RE and TE are immaterial.
- CC1, CC0 must be set to 10
- the maximum clock rate will be E ÷ 16.
- the clock will be at 1× the bit rate and will have a rising edge at mid-bit.

Externally Generated Clock

If the user wishes to provide an external clock for the serial I/O, the following requirements are applicable:

- the CC1, CC0, field in the Rate and Mode Control Register must be set to 11,
- the external clock must be set to 8 times (×8) the desired baud rate and
- the maximum external clock frequency is 1.0 MHz.

- **Serial Operations**

The serial I/O hardware should be initialized by the HD6801S software prior to operation. This sequence will normally consist of;

- writing the desired operation control bits to the Rate and Mode Control Register and
- writing the desired operational control bits in the Transmit/Receive Control and Status Register.

The Transmitter Enable (TE) and Receiver Enable (RE) bits may be left set for dedicated operations.

Transmit Operations

The transmit operation is enabled by the TE bit in the Transmit/Receive Control and Status Register. This bit when set, gates the output of the serial transmit shift register to Port 2 Bit 4 and takes unconditional control over the Data Direction Register value for Port 2, Bit 4.

Following a $\overline{RES}$ the user should configure both the Rate and Mode Control Register and the Transmit/Receive Control and Status Register for desired operation. Setting the TE bit during this procedure initiates the serial output by first transmitting a ten-bit preamble of 1's. Following the preamble, internal synchronization is established and the transmitter section is ready for operation.

At this point one of two situation exist:

1) if the Transmit Data Register is empty (TDRE = 1), a continuous string of ones will be sent indicating an idle line, or,
2) if data has been loaded into the Transmit Data Register (TDRE = 0), the word is transferred to the output shift register and transmission of the data word will begin.

During the transfer itself, the 0 start bit is first transmitted. Then the 8 data bits (beginning with bit 0) followed by the stop bit, are transmitted. When the Transmitter Data Register has been emptied, the hardware sets the TDRE flag bit.

If the HD6801S fails to respond to the flag within the proper time, (TDRE is still set when the next normal transfer from the parallel data register to the serial output register should occur) then a 1 will be sent (instead of a 0) at "Start" bit time, followed by more 1's until more data is supplied to the data register. No 0's will be sent while TDRE remains a 1.

Receive Operation

The receive operation is enabled by the RE bit which gates in the serial input through Port 2 Bit 3. The receiver section operation is conditioned by the contents of the Transmit/ Receive Control and Status Register and the Rate and Mode Control Register.

The receiver bit interval is divided into 8 sub-intervals for internal synchronization. In the NRZ Mode, the received bit stream is synchronized by the first 0 (space) encountered.

The approximate center of each bit time is strobed during the next 10 bits. If the tenth bit is not a 1 (stop bit) a framing error is assumed, and bit ORFE is set. If the tenth bit as a 1, the data is transferred to the Receive Data Register, and interrupt flag RDRF is set. If RDRF is still set at the next tenth bit time, ORFE will be set, indicating an over-run has occurred. When the HD6801S responds to either flag (RDRF or ORFE) by reading the status register followed by reading the Data Register, RDRF (or ORFE) will be cleared.

■ RAM CONTROL REGISTER

This register, which is addressed at $0014, gives status information about the standby RAM. A 0 in the RAM enable bit (RAM E) will disable the standby RAM, thereby protecting it at power down if V_{CC} Standby is held greater than V_{SBB} volts, as explained previously in the signal description for V_{CC} Standby.

RAM Control Register

$0014	STBY PWR	RAME	X	X	X	X	X	X

Bit 0 Not used.
Bit 1 Not used.
Bit 2 Not used.
Bit 3 Not used.
Bit 4 Not used.
Bit 5 Not used.
Bit 6 **RAME** The RAM Enable control bit allows the user the ability to disable the standby RAM. This bit is set to a logic "1" by $\overline{RES}$ which enables the standby RAM and can be written to one or zero uner program control. When the RAM is disabled, data is read from external memory.
Big 7 **STBY PWR** The Standby Power bit is cleared when the standby voltage is removed. This bit is a read/write status flag that the user can read which indicates that the standby RAM voltage has been applied, and the data in the standby RAM is valid.

■ GENERAL DESCRIPTION OF INSTRUCTION SET

The HD6801S is upward object code compatible with the HD6800 as it implements the full HMCS6800 instruction set. The execution times of key instructions have been reduced to increase throughout. In addition, new instructions have been added; these include 16-bit operations and a hardware multiply.

Included in the instruction set section are the following:

- MPU Programming Model (Figure 23)
- Addressing modes
- Accumulator and memory instructions – Table 7
- New instructions
- Index register and stack manipulations instructions – Table 8
- Jump and branch instructions – Table 9
- Condition code register manipulation instructions – Table 10
- Instructions Execution times in machine cycles – Table 11
- Summary of cycle by cycle operation – Table 12
- Op codes Map – Table 13

● MPU Programming Model

The programming model for the HD6801S is shown in Figure 23. The double (D) accumulator is physically the same as the Accumulator A concatenated with the Accumulator B so that any operation using accumulator D will destroy information in A and B.

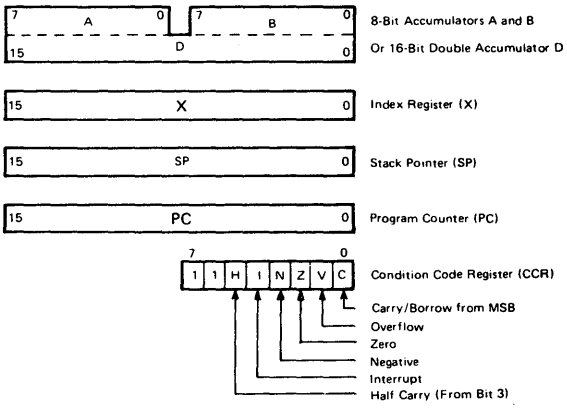


Figure 23 MCU Programming Model

● MPU Addressing Modes

The HD6801S eight-bit microcomputer unit has seven address modes that can be used by a programmer, with the addressing mode a function of both the type of instruction and the coding within the instruction. A summary of the addressing modes for a particular instruction can be found in Table 11 along with the associated instruction execution time that is given in machine cycles. With a clock frequency of 4 MHz, these times would be microseconds.

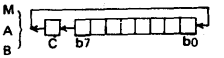
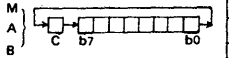
Accumulator (ACCX) Addressing

In accumulator only addressing, either accumulator A or accumulator B is specified. These are one-byte instructions.

Immediate Addressing

In immediate addressing, the operand is contained in the second byte of the instruction except LDS and LDX which have the operand in the second and third bytes of the instruction. The MCU addresses this location when it fetches the immediate instruction for execution. These are two or three-byte instructions.

Table 7 Accumulator & Memory Instructions

Operations	Mnemonic	Addressing Modes															Boolean/ Arithmetic Operation	Condition Code Register					
		IMMED.			DIRECT			INDEX			EXTEND			IMPLIED				5	4	3	2	1	0
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#		H	I	N	Z	V	C
Add	ADDA	8B	2	2	9B	3	2	AB	4	2	BB	4	3				A + B → A	↕	•	↕	↕	↕	↕
	ADDB	CB	2	2	DB	3	2	EB	4	2	FB	4	3				B + M → B	↕	•	↕	↕	↕	↕
Add Double	ADDD	C3	4	3	D3	5	2	E3	6	2	F3	6	3				A : B + M : M + 1 → A : B	•	•	↕	↕	↕	↕
Add Accumulators	ABA													1B	2	1	A + B → A	↕	•	↕	↕	↕	↕
Add With Carry	ADCA	89	2	2	99	3	2	A9	4	2	B9	4	3				A + M + C → A	↕	•	↕	↕	↕	↕
	ADCB	C9	2	2	D9	3	2	E9	4	2	F9	4	3				B + M + C → B	↕	•	↕	↕	↕	↕
AND	ANDA	84	2	2	94	3	2	A4	4	2	B4	4	3				A·M → A	•	•	↕	↕	R	•
	ANDB	C4	2	2	D4	3	2	E4	4	2	F4	4	3				B·M → B	•	•	↕	↕	R	•
Bit Test	BIT A	85	2	2	95	3	2	A5	4	2	B5	4	3				A·M	•	•	↕	↕	R	•
	BIT B	C5	2	2	D5	3	2	E5	4	2	F5	4	3				B·M	•	•	↕	↕	R	•
Clear	CLR							6F	6	2	7F	6	3				00 → M	•	•	R	S	R	R
	CLRA													4F	2	1	00 → A	•	•	R	S	R	R
	CLRB													5F	2	1	00 → B	•	•	R	S	R	R
Compare	CMPA	81	2	2	91	3	2	A1	4	2	B1	4	3				A − M	•	•	↕	↕	↕	↕
	CMPB	C1	2	2	D1	3	2	E1	4	2	F1	4	3				B − M	•	•	↕	↕	↕	↕
Compare Accumulators	CBA													11	2	1	A − B	•	•	↕	↕	↕	↕
Complement, 1's	COM							63	6	2	73	6	3				$\overline{M} \rightarrow M$	•	•	↕	↕	R	S
	COMA													43	2	1	$\overline{A} \rightarrow A$	•	•	↕	↕	R	S
	COMB													53	2	1	$\overline{B} \rightarrow B$	•	•	↕	↕	R	S
Complement, 2's (Negate)	NEG							60	6	2	70	6	3				00 − M → M	•	•	↕	↕	①	②
	NEGA													40	2	1	00 − A → A	•	•	↕	↕	①	②
	NEGB													50	2	1	00 − B → B	•	•	↕	↕	①	②
Decimal Adjust, A	DAA													19	2	1	Converts binary add of BCD characters into BCD format	•	•	↕	↕	↕	③
Decrement	DEC							6A	6	2	7A	6	3				M − 1 → M	•	•	↕	↕	④	•
	DECA													4A	2	1	A − 1 → A	•	•	↕	↕	④	•
	DECB													5A	2	1	B − 1 → B	•	•	↕	↕	④	•
Exclusive OR	EORA	88	2	2	98	3	2	A8	4	2	B8	4	3				A ⊕ M → A	•	•	↕	↕	R	•
	EORB	C8	2	2	D8	3	2	E8	4	2	F8	4	3				B ⊕ M → B	•	•	↕	↕	R	•
Increment	INC							6C	6	2	7C	6	3				M + 1 → M	•	•	↕	↕	⑤	•
	INCA													4C	2	1	A + 1 → A	•	•	↕	↕	⑤	•
	INCB													5C	2	1	B + 1 → B	•	•	↕	↕	⑤	•
Load Accumulator	LDAA	86	2	2	96	3	2	A6	4	2	B6	4	3				M → A	•	•	↕	↕	R	•
	LDAB	C6	2	2	D6	3	2	E6	4	2	F6	4	3				M → B	•	•	↕	↕	R	•
Load Double Accumulator	LDD	CC	3	3	DC	4	2	EC	5	2	FC	5	3				M + 1 → B, M → A	•	•	↕	↕	R	•
Multiply Unsigned	MUL													3D	10	1	A x B → A : B	•	•	•	•	•	⑪
OR, Inclusive	ORAA	8A	2	2	9A	3	2	AA	4	2	BA	4	3				A + M → A	•	•	↕	↕	R	•
	ORAB	CA	2	2	DA	3	2	EA	4	2	FA	4	3				B + M → B	•	•	↕	↕	R	•
Push Data	PSHA													36	3	1	A → Msp, SP − 1 → SP	•	•	•	•	•	•
	PSHB													37	3	1	B → Msp, SP − 1 → SP	•	•	•	•	•	•
Pull Data	PULA													32	4	1	SP + 1 → SP, Msp → A	•	•	•	•	•	•
	PULB													33	4	1	SP + 1 → SP, Msp → B	•	•	•	•	•	•
Rotate Left	ROL							69	6	2	79	6	3					•	•	↕	↕	⑥	↕
	ROLA													49	2	1		•	•	↕	↕	⑥	↕
	ROLB													59	2	1		•	•	↕	↕	⑥	↕
Rotate Right	ROR							66	6	2	76	6	3					•	•	↕	↕	⑥	↕
	RORA													46	2	1		•	•	↕	↕	⑥	↕
	RORB													56	2	1		•	•	↕	↕	⑥	↕

The Condition Code Register notes are listed after Table 10.

(Continued)

Table 7 Accumulator & Memory Instructions (Continued)

Operations	Mnemonic	Addressing Modes															Boolean/ Arithmetic Operation	Condition Code Register					
		IMMED.			DIRECT			INDEX			EXTEND			IMPLIED				5	4	3	2	1	0
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#		H	I	N	Z	V	C
Shift Left Arithmetic	ASL							68	6	2	78	6	3				M	•	•	↕	↕	⑥	↕
	ASLA													48	2	1	A	•	•	↕	↕	⑥	↕
	ASLB													58	2	1	B	•	•	↕	↕	⑥	↕
Double Shift Left, Arithmetic	ASLD													05	3	1		•	•	↕	↕	⑥	↕
Shift Right Arithmetic	ASR							67	6	2	77	6	3				M	•	•	↕	↕	⑥	↕
	ASRA													47	2	1	A	•	•	↕	↕	⑥	↕
	ASRB													57	2	1	B	•	•	↕	↕	⑥	↕
Shift Right Logical	LSR							64	6	2	74	6	3				M	•	•	↕	↕	⑥	↕
	LSRA													44	2	1	A	•	•	↕	↕	⑥	↕
	LSRB													54	2	1	B	•	•	↕	↕	⑥	↕
Double Shift Right Logical	LSRD													04	3	1		•	•	R	↕	⑥	↕
Store Accumulator	STAA				97	3	2	A7	4	2	B7	4	3				A → M	•	•	↕	↕	R	•
	STAB				D7	3	2	E7	4	2	F7	4	3				B → M	•	•	↕	↕	R	•
Store Double Accumulator	STD				DD	4	2	ED	5	2	FD	5	3				A → M B → M + 1	•	•	↕	↕	R	•
Subtract	SUBA	80	2	2	90	3	2	A0	4	2	B0	4	3				A – M → A	•	•	↕	↕	↕	↕
	SUBB	C0	2	2	D0	3	2	E0	4	2	F0	4	3				B – M → B	•	•	↕	↕	↕	↕
Double Subtract	SUBD	83	4	3	93	5	2	A3	6	2	B3	6	3				A : B – M : M + 1 → A : B	•	•	↕	↕	↕	↕
Subtract Accumulators	SBA													10	2	1	A – B → A	•	•	↕	↕	↕	↕
Subtract With Carry	SBCA	82	2	2	92	3	2	A2	4	2	B2	4	3				A – M – C → A	•	•	↕	↕	↕	↕
	SBCB	C2	2	2	D2	3	2	E2	4	2	F2	4	3				B – M – C → B	•	•	↕	↕	↕	↕
Transfer Accumulators	TAB													16	2	1	A → B	•	•	↕	↕	R	•
	TBA													17	2	1	B → A	•	•	↕	↕	R	•
Test Zero or Minus	TST							6D	6	2	7D	6	3				M – 00	•	•	↕	↕	R	R
	TSTA													4D	2	1	A – 00	•	•	↕	↕	R	R
	TSTB													5D	2	1	B – 00	•	•	↕	↕	R	R

The Condition Code Register notes are listed after Table 10.

Direct Addressing

In direct addressing, the address of the operand is contained in the second byte of the instruction. Direct addressing allows the user to directly address the lowest 256 bytes in the machine i.e., locations zero through 255. Enhanced execution times are achieved by storing data in these locations. In most configurations, it should be a random access memory. These are two-byte instructions.

Extended Addressing

In extended addressing, the address contained in the second byte of the instruction is used as the higher eight-bits of the address of the operand. The third byte of the instruction is used as the lower eight-bits of the address for the operand. This is an absolute address in memory. These are three-byte instructions.

Indexed Addressing

In indexed addressing, the address contained in the second byte of the instruction is added to the index register's lowest eight bits in the MCU. The carry is then added to the higher order eight bits of the index register. This result is then used to address memory. The modified address is held in a temporary address register so there is no change to the index register. These are two-byte instructions.

Implied Addressing

In the implied addressing mode the instruction gives the address (i.e., stack pointer, index register, etc.). These are one-byte instructions.

Relative Addressing

In relative addressing, the address contained in the second byte of the instruction is added to the program counter's lowest eight bits plus two. The carry or borrow is then added to the high eight bits. This allows the user to address data within a range of -126 to +129 bytes of the present instruction. These are two-byte instructions.

- **New Instructions**

In addition to the existing 6800 Instruction Set, the following new instructions are incorporated in the HD6801S Microcomputer.

ABX Adds the 8-bit unsigned accumulator B to the 16-bit X-Register taking into account the possible carry out of the low order byte of the X-Register.

ADDD Adds the double precision ACCD* to the double precision value M:M+1 and places the results in ACCD.

ASLD Shifts all bits of ACCD one place to the left. Bit 0 is loaded with zero. The C bit is loaded from the most significant bit of ACCD.

LDD Loads the contents of double precision memory location into the double accumulator A:B. The condition codes are set according to the data.

LSRD Shifts all bits of ACCD one place to the right. Bit 15 is loaded with zero. The C bit is loaded from the least significant bit to ACCD.

MUL Multiplies the 8 bits in accumulator A with the 8 bits in accumulator B to obtain a 16-bit unsigned number in A:B, ACCA contains MSB of result.

PSHX The contents of the index register is pushed onto the stack at the address contained in the stack pointer. The stack pointer is decremented by 2.

PULX The index register is pulled from the stack beginning at the current address contained in the stack pointer +1. The stack pointer is incremented by 2 in total.

STD Stores the contents of double accumulator A:B in memory. The contents of ACCD remain unchanged.

SUBD Subtracts the contents of M:M + 1 from the contents of double accumulator AB and places the result in ACCD.

BRN Never branches. If effect, this instruction can be considered a two byte NOP (No operation) requiring three cycles for execution.

CPX Internal processing modified to permit its use with any conditional branch instruction.

*ACCD is the 16 bit register (A:B) formed by concatenating the A and B accumulators. The A-accumulator is the most significant byte.

Table 8 Index Register and Stack Manipulation Instructions

Pointer Operations	Mnemonic	Addressing Modes															Boolean/ Arithmetic Operation	Condition Code Register					
		IMMED.			DIRECT			INDEX			EXTND			IMPLIED				5	4	3	2	1	0
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#		H	I	N	Z	V	C
Compare Index Reg	CPX	8C	4	3	9C	5	2	AC	6	2	BC	6	3				X–M: M + 1	•	•	↕	↕	↕	↕
Decrement Index Reg	DEX													09	3	1	X – 1 → X	•	•	•	↕	•	•
Decrement Stack Pntr	DES													34	3	1	SP – 1 → SP	•	•	•	•	•	•
Increment Index Reg	INX													08	3	1	X + 1 → X	•	•	•	↕	•	•
Increment Stack Pntr	INS													31	3	1	SP + 1 → SP	•	•	•	•	•	•
Load Index Reg	LDX	CE	3	3	DE	4	2	EE	5	2	FE	5	3				M → X_H, (M + 1) → X_L	•	•	⑦	↕	R	•
Load Stack Pntr	LDS	8E	3	3	9E	4	2	AE	5	2	BE	5	3				M → SP_H, (M+1) → SP_L	•	•	⑦	↕	R	•
Store Index Reg	STX				DF	4	2	EF	5	2	FF	5	3				X_H → M, X_L → (M + 1)	•	•	⑦	↕	R	•
Store Stack Pntr	STS				9F	4	2	AF	5	2	BF	5	3				SP_H → M, SP_L → (M+1)	•	•	⑦	↕	R	•
Index Reg → Stack Pntr	TXS													35	3	1	X – 1 → SP	•	•	•	•	•	•
Stack Pntr → Index Reg	TSX													30	3	1	SP + 1 → X	•	•	•	•	•	•
Add	ABX													3A	3	1	B + X → X	•	•	•	•	•	•
Push Data	PSHX													3C	4	1	X_L → M_{sp}, SP – 1 → SP X_H → M_{sp}, SP – 1 → SP	•	•	•	•	•	•
Pull Data	PULX													38	5	1	SP + 1 → SP, M_{sp} → X_H SP + 1 → SP, M_{sp} → X_L	•	•	•	•	•	•

The Condition Code Register notes are listed after Table 10.

Table 9 Jump and Branch Instructions

Operations	Mnemonic	Addressing Modes														Branch Test	Condition Code Register					
		RELATIVE			DIRECT			INDEX			EXTND			IMPLIED				5	4	3	2	1	0
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#		H	I	N	Z	V	C
Branch Always	BRA	20	3	2													None	•	•	•	•	•	•
Branch Never	BRN	21	3	2													None	•	•	•	•	•	•
Branch If Carry Clear	BCC	24	3	2													C = 0	•	•	•	•	•	•
Branch If Carry Set	BCS	25	3	2													C = 1	•	•	•	•	•	•
Branch If = Zero	BEQ	27	3	2													Z = 1	•	•	•	•	•	•
Branch If ≥ Zero	BGE	2C	3	2													N ⊕ V = 0	•	•	•	•	•	•
Branch If > Zero	BGT	2E	3	2													Z + (N ⊕ V) = 0	•	•	•	•	•	•
Branch If Higher	BHI	22	3	2													C + Z = 0	•	•	•	•	•	•
Branch If ≤ Zero	BLE	2F	3	2													Z + (N ⊕ V) = 1	•	•	•	•	•	•
Branch If Lower Or Same	BLS	23	3	2													C + Z = 1	•	•	•	•	•	•
Branch If < Zero	BLT	2D	3	2													N ⊕ V = 1	•	•	•	•	•	•
Branch If Minus	BMI	2B	3	2													N = 1	•	•	•	•	•	•
Branch If Not Equal Zero	BNE	26	3	2													Z = 0	•	•	•	•	•	•
Branch If Overflow Clear	BVC	28	3	2													V = 0	•	•	•	•	•	•
Branch If Overflow Set	BVS	29	3	2													V = 1	•	•	•	•	•	•
Branch If Plus	BPL	2A	3	2													N = 0	•	•	•	•	•	•
Branch To Subroutine	BSR	8D	6	2													See Special Operations	•	•	•	•	•	•
Jump	JMP							6E	3	2	7E	3	3					•	•	•	•	•	•
Jump To Subroutine	JSR				9D	5	2	AD	6	2	BD	6	3					•	•	•	•	•	•
No Operation	NOP													01	2	1	Advances Prog. Cntr. Only	•	•	•	•	•	•
Return From Interrupt	RTI													3B	10	1	See Special Operations	⑧					
Return From Subroutine	RTS													39	5	1		•	•	•	•	•	•
Software Interrupt	SWI													3F	12	1		•	S	•	•	•	•
Wait for Interrupt	WAI													3E	9	1		•	⑨	•	•	•	•

Table10 Condition Code Register Manipulation Instructions

Operations	Mnemonic	AddressingModes			Boolean Operation	Condition Code Register					
		IMPLIED				5	4	3	2	1	0
		OP	~	#		H	I	N	Z	V	C
Clear Carry	CLC	0C	2	1	0 → C	•	•	•	•	•	R
Clear Interrupt Mask	CLI	0E	2	1	0 → I	•	R	•	•	•	•
Clear Overflow	CLV	0A	2	1	0 → V	•	•	•	•	R	•
Set Carry	SEC	0D	2	1	1 → C	•	•	•	•	•	S
Set Interrupt Mask	SEI	0F	2	1	1 → I	•	S	•	•	•	•
Set Overflow	SEV	0B	2	1	1 → V	•	•	•	•	S	•
Accumulator A → CCR	TAP	06	2	1	A → CCR	⑩					
CCR → Accumulator A	TPA	07	2	1	CCR → A	•	•	•	•	•	•

Condition Code Register Notes: (Bit set it test is true and cleared otherwise)

① (Bit V) Test: Result = 10000000?
② (Bit C) Test: Result = 00000000?
③ (Bit C) Test: Decimal value of most significant BCD Character greater than nine? (Not cleared if previously set)
④ (Bit V) Test: Operand = 10000000 prior to execution?
⑤ (Bit V) Test: Operand = 01111111 prior to execution?
⑥ (Bit V) Test: Set equal to result of N⊕C after shift has occurred.
⑦ (Bit N) Test: Result less than zero? (Bit 15 = 1)
⑧ (All) Load Condition Code Register from Stack. (See Special Operations)
⑨ (Bit I) Set when interrupt occurs. If previously set, a Non-Maskable Interrupt is required to exit the wait state.
⑩ (All) Set according to the contents of Accumulator A.
⑪ (Bit C) Set equal to result of Bit 7 (AccB)

Table 11 Instruction Execution Times in Machine Cycles

	ACCX	Immediate	Direct	Extended	Indexed	Implied	Relative		ACCX	Immediate	Direct	Extended	Indexed	Implied	Relative
ABA	•	•	•	•	•	2	•	INX	•	•	•	•	•	3	•
ABX	•	•	•	•	•	3	•	JMP	•	•	•	3	3	•	•
ADC	•	2	3	4	4	•	•	JSR	•	•	5	6	6	•	•
ADD	•	2	3	4	4	•	•	LDA	•	2	3	4	4	•	•
ADDD	•	4	5	6	6	•	•	LDD	•	3	4	5	5	•	•
AND	•	2	3	4	4	•	•	LDS	•	3	4	5	5	•	•
ASL	2	•	•	6	6	•	•	LDX	•	3	4	5	5	•	•
ASLD	•	•	•	•	•	3	•	LSR	2	•	•	6	6	•	•
ASR	2	•	•	6	6	•	•	LSRD	•	•	•	•	•	3	•
BCC	•	•	•	•	•	•	3	MUL	•	•	•	•	•	10	•
BCS	•	•	•	•	•	•	3	NEG	2	•	•	6	6	•	•
BEQ	•	•	•	•	•	•	3	NOP	•	•	•	•	•	2	•
BGE	•	•	•	•	•	•	3	ORA	•	2	3	4	4	•	•
BGT	•	•	•	•	•	•	3	PSH	3	•	•	•	•	•	•
BHI	•	•	•	•	•	•	3	PSHX	•	•	•	•	•	4	•
BIT	•	2	3	4	4	•	•	PUL	4	•	•	•	•	•	•
BLE	•	•	•	•	•	•	3	PULX	•	•	•	•	•	5	•
BLS	•	•	•	•	•	•	3	ROL	2	•	•	6	6	•	•
BLT	•	•	•	•	•	•	3	ROR	2	•	•	6	6	•	•
BMI	•	•	•	•	•	•	3	RTI	•	•	•	•	•	10	•
BNE	•	•	•	•	•	•	3	RTS	•	•	•	•	•	5	•
BPL	•	•	•	•	•	•	3	SBA	•	•	•	•	•	2	•
BRA	•	•	•	•	•	•	3	SBC	•	2	3	4	4	•	•
BRN	•	•	•	•	•	•	3	SEC	•	•	•	•	•	2	•
BSR	•	•	•	•	•	•	6	SEI	•	•	•	•	•	2	•
BVC	•	•	•	•	•	•	3	SEV	•	•	•	•	•	2	•
BVS	•	•	•	•	•	•	3	STA	•	•	3	4	4	•	•
CBA	•	•	•	•	•	2	•	STD	•	•	4	5	5	•	•
CLC	•	•	•	•	•	2	•	STS	•	•	4	5	5	•	•
CLI	•	•	•	•	•	2	•	STX	•	•	4	5	5	•	•
CLR	2	•	•	6	6	•	•	SUB	•	2	3	4	4	•	•
CLV	•	•	•	•	•	2	•	SUBD	•	4	5	6	6	•	•
CMP	•	2	3	4	4	•	•	SWI	•	•	•	•	•	12	•
COM	2	•	•	6	6	•	•	TAB	•	•	•	•	•	2	•
CPX	•	4	5	6	6	•	•	TAP	•	•	•	•	•	2	•
DAA	•	•	•	•	•	2	•	TBA	•	•	•	•	•	2	•
DEC	2	•	•	6	6	•	•	TPA	•	•	•	•	•	2	•
DES	•	•	•	•	•	3	•	TST	2	•	•	6	6	•	•
DEX	•	•	•	•	•	3	•	TSX	•	•	•	•	•	3	•
EOR	•	2	3	4	4	•	•	TXS	•	•	•	•	•	3	•
INC	2	•	•	6	6	•	•	WAI	•	•	•	•	•	9	•
INS	•	•	•	•	•	3	•								

- **Summary of Cycle by Cycle Operation**

Table 12 provides a detailed description of the information present on the Address Bus, Data Bus, and the Read/Write line (R/$\overline{W}$) during each cycle for each instruction.

This information is useful in comparing actual with expected results during debug of both software and hardware as the control program is executed. The information is categorized in groups according to addressing mode and number of cycles per instruction. (In general, instructions with the same addressing mode and number of cycles execute in the same manner; exceptions are indicated in the table).

Table 12 Cycle by Cycle Operation

Address Mode & Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
IMMEDIATE					
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	2	1 2	Op Code Address Op Code Address + 1	1 1	Op Code Operand Data
LDS LDX	3	1 2 3	Op Code Address Op Code Address + 1 Op Code Address + 2	1 1 1	Op Code Operand Data (High Order Byte) Operand Data (Low Order Byte)
CPX SUBD ADDD	4	1 2 3 4	Op Code Address Op Code Address + 1 Op Code Address + 2 Address Bus FFFF	1 1 1 1	Op Code Operand Data (High Order Byte) Operand Data (Low Order Byte) Low Byte of Restart Vector
DIRECT					
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	3	1 2 3	Op Code Address Op Code Address + 1 Address of Operand	1 1 1	Op Code Address of Operand Operand Data
STA	3	1 2 3	Op Code Address Op Code Address + 1 Destination Address	1 1 0	Op Code Destination Address Data from Accumulator
LDS LDX LDD	4	1 2 3 4	Op Code Address Op Code Address + 1 Address of Operand Operand Address + 1	1 1 1 1	Op Code Address of Operand Operand Data (High Order Byte) Operand Data (Low Order Byte)
STS STX STD	4	1 2 3 4	Op Code Address Op Code Address + 1 Address of Operand Address of Operand + 1	1 1 0 0	Op Code Address of Operand Register Data (High Order Byte) Register Data (Low Order Byte)
CPX SUBD ADDD	5	1 2 3 4 5	Op Code Address Op Code Address + 1 Operand Address Operand Address + 1 Address Bus FFFF	1 1 1 1 1	Op Code Address of Operand Operand Data (High Order Byte) Operand Data (Low Order Byte) Low Byte of Restart Vector
JSR	5	1 2 3 4 5	Op Code Address Op Code Address + 1 Subroutine Address Stack Pointer Stack Pointer + 1	1 1 1 0 0	Op Code Irrelevant Data First Subroutine Op Code Return Address (Low Order Byte) Return Address (High Order Byte)

(Continued)

Table 12 Cycle by Cycle Operation (Continued)

Address Mode & Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
INDEXED					
JMP	3	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
ADC EOR	4	1	Op Code Address	1	Op Code
ADD LDA		2	Op Code Address + 1	1	Offset
AND ORA		3	Address Bus FFFF	1	Low Byte of Restart Vector
BIT SBC		4	Index Register Plus Offset	1	Operand Data
CMP SUB					
STA	4	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	0	Operand Data
LDS	5	1	Op Code Address	1	Op Code
LDX		2	Op Code Address + 1	1	Offset
LDD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	1	Operand Data (High Order Byte)
		5	Index Register Plus Offset + 1	1	Operand Data (Low Order Byte)
STS	5	1	Op Code Address	1	Op Code
STX		2	Op Code Address + 1	1	Offset
STD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	0	Operand Data (High Order Byte)
		5	Index Register Plus Offset + 1	0	Operand Data (Low Order Byte)
ASL LSR	6	1	Op Code Address	1	Op Code
ASR NEG		2	Op Code Address + 1	1	Offset
CLR ROL		3	Address Bus FFFF	1	Low Byte of Restart Vector
COM ROR		4	Index Register Plus Offset	1	Current Operand Data
DEC TST*		5	Address Bus FFFF	1	Low Byte of Restart Vector
INC		6	Index Register Plus Offset	0	New Operand Data
CPX	6	1	Op Code Address	1	Op Code
SUBD		2	Op Code Address + 1	1	Offset
ADDD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register + Offset	1	Operand Data (High Order Byte)
		5	Index Register + Offset + 1	1	Operand Data (Low Order Byte)
		6	Address Bus FFFF	1	Low Byte of Restart Vector
JSR	6	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register + Offset	1	First Subroutine Op Code
		5	Stack Pointer	0	Return Address (Low Order Byte)
		6	Stack Pointer – 1	0	Return Address (High Order Byte)

*In the TST instruction, R/$\overline{W}$ line of the sixth cycle is "1" level, and AB = FFFF, DB = Low Byte of Reset Vector. (Continued)

Table 12 Cycle by Cycle Operation (Continued)

Address Mode & Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
EXTENDED					
JMP	3	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Jump Address (High Order Byte)
		3	Op Code Address + 2	1	Jump Address (Low Order Byte)
ADC EOR	4	1	Op Code Address	1	Op Code
ADD LDA		2	Op Code Address + 1	1	Address of Operand
AND ORA		3	Op Code Address + 2	1	Address of Operand (Low Order Byte)
BIT SBC		4	Address of Operand	1	Operand Data
CMP SUB					
STA A	4	1	Op Code Address	1	Op Code
STA B		2	Op Code Address + 1	1	Destination Address (High Order Byte)
		3	Op Code Address + 2	1	Destination Address (Low Order Byte)
		4	Operand Destination Address	0	Data from Accumulator
LDS	5	1	Op Code Address	1	Op Code
LDX		2	Op Code Address + 1	1	Address of Operand (High Order Byte)
LDD		3	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	Address of Operand	1	Operand Data (High Order Byte)
		5	Address of Operand + 1	1	Operand Data (Low Order Byte)
STS	5	1	Op Code Address	1	Op Code
STX		2	Op Code Address + 1	1	Address of Operand (High Order Byte)
STD		3	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	Address of Operand	0	Operand Data (High Order Byte)
		5	Address of Operand + 1	0	Operand Data (Low Order Byte)
ASL LSR	6	1	Op Code Address	1	Op Code
ASR NEG		2	Op Code Address + 1	1	Address of Operand (High Order Byte)
CLR ROL		3	Op Code Address + 2	1	Address of Operand (Low Order Byte)
COM ROR		4	Address of Operand	1	Current Operand Data
DEC TST*		5	Address Bus FFFF	1	Low Byte of Restart Vector
INC		6	Address of Operand	0	New Operand Data
CPX	6	1	Op Code Address	1	Op Code
SUBD		2	Op Code Address + 1	1	Operand Address (High Order Byte)
ADDD		3	Op Code Address + 2	1	Operand Address (Low Order Byte)
		4	Operand Address	1	Operand Data (High Order Byte)
		5	Operand Address + 1	1	Operand Data (Low Order Byte)
		6	Address Bus FFFF	1	Low Byte of Restart Vector
JSR	6	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Address of Subroutine (High Order Byte)
		3	Op Code Address + 2	1	Address of Subroutine (Low Order Byte)
		4	Subroutine Starting Address	1	Op Code of Next Instruction
		5	Stack Pointer	0	Return Address (Low Order Byte)
		6	Stack Pointer – 1	0	Address of Operand (High Order Byte)

*In the TST instruction, R/$\overline{W}$ line of the sixth cycle is "1" level, and AB=FFFF, DB=Low Byte of Reset Vector.

(Continued)

Table 12 Cycle by Cycle Operation (Continued)

Address Mode & Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
IMPLIED					
ABA DAA SEC ASL DEC SEI ASR INC SEV CBA LSR TAB CLC NEG TAP CLI NOP TBA CLR ROL TPA CLV ROR TST COM SBA	2	1 2	Op Code Address Op Code Address + 1	1 1	Op Code Op Code of Next Instruction
ABX	3	1 2 3	Op Code Address Op Code Address + 1 Address Bus FFFF	1 1 1	Op Code Irrelevant Data Low Byte of Restart Vector
ASLD LSRD	3	1 2 3	Op Code Address Op Code Address + 1 Address Bus FFFF	1 1 1	Op Code Irrelevant Data Low Byte of Restart Vector
DES INS	3	1 2 3	Op Code Address Op Code Address + 1 Previous Register Contents	1 1 1	Op Code Op Code of Next Instruction Irrelevant Data
INX DEX	3	1 2 3	Op Code Address Op Code Address + 1 Address Bus FFFF	1 1 1	Op Code Op Code of Next Instruction Low Byte of Restart Vector
PSHA PSHB	3	1 2 3	Op Code Address Op Code Address + 1 Stack Pointer	1 1 0	Op Code Op Code of Next Instruction Accumulator Data
TSX	3	1 2 3	Op Code Address Op Code Address + 1 Stack Pointer	1 1 1	Op Code Op Code of Next Instruction Irrelevant Data
TXS	3	1 2 3	Op Code Address Op Code Address + 1 Address Bus FFFF	1 1 1	Op Code Op Code of Next Instruction Low Byte of Restart Vector
PULA PULB	4	1 2 3 4	Op Code Address Op Code Address + 1 Stack Pointer Stack Pointer + 1	1 1 1 1	Op Code Op Code of Next Instruction Irrelevant Data
PSHX	4	1 2 3 4	Op Code Address Op Code Address + 1 Stack Pointer Stack Pointer – 1	1 1 0 0	Op Code Irrelevant Data Index Register (Low Order Byte) Index Register (High Order Byte)
PULX	5	1 2 3 4 5	Op Code Address Op Code Address + 1 Stack Pointer Stack Pointer + 1 Stack Pointer +2	1 1 1 1 1	Op Code Irrelevant Data Irrelevant Data Index Register (High Order Byte) Index Register (Low Order Byte)
RTS	5	1 2 3 4 5	Op Code Address Op Code Address + 1 Stack Pointer Stack Pointer + 1 Stack Pointer + 2	1 1 1 1 1	Op Code Irrelevant Data Irrelevant Data Address of Next Instruction (High Order Byte) Address of Next Instruction (Low Order Byte)
WAI**	9	1 2 3 4	Op Code Address Op Code Address + 1 Stack Pointer Stack Pointer – 1	1 1 0 0	Op Code Op Code of Next Instruction Return Address (Low Order Byte) Return Address (High Order Byte)

(Continued)

Table 12 Cycle by Cycle Operation (Continued)

Address Mode & Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
WAI**		5	Stack Pointer – 2	0	Index Register (Low Order Byte)
		6	Stack Pointer – 3	0	Index Register (High Order Byte)
		7	Stack Pointer – 4	0	Contents of Accumulator A
		8	Stack Pointer – 5	0	Contents of Accumulator B
		9	Stack Pointer – 6	0	Contents of Cond. Code Register
MUL	10	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Irrelevant Data
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Address Bus FFFF	1	Low Byte of Restart Vector
		5	Address Bus FFFF	1	Low Byte of Restart Vector
		6	Address Bus FFFF	1	Low Byte of Restart Vector
		7	Address Bus FFFF	1	Low Byte of Restart Vector
		8	Address Bus FFFF	1	Low Byte of Restart Vector
		9	Address Bus FFFF	1	Low Byte of Restart Vector
		10	Address Bus FFFF	1	Low Byte of Restart Vector
RTI	10	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Irrelevant Data
		3	Stack Pointer	1	Irrelevant Data
		4	Stack Pointer + 1	1	Contents of Cond. Code Reg. from Stack
		5	Stack Pointer + 2	1	Contents of Accumulator B from Stack
		6	Stack Pointer + 3	1	Contents of Accumulator A from Stack
		7	Stack Pointer + 4	1	Index Register from Stack (High Order Byte)
		8	Stack Pointer + 5	1	Index Register from Stack (Low Order Byte)
		9	Stack Pointer + 6	1	Next Instruction Address from Stack (High Order Byte)
		10	Stack Pointer + 7	1	Next Instruction Address from Stack (Low Order Byte)
SWI	12	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Irrelevant Data
		3	Stack Pointer	0	Return Address (Low Order Byte)
		4	Stack Pointer – 1	0	Return Address (High Order Byte)
		5	Stack Pointer – 2	0	Index Register (Low Order Byte)
		6	Stack Pointer – 3	0	Index Register (High Order Byte)
		7	Stack Pointer – 4	0	Contents of Accumulator A
		8	Stack Pointer – 5	0	Contents of Accumulator B
		9	Stack Pointer – 6	0	Contents of Cond. Code Register
		10	Stack Pointer – 7	1	Irrelevant Data
		11	Vector Address FFFA (Hex)	1	Address of Subroutine (High Order Byte)
		12	Vector Address FFFB (Hex)	1	Address of Subroutine (Low Order Byte)

**While the MPU is in the "Wait" state, its bus state will appear as a series of MPU reads of an address which is seven locations less than the original contents of the Stack Pointer. Contrary to the HD6800, none of the ports are driven to the high impedance state by a WAI instruction.

(Continued)

Table 12 Cycle by Cycle Operation (Continued)

Address Mode & Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
RELATIVE					
BCC BHT BNE BCS BLE BPL BEQ BLS BRA BGE BLT BVC BGT BMT BVS BRN	3	1 2 3	Op Code Address Op Code Address + 1 Address Bus FFFF	1 1 1	Op Code Branch Offset Low Byte of Restart Vector
BSR	6	1 2 3 4 5 6	Op Code Address Op Code Address + 1 Address Bus FFFF Subroutine Starting Address Stack Pointer Stack Pointer – 1	1 1 1 1 0 0	Op Code Branch Offset Low Byte of Restart Vector Op Code of Next Instruction Return Address (Low Order Byte) Return Address (High Order Byte)

- **Summary of Undefined Instruction Operations**

The HD6801S has 36 undefined instructions. When these are carried out, the contents of Register and Memory in MPU change at random.

When the op codes (4E, 5E) are used to execute, the MPU continues to increase the program counter and it will not stop until the Reset signal enters. These op codes are used to test the LSI.

Table 13 Op codes Map

HD6801S MICROCOMPUTER INSTRUCTIONS																		
OP CODE						ACC A	ACC B	IND	EXT	ACCA or SP				ACCB or X				
										IMM	DIR	IND	EXT	IMM	DIR	IND	EXT	
LO \ HI		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111	
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0000	0		SBA	BRA	TSX	NEG				SUB								0
0001	1	NOP	CBA	BRN	INS					CMP								1
0010	2			BHI	PULA (+1)					SBC								2
0011	3			BLS	PULB (+1)	COM				*	SUBD (+2)			*	ADDD (+2)			3
0100	4	LSRD (+1)		BCC	DES	LSR				AND								4
0101	5	ASLD (+1)		BCS	TXS					BIT								5
0110	6	TAP	TAB	BNE	PSHA	ROR				LDA								6
0111	7	TPA	TBA	BEQ	PSHB	ASR					STA				STA			7
1000	8	INX (+1)		BVC	PULX (+2)	ASL				EOR								8
1001	9	DEX (+1)	DAA	BVS	RTS (+2)	ROL				ADC								9
1010	A	CLV		BPL	ABX	DEC				ORA								A
1011	B	SEV	ABA	BMI	RTI (+7)					ADD								B
1100	C	CLC		BGE	PSHX (+1)	INC				*	CPX (+2)			*	LDD (+1)			C
1101	D	SEC		BLT	MUL (+7)	TST				BSR (+4)	JSR (+2)			* (+1)	STD (+1)			D
1110	E	CLI		BGT	WAI (+6)	**		JMP (-3)		*	LDS (+1)			*	LDX (+1)			E
1111	F	SEI		BLE	SWI (+9)	CLR				* (+1)	STS (+1)			* (+1)	STX (+1)			F
BYTE/CYCLE		1/2	1/2	2/3	1/3	1/2	1/2	2/6	3/6	2/2	2/3	2/4	3/4	2/2	2/3	2/4	3/4	

[NOTES] 1) Undefined Op codes are marked with ▱.

2) ( ) indicate that the number in parenthesis must be added to the cycle count for that instruction.

3) The instructions shown below are all 3 bytes and are marked with "*".
Immediate addressing mode of SUBD, CPX, LDS, ADDD, LDD and LDX instructions, and undefined op codes (8F, CD, CF).

4) The Op codes (4E, 5E) are 1 byte/∞ cycles instructions, and are marked with "**".

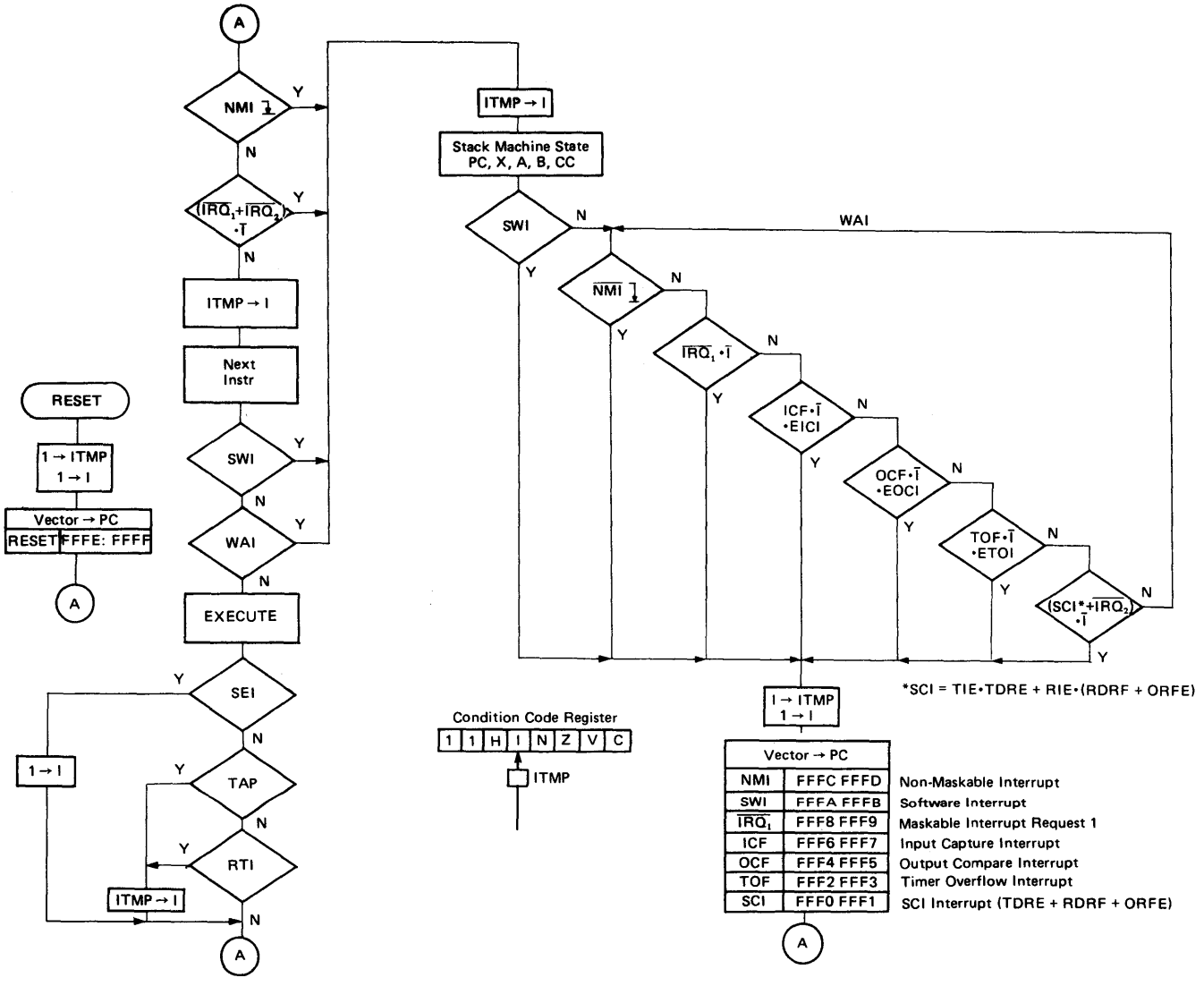


Figure 24 Interrupt Flowchart

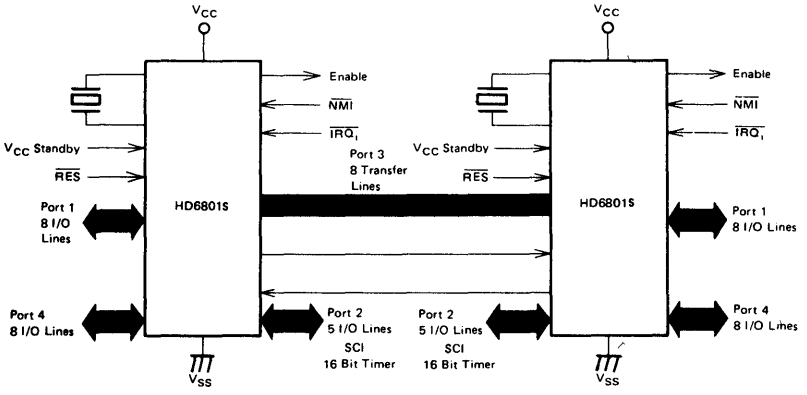


Figure 25 HD6801S MCU Single-Chip Dual Processor Configuration

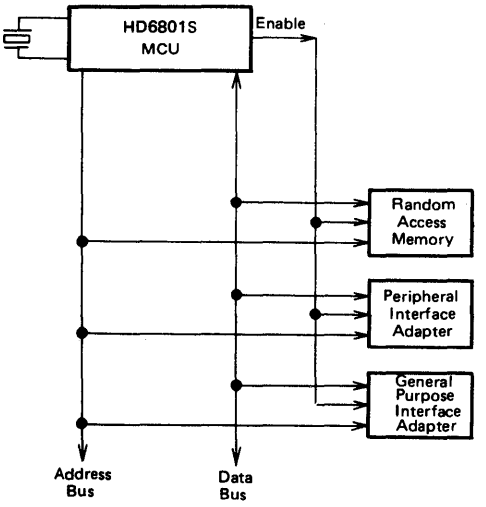


Figure 26 HD6801S MCU Expanded Non-Multiplexed Mode

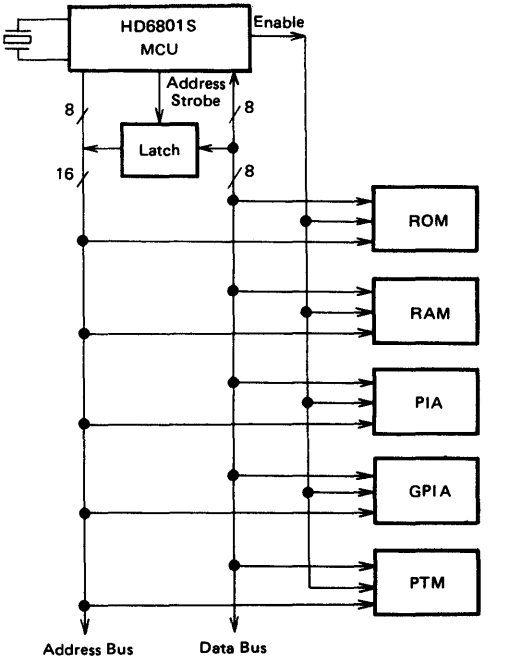


Figure 27 HD6801S MCU Expanded Multiplexed Mode

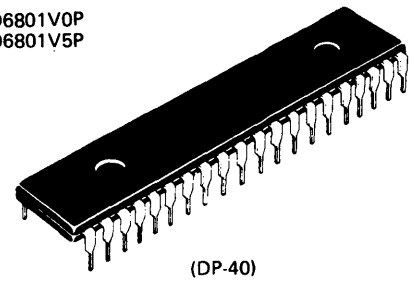
HD6801V0, HD6801V5

MCU (Microcomputer Unit)

—PRELIMINARY—

The HD6801V MCU is an 8-bit microcomputer system which is compatible with the HD6801S except the ROM size. The HD6801V MCU is object code compatible with the HD6800 with improved execution times of key instructions plus several new 16-bit and 8-bit instructions including an 8x8 unsigned multiply with 16-bit result. The HD6801V MCU can operate as a single chip microcomputer or be expanded to 65k words. The HD6801V MCU is TTL compatible and requires one +5.0 volt power supply. The HD6801V MCU has 4k bytes of ROM and 128 bytes of RAM on chip. Serial Communications interface (SCI), and parallel I/O as well as a three function 16-bit timer. Features and Block diagram of the HD6801V include the following:

HD6801V0P
HD6801V5P

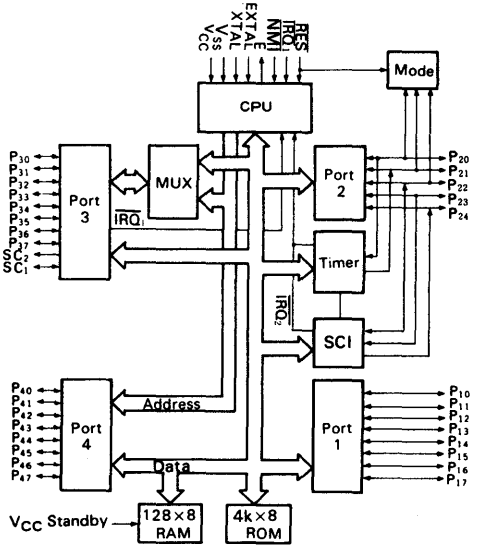


(DP-40)

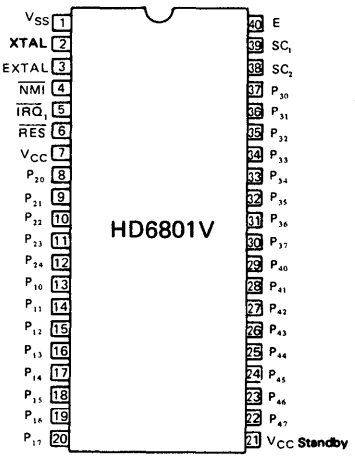
■ FEATURES

- Expanded HMCS6800 Instruction Set
- 8 x 8 Multiply
- On-Chip Serial Communications Interface (SCI)
- Object Code Compatible With The HD6800 MPU
- 16-Bit Timer
- Single Chip Or Expandable To 65k Words
- 4k Bytes Of ROM
- 128 Bytes Of RAM (64 Bytes Retainable On Power Down)
- 29 Parallel I/O Lines And 2 Handshake Control Lines
- Internal Clock/Divided-By-Four Circuitry
- TTL Compatible Inputs And Outputs
- Interrupt Capability
- Compatible with MC6801 (except ROM size)

■ BLOCK DIAGRAM



■ PIN ARRANGEMENT



(Top View)

■ TYPE OF PRODUCTS

MCU	Bus Timing
HD6801V0	1 MHz
HD6801V5	1.25 MHz

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC} *	-0.3 ~ +7.0	V
Input Voltage	V_{in} *	-0.3 ~ +7.0	V
Operating Temperature	T_{opr}	0 ~ +70	°C
Storage Temperature	T_{stg}	-55 ~ +150	°C

* With respect to V_{SS} (SYSTEM GND)

[NOTE] Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ ELECTRICAL CHARACTERISTICS

● DC CHARACTERISTICS (V_{CC} =5.0V±5%, V_{SS} = 0V, Ta = 0 ~ +70°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Input "High" Voltage	$\overline{RES}$	V_{IH}		4.0	–	V_{CC}	V
	Other Inputs*			2.0	–	V_{CC}	
Input "Low" Voltage	All Inputs*	V_{IL}		-0.3	–	0.8	V
Input Load Current	P_{40} ~ P_{47}	$\|I_{in}\|$	V_{in} = 0 ~ 2.4V	–	–	0.5	mA
	SC_1			–	–	0.8	
	EXTAL		V_{in} = 0 ~ V_{CC}	–	–	0.8	
Input Leakage Current	$\overline{NMI}$, $\overline{IRQ_1}$, $\overline{RES}$	$\|I_{in}\|$	V_{in} = 0 ~ 5.25V	–	–	2.5	μA
Three State (Offset) Leakage Current	P_{10} ~ P_{17}, P_{30} ~ P_{37}	$\|I_{TSI}\|$	V_{in} = 0.5 ~ 2.4V	–	–	10	μA
	P_{20} ~ P_{24}			–	–	100	
Output "High" Voltage	P_{30} ~ P_{37}	V_{OH}	I_{LOAD} = -205 μA	2.4	–	–	V
	P_{40} ~ P_{47}, E, SC_1, SC_2		I_{LOAD} = -145 μA	2.4	–	–	
	Other Outputs		I_{LOAD} = -100 μA	2.4	–	–	
Output "Low" Voltage	All Outputs	V_{OL}	I_{LOAD} = 1.6 mA	–	–	0.5	V
Darlington Drive Current	P_{10} ~ P_{17}	$-I_{OH}$	V_{out} = 1.5V	1.0	–	10.0	mA
Power Dissipation		P_D		–	–	1200	mW
Input Capacitance	P_{30} ~ P_{37}, P_{40} ~ P_{47}, SC_1	C_{in}	V_{in} = 0V, Ta = 25°C, f = 1.0 MHz	–	–	12.5	pF
	Other Inputs			–	–	10.0	
V_{CC} Standby	Powerdown	V_{SBB}		4.0	–	5.25	V
	Operating	V_{SB}		4.75	–	5.25	
Standby Current	Powerdown	I_{SBB}	V_{SBB} = 4.0 V	–	–	8.0	mA

*Except Mode Programming Levels.

• AC CHARACTERISTICS

BUS TIMING (V_{CC} = 5.0V±5%, V_{SS} = 0V, Ta = 0 ~ +70°C, unless otherwise noted.)

Item		Symbol	Test Condition	HD6801V0			HD6801V5			Unit
				min	typ	max	min	typ	max	
Cycle Time		t_{cyc}		1	–	10	0.8	–	10	μs
Address Strobe Pulse Width "High"		PW_{ASH}		200	–	–	150	–	–	ns
Address Strobe Rise Time		t_{ASr}		5	–	50	5	–	50	ns
Address Strobe Fall Time		t_{ASf}		5	–	50	5	–	50	ns
Address Strobe Delay Time		t_{ASD}		60	–	–	30	–	–	ns
Enable Rise Time		t_{Er}		5	–	50	5	–	50	ns
Enable Fall Time		t_{Ef}		5	–	50	5	–	50	ns
Enable Pulse Width "High" Time		PW_{EH}		450	–	–	340	–	–	ns
Enable Pulse Width "Low" Time		PW_{EL}		450	–	–	350	–	–	ns
Address Strobe to Enable Delay Time		t_{ASED}		60	–	–	30	–	–	ns
Address Delay Time		t_{AD}	Fig. 1	–	–	260	–	–	260	ns
Address Delay Time for Latch		t_{ADL}	Fig. 2	–	–	270	–	–	260	ns
Data Set-up Write Time		t_{DSW}		225	–	–	115	–	–	ns
Data Set-up Read Time		t_{DSR}		80	–	–	70	–	–	ns
Data Hold Time	Read	t_{HR}		10	–	–	10	–	–	ns
	Write	t_{HW}		20	–	–	20	–	–	
Address Set-up Time for Latch		t_{ASL}		60	–	–	50	–	–	ns
Address Hold Time for Latch		t_{AHL}		20	–	–	20	–	–	ns
Address Hold Time		t_{AH}		20	–	–	20	–	–	ns
Peripheral Read Access Time	Non-Multiplexed Bus	(t_{ACCN})		–	–	(610)	–	–	(410)	ns
	Multiplexed Bus	(t_{ACCM})		–	–	(600)	–	–	(400)	
Oscillator stabilization Time		t_{RC}	Fig. 10	100	–	–	100	–	–	ms
Processor Control Set-up Time		t_{PCS}	Fig. 11	200	–	–	200	–	–	ns

PERIPHERAL PORT TIMING (V_{CC} = 5.0V ±5%, V_{SS} = 0V, Ta = 0 ~ +70°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Peripheral Data Setup Time	Port 1, 2, 3, 4	t_{PDSU}	Fig. 3	200	–	–	ns
Peripheral Data Hold Time	Port 1, 2, 3, 4	t_{PDH}	Fig. 3	200	–	–	ns
Delay Time, Enable Positive Transition to $\overline{OS3}$ Negative Transition		t_{OSD1}	Fig. 5	–	–	350	ns
Delay Time, Enable Positive Transition to $\overline{OS3}$ Positive Transition		t_{OSD2}	Fig. 5	–	–	350	ns
Delay Time, Enable Negative Transition to Peripheral Data Valid	Port 1, 2*, 3, 4	t_{PWD}	Fig. 4	–	–	400	ns
Delay Time, Enable Negative Transition to Peripheral CMOS Data Valid	Port 2**, 4	t_{CMOS}	Fig. 4	–	–	2.0	μs
Input Strobe Pulse Width		t_{PWIS}	Fig. 6	200	–	–	ns
Input Data Hold Time	port 3	t_{IH}	Fig. 6	50	–	–	ns
Input Data Set-up Time	Port 3	t_{IS}	Fig. 6	20	–	–	ns

*Except P_{21} **10kΩ pull up register required for Port 2

TIMER, SCI TIMING (V_{CC} = 5.0V ±5%, V_{SS} = 0V, Ta = 0 ~ +70°C, unless otherwise noted.)

Item	Symbol	Test Condition	min	typ	max	Unit
Timer Input Pulse Width	t_{PWT}		$2t_{cyc}$+200	–	–	ns
Delay Time, Enable Positive Transition to Timer Out	t_{TOD}	Fig. 7	–	–	600	ns
SCI Input Clock Cycle	t_{Scyc}		1	–	–	t_{cyc}
SCI Input Clock Pulse Width	t_{PWSCK}		0.4	–	0.6	t_{Scyc}

MODE PROGRAMMING (V_{CC} = 5.0V ±5%, V_{SS} = 0V, Ta = 0 ~ +70°C, unless otherwise noted.)

Item	Symbol	Test Condition	min	typ	max	Unit
Mode Programming Input "Low" Voltage	V_{MPL}	Fig. 8	–	–	1.7	V
Mode Programming Input "High" Voltage	V_{MPH}		4.0	–	–	V
$\overline{RES}$ "Low" Pulse Width	PW_{RSTL}		3.0	–	–	t_{cyc}
Mode Programming Set-up Time	t_{MPS}		2.0	–	–	t_{cyc}
Mode Programming Hold Time	t_{MPH}		150	–	–	ns

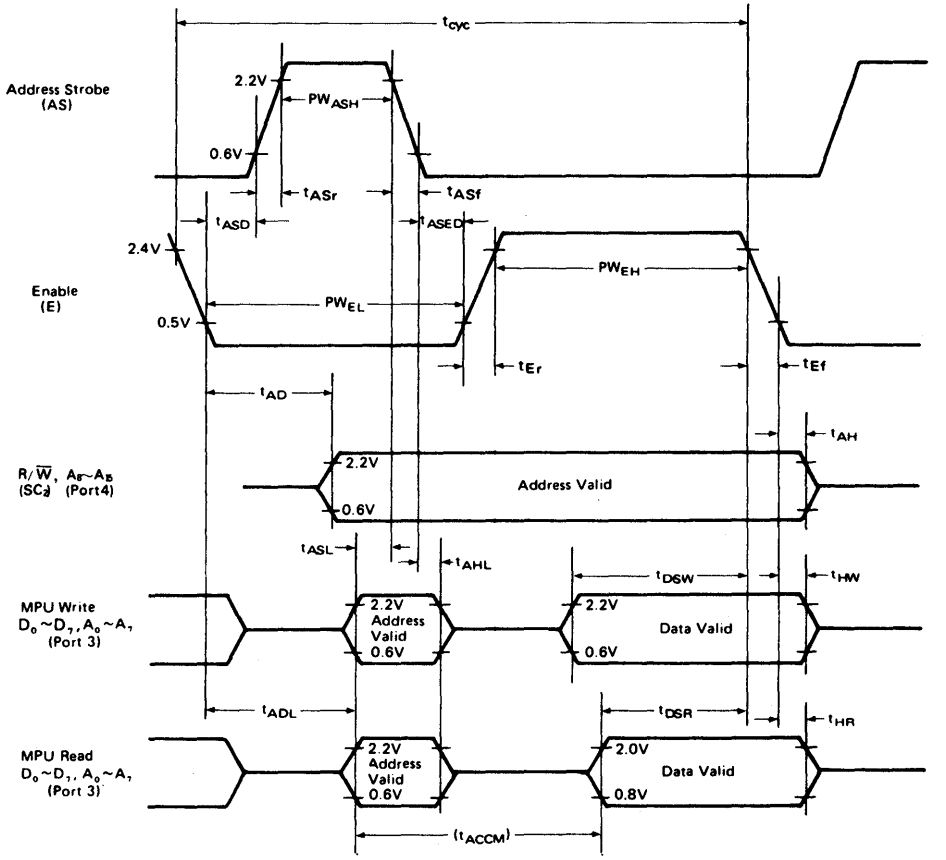


Figure 1 Expanded Multiplexed Bus Timing

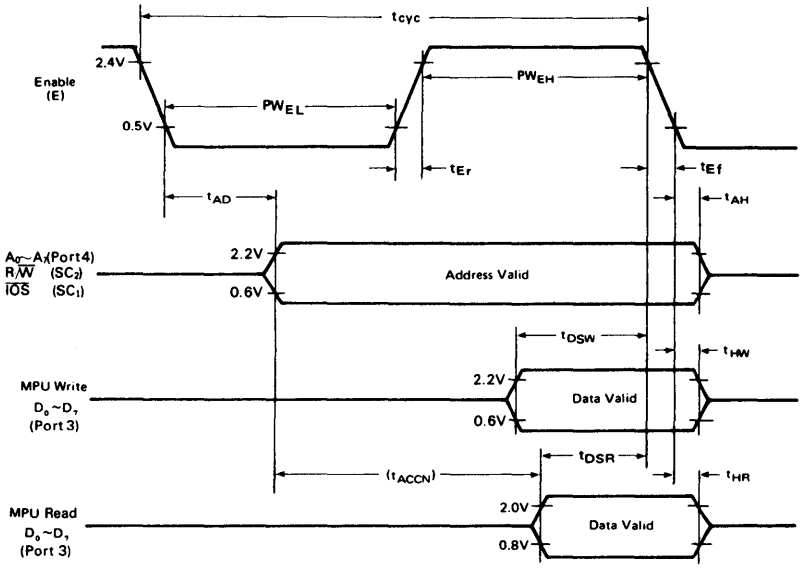
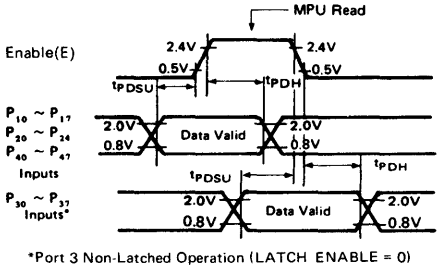
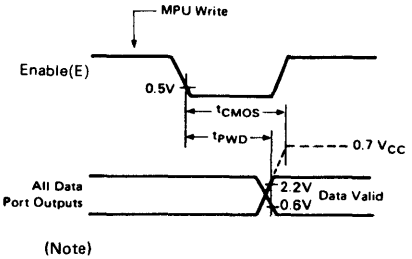


Figure 2 Expanded Non-Multiplexed Bus Timing



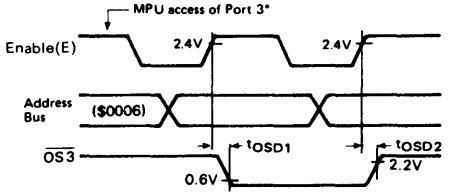
*Port 3 Non-Latched Operation (LATCH ENABLE = 0)

Figure 3 Data Set-up and Hold Times (MPU Read)



(Note)
1. 10 kΩ Pullup resistor required for Port 2 to reach 0.7 V_{CC}
2. Not applicable to P_{21}
3. Port 4 cannot be pulled above V_{CC}

Figure 4 Port Data Delay Timing (MPU Write)



*Access matches Output Strobe Select (OSS = 0, a read; OSS = 1, a write)

Figure 5 Port 3 Output Strobe Timing (Single Chip Mode)

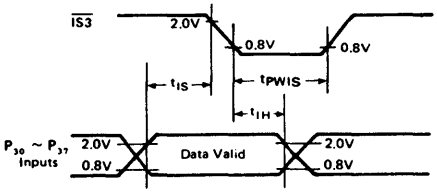


Figure 6 Port 3 Latch Timing (Single Chip Mode)

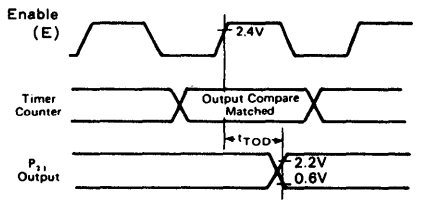


Figure 7 Timer Output Timing

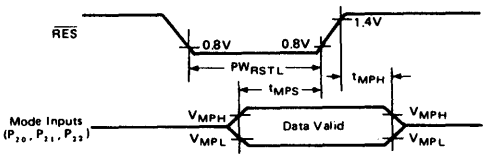
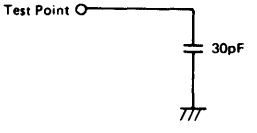
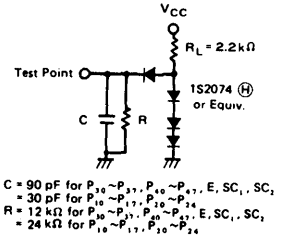


Figure 8 Mode Programming Timing

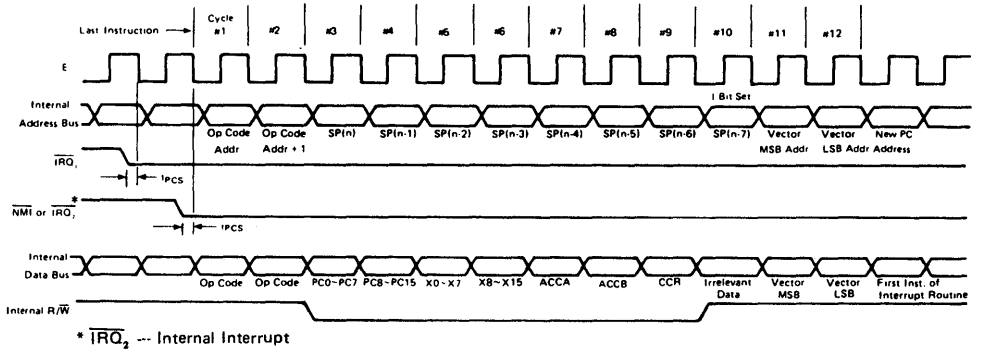


(a) CMOS Load



(b) TTL Load

Figure 9 Bus Timing Test Loads



* $\overline{IRQ_2}$ --- Internal Interrupt

Figure 10 Interrupt Sequence

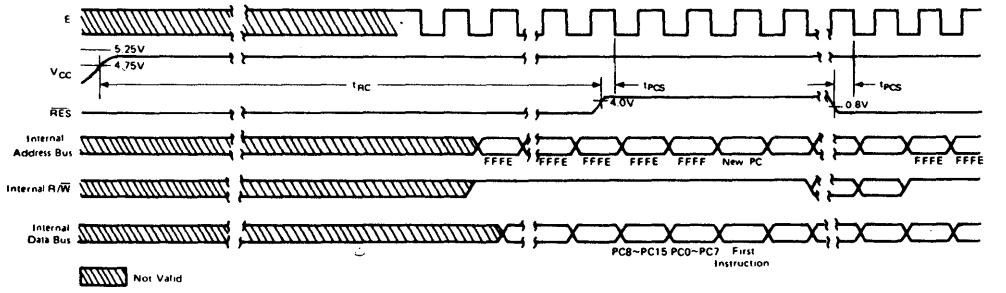


Figure 11 Reset Timing

■ SIGNAL DESCRIPTIONS

• Vcc and Vss

These two pins are used to supply power and ground to the chip. The voltage supplied will be +5 volts ±5%.

• XTAL and EXTAL

These connections are for a parallel resonant fundamental crystal, AT cut. Divide by 4 circuitry is included with the internal clock, so a 4 MHz crystal may be used to run the system at 1 MHz. The divide by 4 circuitry allows for use of the inexpensive 3.58 MHz Color TV crystal for non-time critical applications. Two 22pF capacitors are needed from the two crystal pins to ground to insure reliable operation. EXTAL may be driven by an external TTL compatible clock source with a 50% (±10%) duty cycle. It will divide by 4 any frequency less than or equal to 5 MHz. XTAL must be grounded if an external clock is used. The following are the recommended crystal parameters:

Nominal Crystal Parameter

Item \ Crystal	4 MHz	5 MHz
Co	7 pF max.	4.7 pF max.
R_S	60Ω max.	30Ω typ.

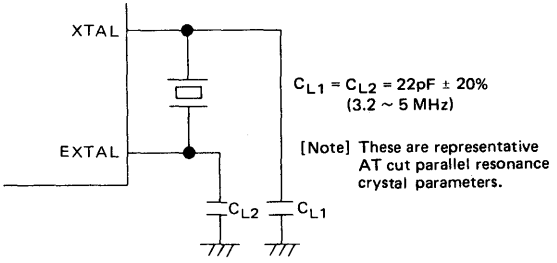


Figure 12 Crystal Interface

• Vcc Standby

This pin will supply +5 volts ±5% to the standby RAM on the chip. The first 64 bytes of RAM will be maintained in the power down mode with 8 mA current max. The circuit of figure 13 can be utilized to assure that V_{CC} Standby does not go below V_{SBB} during power down.

To retain information in the RAM during power down the following procedure is necessary:

1) Write "0" into the RAM enable bit, RAM E. RAM E is bit 6 of the RAM Control Register at location $0014. This disables the standby RAM, thereby protecting it at power down.
2) Keep V_{CC} Standby greater than V_{SBB}.

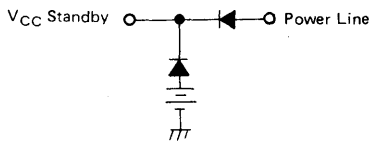


Figure 13 Battery Backup for V_{CC} Standby

• Reset ($\overline{RES}$)

This input is used to reset and start the MPU from a power down condition, resulting from a power failure or an initial startup of the processor. On power up, the reset must be held "Low" for at least 100 ms. During operation, $\overline{RES}$, when brought "Low" must be held "Low" at least 3 clock cycles.

When a "High" level is detected, the MPU does the following:

1) All the higher order address lines will be forced "High".
2) I/O Port 2 bits, 2, 1, and 0 are latched into programmed control bits PC2, PC1 and PC0.
3) The last two ($FFFE, $FFFF) locations in memory will be used to load the program addressed by the program counter.
4) The interrupt mask bit is set, must be cleared before the MPU can recognize maskable interrupts.

• Enable (E)

This supplies the external clock for the rest of the system when the internal oscillator is used. It is a single phase, TTL compatible clock, and will be the divide by 4 result of the crystal frequency. It will drive one TTL load and 90 pF.

• Non-Maskable Interrupt ($\overline{NMI}$)

A low-going edge on this input requests that a non-maskable-interrupt sequence be generated within the processor. As with interrupt Request signal, the processor will complete the current instruction that is being executed before it recognizes the $\overline{NMI}$ signal. The interrupt mask bit in the Condition Code Register has no effect on $\overline{NMI}$.

In response to an $\overline{NMI}$ interrupt, the Index Register, Program Counter, Accumulators, and Condition Code Register are stored on the stack. At the end of the sequence, a 16-bit address will be loaded that points to a vectoring address located in memory locations $FFFC and $FFFD. An address loaded at these locations causes the MPU to branch to a non-maskable interrupt service routine in memory.

A 3.3 kΩ external resistor to Vcc should be used for wire-OR and optimum control of interrupts.

Inputs $\overline{IRQ_1}$ and $\overline{NMI}$ are hardware interrupt lines that are sampled during E and will start the interrupt routine on the $\overline{E}$ following the completion of an instruction.

• Interrupt Request ($\overline{IRQ_1}$)

This level sensitive input requests that an interrupt sequence be generated within the machine. The processor will wait until it completes the current instruction that it being executed before it recognizes the request. At that time, if the interrupt mask bit in the Condition Code Register is not set, the machine will begin an interrupt sequence. The Index Register, Program Counter, Accumulators, and Condition Code Register are stored on the stack. Next the MPU will respond to the interrupt request by setting the interrupt mask bit "High" so that no further maskable interrupts may occur. At the end of the cycle, a 16-bit address will be loaded that points to a vectoring address which is located in memory locations $FFF8 and $FFF9. An address loaded at these locations causes the MPU to branch to an interrupt routine in memory.

The $\overline{IRQ_1}$ requires a 3.3 kΩ external resister to V_{CC} which should be used for wire-OR and optimum control of interrupts. Internal Interrupts will use an internal interrupt line ($\overline{IRQ_2}$). This interrupt will operate the same as $\overline{IRQ_1}$ except that it will use the vector address of $FFF0 through $FFF7. $\overline{IRQ_1}$ will have priority over $\overline{IRQ_2}$ if both occur at the same time. The Interrupt Mask Bit in the condition code register masks both interrupts (See Table 1).

Table 1 Interrupt Vector Location

	Vector		Interrupt
	MSB	LSB	
Highest Priority	FFFE	FFFF	$\overline{RES}$
	FFFC	FFFD	$\overline{NMI}$
	FFFA	FFFB	Software Interrupt (SWI)
	FFF8	FFF9	$\overline{IRQ_1}$ (or $\overline{IS3}$)
	FFF6	FFF7	ICF (Input Capture)
	FFF4	FFF5	OCF (Output Compare)
	FFF2	FFF3	TOF (Timer Overflow)
Lowest Priority	FFF0	FFF1	SC_1 (RDRF + ORFE + TDRE)

The following pins are available in the Single Chip Mode, and are associated with Port 3 only.

- **Input Strobe ($\overline{IS3}$) (SC_1)**

The function of the $\overline{IS3}$ signal depends on the I/O Port 3 Control/Status Register. If $\overline{IS3}$ Enable bit is set, an interrupt will occur by the fall of the $\overline{IS3}$ signal. If the latch enable bit is set, the data in the I/O Port 3 will be latched at the I/O Port 3 Data Register. The timing condition of the $\overline{IS3}$ signal that is necessary to be latched the input data normally is shown in Figure 6.

- **Output Strobe ($\overline{OS3}$) (SC_2)**

This signal is used by the processor to strobe an external device, indicating valid data is on the I/O pins. The timing for the Output Strobe is shown in Figure 5 I/O Port 3 Control/Status Register is discussed in the following section.

The following pins are available in the Expanded Modes.

- **Read/Write (R/$\overline{W}$) (SC_2)**

This TTL compatible output signals the peripherals and memory devices whether the MPU is in a Read ("High") or a Write ("Low") state. The normal standby state of this signal is Read ("High"). This output is capable of driving one TTL load and 90 pF.

- **I/O Strobe ($\overline{IOS}$) (SC_1)**

In the expanded non-multiplexed mode of operation, $\overline{IOS}$ internally decodes A_9 through A_{15} as zero's and A_8 as a one. This allows external access of the 256 locations from $0100 to $01FF. The timing diagrams are shown as figure 2.

- **Address Strobe (AS) (SC_1)**

In the expanded multiplexed mode of operation address strobe is output on this pin. This signal is used to latch the 8 LSB's of address which are multiplexed with data on Port 3. An 8-bit latch is utilized in conjunction with Address Strobe, as shown in figure 19. Expanded Multiplexed Mode. Address Strobe signals the latch when it is time to latch the address lines so the lines can become data bus lines during the E pulse. The timing for this singal is shown in Figure 1 of Bus Timing. This signal is also used to disable the address from the multiplexed bus allowing a deselect time, t_{ASD} before the data is enabled to the bus.

- **PORTS**

There are four I/O ports on the HD6801V MCU; three 8-bit ports and one 5-bit port. There are two control lines associated with one of the 8-bit ports. Each port has an associated write only Data Direction Register which allows each I/O line to be programmed to act as an input or an output*. A "1" in the corresponding Data Direction Register bit will cause that I/O line to be an output. A "0" in the corresponding Data Direction Register bit will cause that I/O line to be an input. There are four ports: Port 1, Port 2, Port 3, and Port 4. Their addresses and the addresses of their Data Direction registers are given in Table 2.

* The only exception is bit 1 of Port 2, which can either be data input or Timer output.

Table 2 Port and Data Direction Register Addresses

Ports	Port Address	Data Direction Register Address
I/O Port 1	$0002	$0000
I/O Port 2	$0003	$0001
I/O Port 3	$0006	$0004
I/O Port 4	$0007	$0005

- **I/O Port 1**

This is an 8-bit port whose individual bits may be defined as inputs or outputs by the corresponding bit in its data direction register. The 8 output buffers have three-state capability, allowing them to enter a high impedance state when the peripheral data lines are used as inputs. In order to be read properly, the voltage on the input lines must be greater than 2.0 V for a logic "1" and less than 0.8 V for a logic "0". As outputs these lines are TTL compatible and may also be used as a source of up to 1 mA at 1.5 V to directly drive a Darlington base. After Reset, the I/O lines are configured as inputs. In all three modes, Port 1 is always parallel I/O.

- **I/O Port 2**

This port has five lines that may be defined as inputs or outputs by its data direction register. The 5 output buffers have three-state capability, allowing them to enter a high impedance state when used as an input. In order to be read properly, the voltage on the input lines must be greater than 2.0 V for a logic "1" and less than 0.8 V for a logic "0". As outputs, this port has no internal pullup resistors but will drive TTL inputs directly. For driving CMOS inputs, external pullup resistors are required. After Reset, the I/O lines are configured as inputs. Three pins on Port 2 (pins 10, 9, and 8 of the chip) are used to program the mode of operation during reset. The values of these pins at reset are latched into the three MSB's (bits 7, 6, and 5) of Port 2 which are read only. This is explained in the Mode Selection Section.

In all three modes, Port 2 can be configured as I/O and provides access to the Serial Communications Interface and the Timer. Bit 1 is the only pin restricted to data input or Timer output.

- **I/O Port 3**

This is an 8-bit port that can be configured as I/O, a data bus, or an address bus multiplexed with the data bus – depending on the mode of operation hardware programmed by the user at reset. As a data bus, Port 3 is bi-directional. As an input for peripherals, it must be supplied regular TTL levels, that is, greater than 2.0 V for a logic "1" and less than 0.8 V for a logic "0"

Its TTL compatible three-state output buffers are capable of driving one TTL load and 90 pF. In the Expanded Modes, after reset, the data direction register is inhibited and data flow depends on the state of the R/$\overline{W}$ line. The input strobe ($\overline{IS3}$) and the output strobe ($\overline{OS3}$) used for handshaking are explained later.

In the three modes, Port 3 assumes the following characteristics:

Single Chip Mode: Parallel Inputs/Outputs as programmed by its associated Data Direction Register. There are two control lines associated with this port in this mode, an input strobe and an output strobe, that can be used for handshaking. They are controlled by the I/O Port 3 Control/Status Register explained at the end of this section. Three options of Port 3 operations are sumarized as follows: (1) Port 3 input data can be latched using $\overline{IS3}$ (SC_1) as a control signal, (2) $\overline{OS3}$ can be generated by either an MPU read or write to Port 3's Data Register, and (3) and $\overline{IRQ_1}$ interrupt can be enabled by an $\overline{IS3}$ negative edge. Port 3 latch and strobe timing is shown in Fig. 5 and Fig. 6.

Expanded Non-Multiplexed Mode: In this mode, Port 3 becomes the data bus (D_0~D_7).

Expanded Multiplexed Mode: In this mode, Port 3 becomes both the data bus (D_0~D_7) and lower bits of the address bus (A_0~A_7). An address strobe output is true when the address is on the port.

I/O PORT 3 CONTROL/STATUS REGISTER

	7	6	5	4	3	2	1	0
	$\overline{IS3}$	$\overline{IS3}$ $\overline{IRQ_1}$	X	OSS	LATCH	X	X	X
$000F	FLAG	ENABLE			ENABLE			

Bit 0; Not used.
Bit 1; Not used.
Bit 2; Not used.
Bit 3; LATCH ENABLE. This controls the input latch for I/O Port 3. If this bit is set "High" the input data will be latched with the falling edge of the Input Strobe, $\overline{IS3}$. This bit is cleared by reset, and the latch is "re-opened" with MCU read Port 3.
Bit 4; OSS. (Output Strobe Select) This bit will select if the Output Strobe should be generated at $\overline{OS3}$ (SC_2) by a write to I/O Port 3 or a read of I/O Port 3. When this bit is cleared the strobe is generated by a read Port 3. When this bit is set the strobe is generated by a write Port 3.
Bit 5; Not used.
Bit 6; $\overline{IS3}$ $\overline{IRQ_1}$ ENABLE. When set, interrupt will be enabled whenever $\overline{IS3}$ FLAG is set; when clear, interrupt is inhibited. This bit is cleared by $\overline{RES}$.
Bit 7; $\overline{IS3}$ FLAG. This is a read only status bit that is set by the falling edge of the input strobe, $\overline{IS3}$ (SC_1). It is cleared by a read of the Control/Status Register followed by a read or write of I/O Port 3. Reset will clear this bit.

• I/O Port 4

This is an 8-bit port that can be configured as I/O or as address lines depending on the mode of operation. In order to be read properly, the voltage on the input lines must be greater than 2.0 V for a logic "1" and less than 0.8 V for a logic "0".

As outputs, each line is TTL compatible and can drive 1 TTL load and 90 pF. After reset, the lines are configured as inputs. To use the pins as addresses, therefore, they should be programmed as outputs. In the three modes, Port 4 assumes the following characteristics:

Single Chip Mode: Parallel Inputs/Outputs as programmed by its associated Data Direction Register.

Expanded Non-Multiplexed Mode: In this mode, Port 4 is configured as the lower order address lines (A_0~A_7) by writing one's to the data direction register. When all eight address lines are not needed, the remaining lines, starting with the most significant bit, may be used as I/O (inputs only).

Expanded Multiplexed Mode: In this mode, Port 4 is configured as the higher order address lines (A_8~A_{15}) by writing one's to the data direction register. When all eight address lines are not needed, the remaining lines, starting with the most significant bit, may be used as I/O (inputs only).

■ OPERATION MODES

The mode of operation that HD6801V will operate in after Reset is determined by hardware that the user must wire on pins 10, 9, and 8 of the chip. These pins are the three LSB's (I/O 2, I/O 1, and I/O 0 respectively) of Port 2. They are latched into programmed control bits PC2, PC1, and PC0 when reset goes high. I/O Port 2 Register is shown below.

PORT 2 DATA REGISTER

	7	6	5	4	3	2	1	0
$0003	PC2	PC1	PC0	I/O 4	I/O 3	I/O 2	I/O 1	I/O 0

An example of external hardware that could be used for Mode Selection is shown in Fig 14. The HD14053B provides the isolation between the peripheral device and MCU during Reset, which is necessary if data conflict can occur between peripheral device and Mode generation circuit.

As bits 5, 6 and 7 of Port 2 are read only, the mode cannot be changed through software. The mode selections are shown in Table 3.

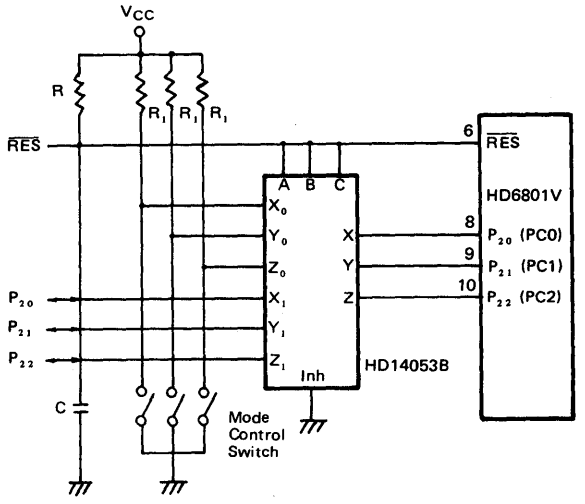
The HD6801V is capable of operating in three basic modes; (1) Single Chip Mode, (2) Expanded Multiplexed Mode (compatible with HMCS6800 peripheral family) (3) Expanded Non-Multiplexed Mode.

• Single Chip Mode

In the Single Chip Mode the Ports are configured for I/O. This is shown in Figure 16 the single Chip Mode. In this mode, Port 3 will have two associated control lines, an input strobe and an output strobe for handshaking data.

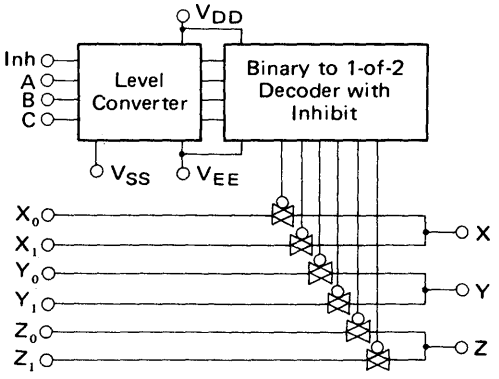
• Expanded Non-Multiplexed Mode

In this mode the HD6801V will directly address HMCS6800 peripherals with no external logic. In this mode Port 3 becomes the data bus. Port 4 becomes the A_0~A_7 address bus or partial address and I/O (inputs only). Port 2 can be parallel I/O, serial I/O, Timer, or any combination of them. Port 1 is parallel I/O only. In this mode the HD6801V is expandable to 256 locations. The eight address lines associated with Port 4 may be substituted for I/O (inputs only) if a fewer number of address lines will satisfy the application (See Figure 17).



[NOTES] 1) Mode 7 as shown
2) RC≈Reset time constant
3) R_1=10kΩ

Figure 14 Recommended Circuit for Mode Selection



Truth Table

Control Input				On Switch		
Inhibit	Select					
	C	B	A	HD14053B		
0	0	0	0	Z_0	Y_0	X_0
0	0	0	1	Z_0	Y_0	X_1
0	0	1	0	Z_0	Y_1	X_0
0	0	1	1	Z_0	Y_1	X_1
0	1	0	0	Z_1	Y_0	X_0
0	1	0	1	Z_1	Y_0	X_1
0	1	1	0	Z_1	Y_1	X_0
0	1	1	1	Z_1	Y_1	X_1
1	X	X	X	–		

Figure 15 HD14053B Multiplexers/Demultiplexers

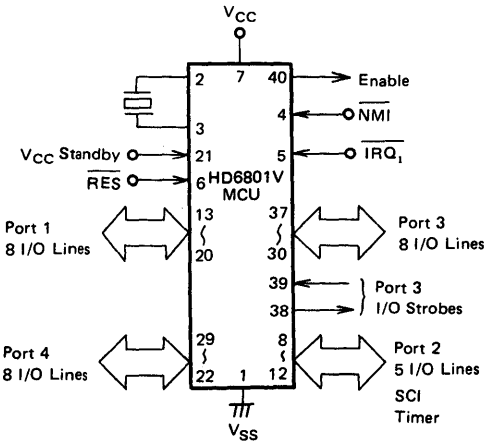


Figure 16 HD6801V MCU Single-Chip Mode

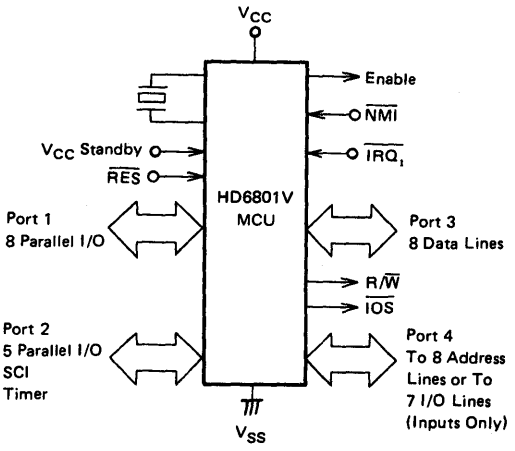


Figure 17 HD6801V MCU Expanded Non-Multiplexed Mode

- **Expanded Multiplexed Mode**

In this mode Port 4 becomes higher order address lines with an alternative of substituting some of the address lines for I/O (inputs only). Port 3 is the data bus multiplexed with the lower order address lines differentiated by an output called Address Strobe. Port 2 is 5 lines of Parallel I/O, SCI, Timer, or any combination of them. Port 1 is 8 Parallel I/O lines. In this mode it is expandable to 65k words. (See Figure 18).

- **Lower order Address Bus Latches**

Since the data bus is multiplexed with the lower order address bus in Port 3, latches are required to latch those address bits. The 74LS373 Transparent octal D-type latch can be used with the HD6801V to latch the least significant address byte. Figure 19 shows how to connect the latch to the HD6801V. The output control to the 74LS373 may be connected to ground.

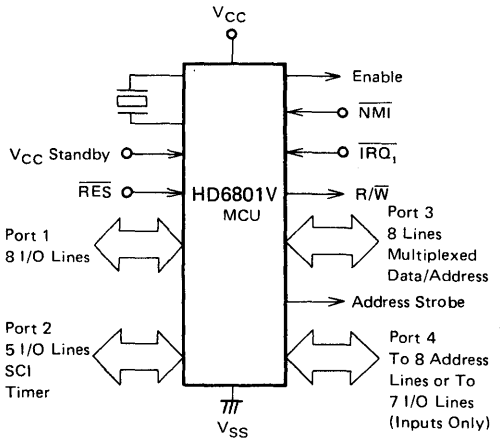
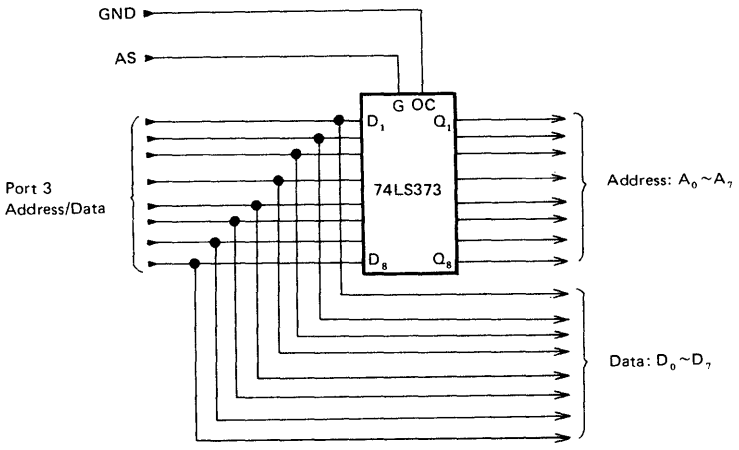


Figure 18 HD6801V MCU Expanded Multiplexed Mode



Function Table

Output Control	Enable G	Enable D	Output Q
L	H	H	H
L	H	L	L
L	L	X	Q_0
H	X	X	Z

Figure 19 Latch Connection

- **Mode and Port Summary MCU Signal Description**

This section gives a description of the MCU signals for the various modes. SC_1 and SC_2 are signals which vary with the mode that the chip is in.

MODE	PORT 1 Eight Lines	PORT 2 Five Lines	PORT 3 Eight Lines	PORT 4 Eight Lines	SC_1	SC_2
SINGLE CHIP	I/O	I/O	I/O	I/O	$\overline{IS3}$ (I)	$\overline{OS3}$ (O)
EXPANDED MUX	I/O	I/O	ADDRESS BUS (A_0~A_7) DATA BUS (D_0~D_7)	ADDRESS BUS* (A_8~A_{15})	AS(O)	R/$\overline{W}$(O)
EXPANDED NON-MUX	I/O	I/O	DATA BUS (D_0~D_7)	ADDRESS BUS* (A_0~A_7)	$\overline{IOS}$(O)	R/$\overline{W}$(O)

*These lines can be substituted for I/O (Input Only) starting with the most significant address line.

I = Input
O = Output
R/$\overline{W}$ = Read/Write
$\overline{IS3}$ = Input Strobe
$\overline{OS3}$ = Output Strobe
$\overline{IOS}$ = I/O Select
SC = Strobe Control
AS = Address Strobe

Table 3 Mode Selection Summary

Mode	$P_{2.2}$ (PC2)	$P_{2.1}$ (PC1)	$P_{2.0}$ (PC0)	ROM	RAM	Interrupt Vectors	Bus Mode	Operating Mode
7	H	H	H	I	I	I	I	Single Chip
6	H	H	L	I	I	I	MUX(6)	Multiplexed/Partial Decode
5	H	L	H	I	I	I	NMUX(6)	Non-Multiplexed/Partial Decode
4	H	L	L	I(2)	I(1)	I	I	Single Chip Test
3	L	H	H	E	E	E	MUX	Multiplexed/No RAM & ROM
2	L	H	L	E	I	E	MUX	Multiplexed/RAM
1	L	L	H	I	I	E	MUX	Multiplexed/RAM & ROM
0	L	L	L	I	I	I(3)	MUX	Multiplexed Test

LEGEND:
I – Internal
E – External
MUX – Multiplexed
NMUX – Non-Multiplexed
L – Logic "0"
H – Logic "1"

[NOTES]
1) Internal RAM is addressed at $XX80
2) Internal ROM is disabled
3) $\overline{RES}$ vector is external for 2 cycles after $\overline{RES}$ goes "High"
4) Addresses associated with Ports 3 and 4 are considered external in Modes 0, 1, 2, and 3
5) Addresses associated with Port 3 are considered external in Modes 5 and 6
6) Port 4 default is user data input; address output is optional by writing to Port 4 Data Direction Register

■ MEMORY MAPS

The MCU can provide up to 65k byte address space depending on the operating mode. A memory map for each operating mode is shown in Figure 20. The first 32 locations of each map are reserved for the MCU's internal register area, as shown in Table 4. With exceptions as indicated.

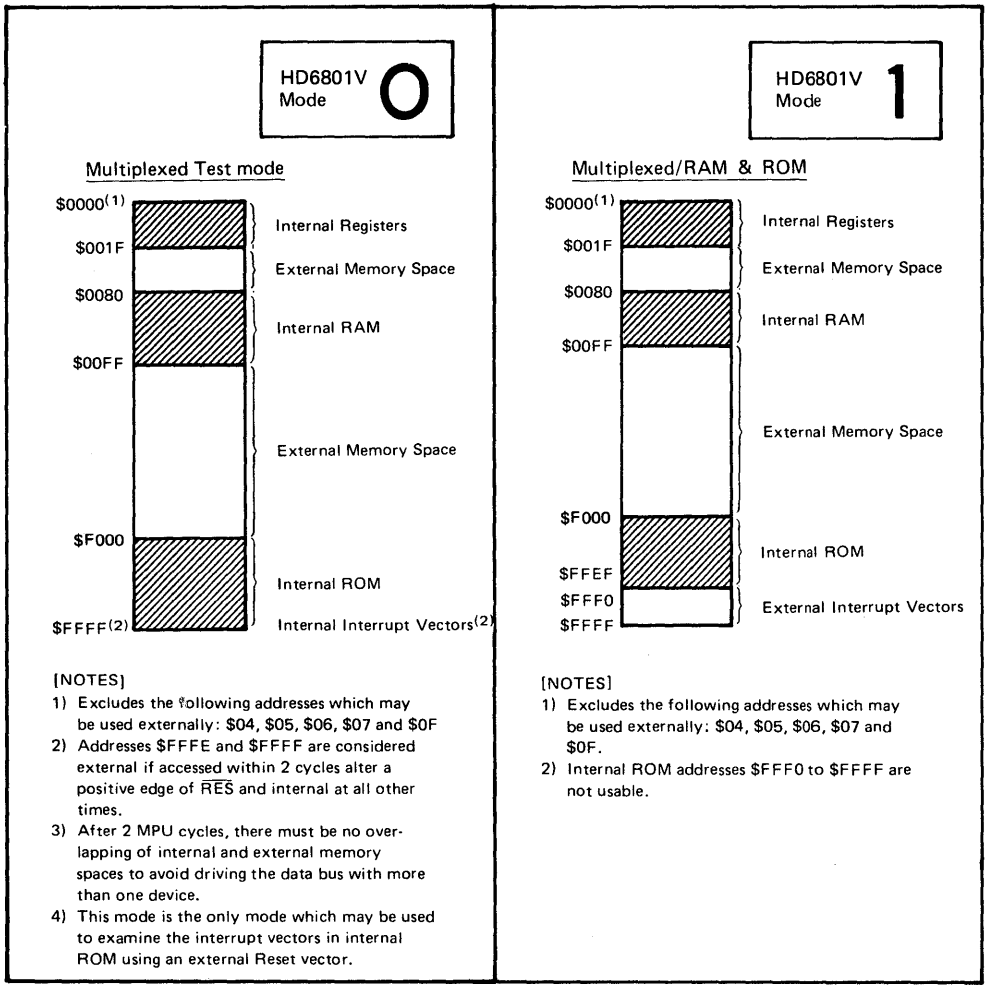
Table 4 Internal Register Area

Register	Address
Port 1 Data Direction Register***	00
Port 2 Data Direction Register***	01
Port 1 Data Register	02
Port 2 Data Register	03
Port 3 Data Direction Register***	04*
Port 4 Data Direction Register***	05**
Port 3 Data Register	06*
Port 4 Data Register	07**
Timer Control and Status Register	08
Counter (High Byte)	09
Counter (Low Byte)	0A
Output Compare Register (High Byte)	0B
Output Compare Register (Low Byte)	0C
Input Capture Register (High Byte)	0D
Input Capture Register (Low Byte)	0E
Port 3 Control and Status Register	0F*
Rate and Mode Control Register	10
Transmit/Receive Control and Status Register	11
Receive Data Register	12
Transmit Data Register	13
RAM Control Register	14
Reserved	15-1F

* External address in Modes 0, 1, 2, 3, 5, 6; cannot be accessed in Mode 5 (No. $\overline{IOS}$)
** External addresses in Modes 0, 1, 2, 3
*** 1=Output, 0=Input

■ INTERRUPT FLOWCHART

The Interrupt flowchart is depicted in Figure 24 and is common to every interrupt excluding reset.



[NOTES]
1) Excludes the following addresses which may be used externally: $04, $05, $06, $07 and $0F
2) Addresses $FFFE and $FFFF are considered external if accessed within 2 cycles alter a positive edge of $\overline{RES}$ and internal at all other times.
3) After 2 MPU cycles, there must be no overlapping of internal and external memory spaces to avoid driving the data bus with more than one device.
4) This mode is the only mode which may be used to examine the interrupt vectors in internal ROM using an external Reset vector.

[NOTES]
1) Excludes the following addresses which may be used externally: $04, $05, $06, $07 and $0F.
2) Internal ROM addresses $FFF0 to $FFFF are not usable.

Figure 20 HD6801V Memory Maps

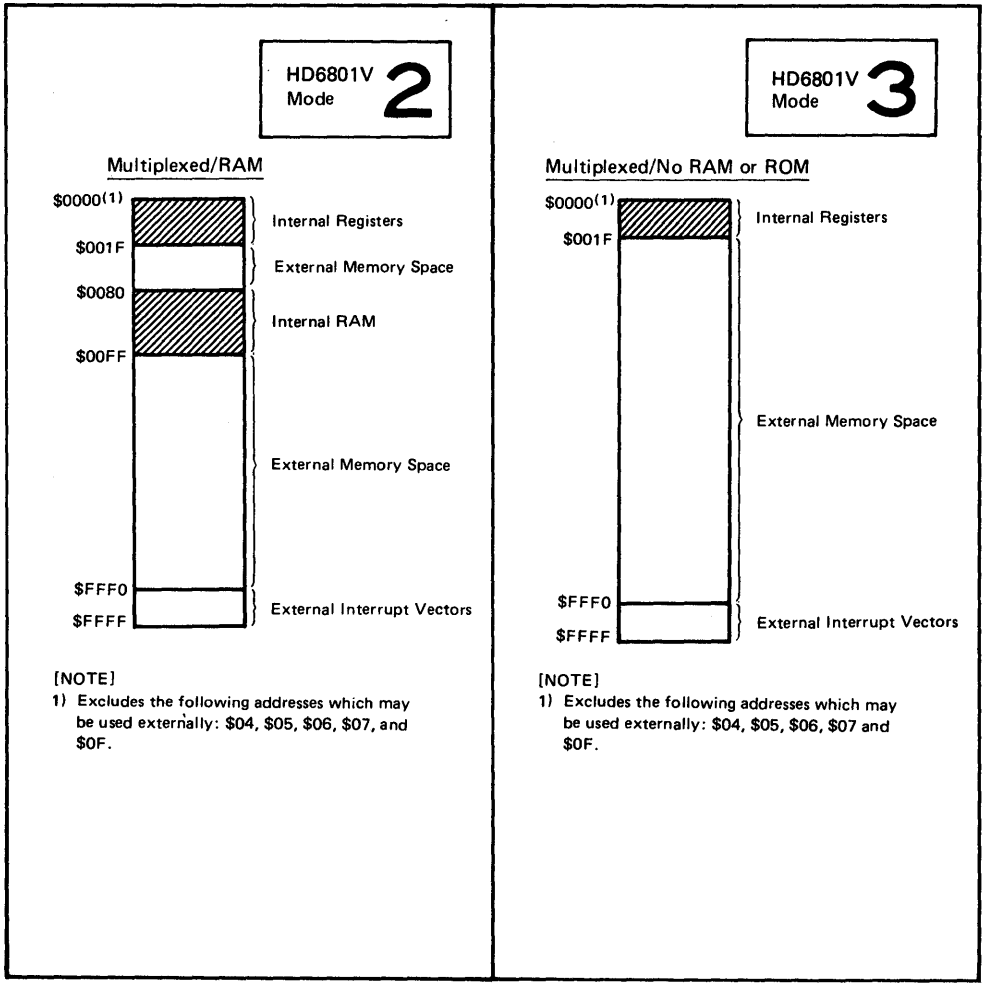


Figure 20 HD6801V Memory Maps (Continued)

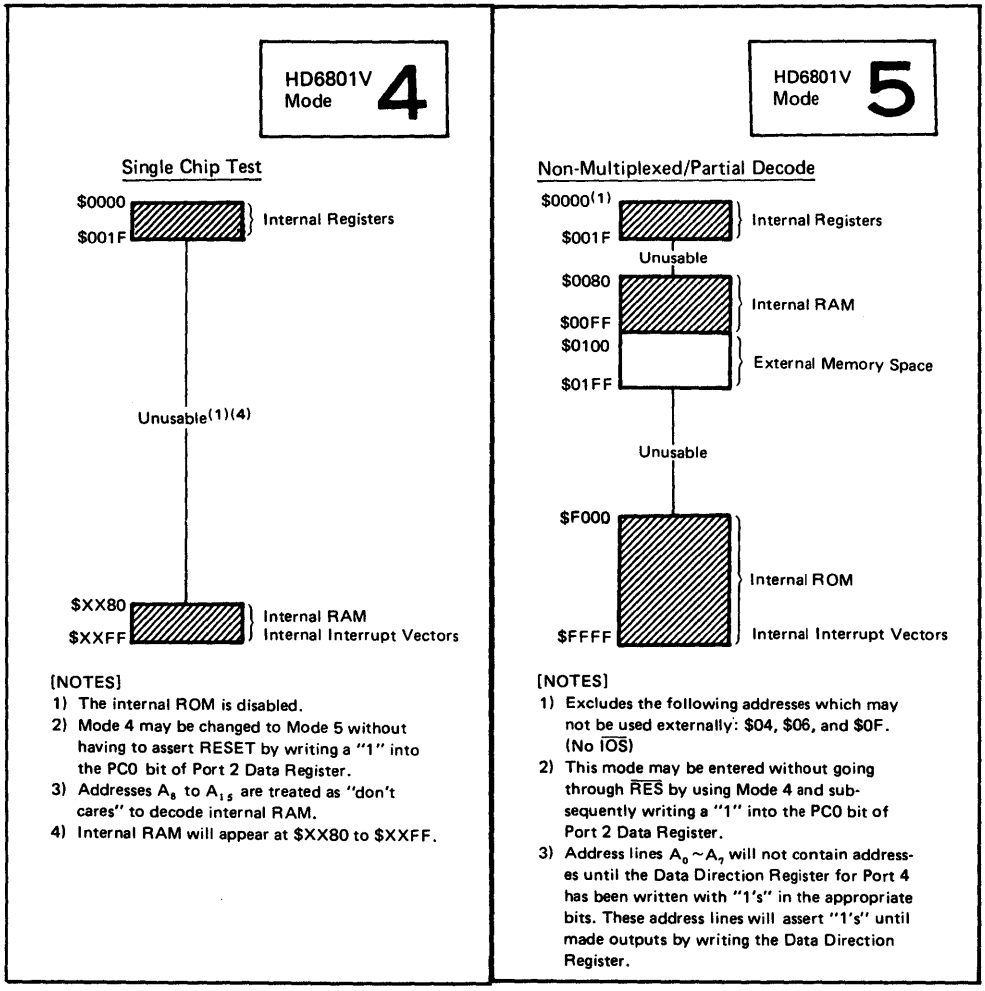
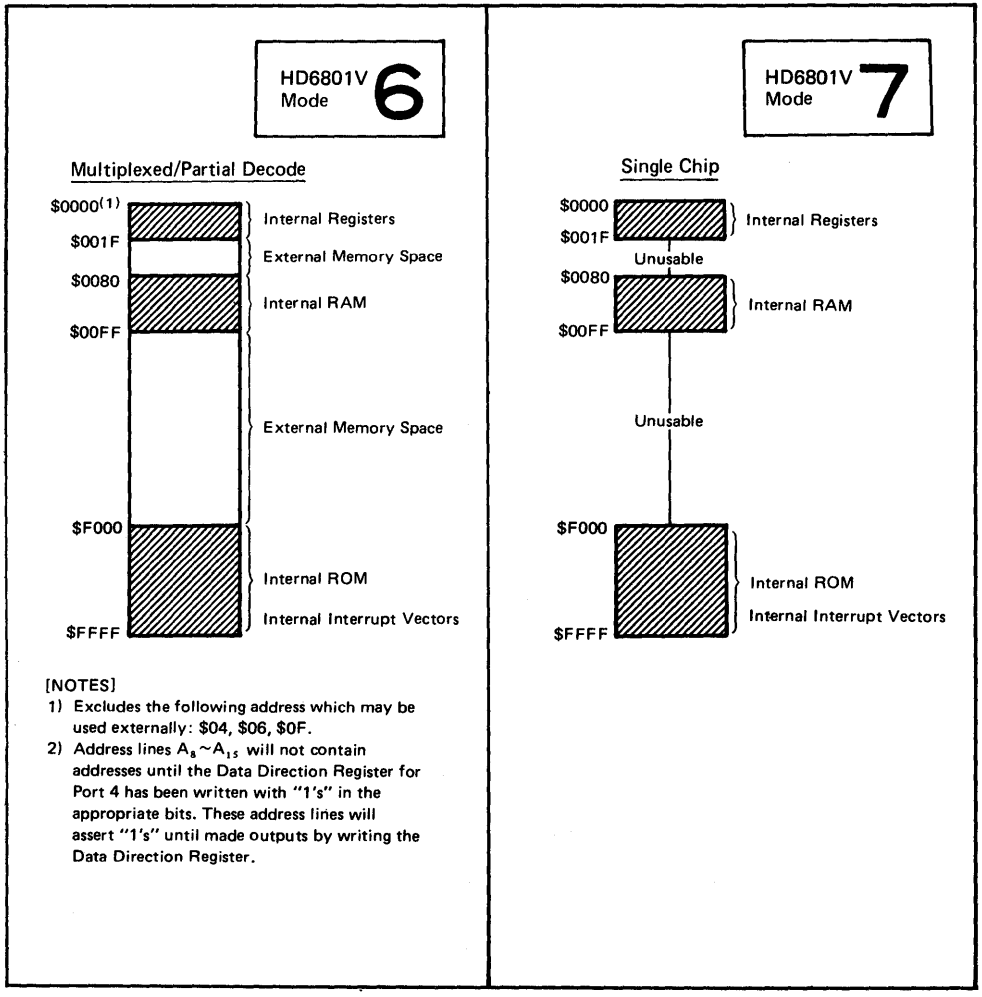


Figure 20 HD6801V Memory Maps (Continued)



[NOTES]
1) Excludes the following address which may be used externally: $04, $06, $0F.
2) Address lines A_8~A_{15} will not contain addresses until the Data Direction Register for Port 4 has been written with "1's" in the appropriate bits. These address lines will assert "1's" until made outputs by writing the Data Direction Register.

Figure 20 HD6801V Memory Maps (Continued)

■ PROGRAMMABLE TIMER

The HD6801V contains an on-chip 16-bit programmable timer which may be used to perform measurements on an input waveform while independently generating an output waveform. Pulse widths for both input and output signals may vary from a few microseconds to many seconds. The timer hardware consists of

- an 8-bit control and status register,
- a 16-bit free running counter,
- a 16-bit output compare register, and
- a 16-bit input capture register

A block diagram of the timer registers is shown in Figure 21.

● Free Running Counter ($0009:000A)

The key element in the programmable timer is a 16-bit free running counter which is driven to increasing values by E (Enable). The counter value may be read by the MPU software at any time. The counter is cleared to zero on $\overline{RES}$ and may be considered a read-only register with one exception. Any MPU write to the counter's address ($09) will always result in preset value of $FFF8 being loaded into the counter regardless of the value involved in the write. This preset figure is intended for testing operation of the part, but may be of value in some applications.

● Output Compare Register ($000B:000C)

The Output Compare Register is a 16-bit read/write register which is used to control an output waveform. The contents of this register are constantly compared with the current value of the free running counter. When a match is found, a flag is set (OCF) in the Timer Control and Status Register (TCSR) and the current value of the Output Level bit (OLVL) in the TCSR is clocked to the Output Level Register. Providing the Data Direction Register for Port 2, Bit 1 contains a "1" (Output), the output level register value will appear on the pin for Port 2 Bit 1. The values in the Output Compare Register and Output level bit may then be changed to control the output level on the next compare value. The Output Compare Register is set to $FFFF during $\overline{RES}$. The Compare function is inhibited for one cycle following a write to the high byte of the Output Compare Register to insure a valid 16-bit value is in the register before a compare is made.

● Input Capture Register ($000D:000E)

The Input Capture Register is a 16-bit read-only register used to store the current value of the free running counter when the proper transition of an external input signal occurs. The input transition change required to trigger the counter transfer is controlled by the input Edge bit (IEDG) in the TCSR. The Data Direction Register bit for Port 2 Bit 0, should* be clear (zero) in order to gate in the external input signal to the edge detect unit in the timer.

* With Port 2 Bit 0 configured as an output and set to "1", the external input will still be seen by the edge detect unit.

● Timer Control and Status Register (TCSR) ($0008)

The Timer Control and Status Register consists of an 8-bit register of which all 8 bits are readable but only the low order 5 bits may be written. The upper three bits contain read-only timer status information and indicate that:

- a proper transition has taken place on the input pin with a subsequent transfer of the current counter value to the input capture register.
- a match has been found between the value in the free running counter and the output compare register, and
- when $0000 is in the free running counter.

Each of the flags may be enabled onto the HD6801V internal bus ($\overline{IRQ_2}$) with an individual Enable bit in the TCSR. If the

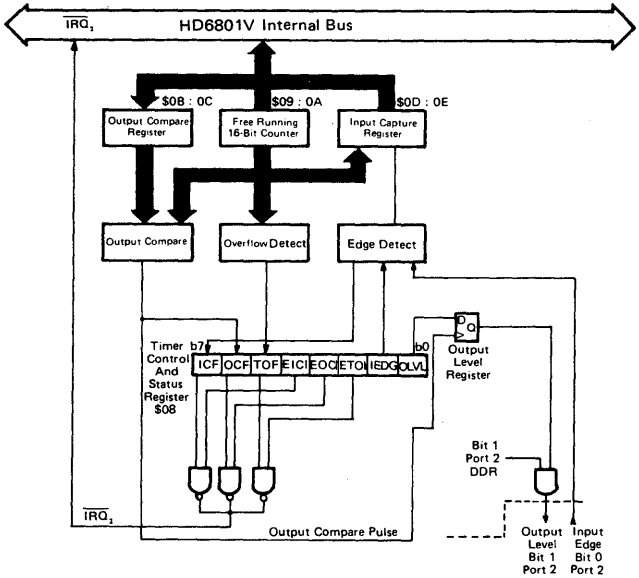


Figure 21 Block Diagram of Programmable Timer

Timer Control and Status Register

7	6	5	4	3	2	1	0	
ICF	OCF	TOF	EICI	EOCI	ETOI	IEDG	OLVL	$0008

I-bit in the HD6801V Condition Code register has been cleared, a priority vectored interrupt will occur corresponding to the flag bit(s) set. A description for each bit follows:

Bit 0 **OLVL** Output Level – This value is clocked to the output level register on a successful output compare. If the DDR for Port 2 bit 1 is set, the value will appear on the output pin.

Bit 1 **IEDG** Input Edge – This bit controls which transition of an input will trigger a transfer of the counter to the input capture register. The DDR for Port 2 Bit 0 must be clear for this function to operate. IEDG = 0 Transfer takes place on a negative edge ("High"-to-"Low" transition).
IEDG = 1 Transfer takes place on a positive edge ("Low"-to-"High" transition).

Bit 2 **ETOI** Enable Timer Overflow Interrupt – When set, this bit enables $\overline{IRQ_2}$ to occur on the internal bus for a TOF interrupt; when clear the interrupt is inhibited.

Bit 3 **EOCI** Enable Output Compare Interrupt – When set, this bit enables $\overline{IRQ_2}$ to appear on the internal bus for an output compare interrupt; when clear the interrupt is inhibited.

Bit 4 **EICI** Enable input Capture Interrupt – When set, this bit enables $\overline{IRQ_2}$ to occur on the internal bus for an input capture interrupt; when clear the interrupt is inhibited.

Bit 5 **TOF** Timer Overflow Flag – This read-only bit is set when the counter contains $0000. It is cleared by a read of the TCSR (with TOF set) followed by an MPU read of the Counter ($09).

Bit 6 **OCF** Output Compare Flag – This read-only bit is set when a match is found between the output compare register and the free running counter. It is cleared by a read of the TCSR (with OCF set) followed by an MPU write to the output compare register ($0B or $0C).

Bit 7 **ICF** Input Capture Flag – This read-only status bit is set by a proper transition on the input; it is cleared by a read of the TCSR (with ICF set) followed by an MPU read of the Input Capture Register ($0D).

■ SERIAL COMMUNICATIONS INTERFACE

The HD6801V contains a full-duplex asynchronous serial communications interface (SCI) on chip. The controller comprises a transmitter and a receiver which operate independently or each other but in the same data format and at the same data rate. Both transmitter and receiver communicate with the MPU via the data bus and with the outside world via pins 2, 3, and 4 of Port 2. The hardware, software, and registers are explained in the following paragraphs.

• Wake-Up Feature

In a typical multi-processor application, the software protocol will usually contain a destination address in the initial byte(s) of the message. In order to permit non-selected MPU's to ignore the remainder of the message, a wake-up feature is included whereby all further interrupt processing may be optionally inhibited until the beginning of the next message. When the next message appears, the hardware re-enables (or "wakes-up") for the next message. The "wake-up" is automatically triggered by a string of ten consecutive 1's which indicates an idle transmit line. The software protocol must provide for the short idle period between any two consecutive messages.

• Programmable Options

The following features of the HD6801V serial I/O section are programmable:

- format – standard mark/space (NRZ)
- Clock – external or internal
- baud rate – one of 4 per given MPU ϕ_2 clock frequency or external clock ×8 input
- wake-up feature – enabled or disabled
- Interrupt requests – enabled or masked individually for transmitter and receiver data registers
- clock output – internal clock enabled or disabled to Port 2 (Bit 2)
- Port 2 (bits 3 and 4) – dedicated or not dedicated to serial I/O individually for transmitter and receiver.

• Serial Communications Hardware

The serial communications hardware is controlled by 4 registers as shown in Figure 22. The registers include:

- an 8-bit control and status register
- a 4-bit rate and mode control register (write only)
- an 8-bit read only receive data register and
- an 8-bit write only transmit data register.

In addition to the four registers, the serial I/O section utilizes bit 3 (serial input) and bit 4 (serial output) of Port 2. Bit 2 of Port 2 is utilized if the internal-clock-out or external-clock-in options are selected.

Transmit/Receive Control and Status (TRCS) Register

The TRCS register consists of an 8-bit register of which all 8 bits may be read while only bits 0~4 may be written. The register is initialized to $20 on $\overline{RES}$. The bits in the TRCS register are defined as follows:

Transmit/Receive Control and Status Register

7	6	5	4	3	2	1	0	
RDRF	ORFE	TDRE	RIE	RE	TIE	TE	WU	ADDR : $0011

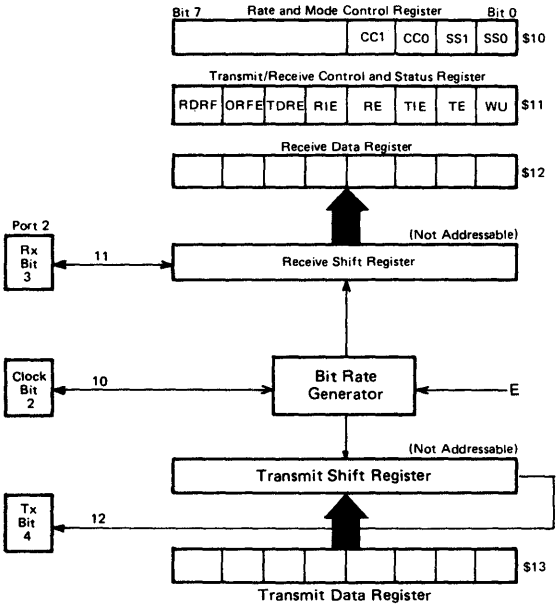


Figure 22 Serial I/O Registers

Bit 0 **WU** "Wake-up" on Next Message – set by HD6801V software and cleared by hardware on receipt of ten consecutive 1's or reset of RE flag. It should be noted that RE flag should be set in advance of MPU set of WU flag.

Bit 1 **TE** Transmit Enable – set by HD6801V to produce preamble of nine consecutive 1's and to enable gating of transmitter output to Port 2, bit 4 regardless of the DDR value corresponding to this bit; when clear, serial I/O has no effect on Port 2 bit 4.

TE set should be after at least one bit time of data transmit rate from the set-up of transmit data rate and mode.

Bit 2 **TIE** Transmit Interrupt Enable – when set, will permit an $\overline{IRQ_2}$ interrupt to occur when bit 5 (TDRE) is set; when clear, the TDRE value is masked from the bus.

Bit 3 **RE** Receiver Enable – when set, gates Port 2 bit 3 to input of receiver regardless of DDR value for this bit; when clear, serial I/O has no effect on Port 2 bit 3.

Bit 4 **RIE** Receiver Interrupt Enable – when set, will permit an $\overline{IRQ_2}$ interrupt to occur when bit 7 (RDRF) or bit 6 (ORFE) is set; when clear, the interrupt is masked.

Bit 5 **TDRE** Transmit Data Register Empty – set by hardware when a transfer is made from the transmit data register to the output shift register. The TDRE bit is cleared by reading the status register, then writing a new byte into the transmit data register, TDRE is initialized to 1 by $\overline{RES}$.

Bit 6 **ORFE** Over-Run-Framing Error – set by hardware when an overrun or framing error occurs (receive only). An overrun is defined as a new byte received with last byte still in Data Register/Buffer. A framing error has occurred when the byte boundaries in bit stream are not synchronized to bit counter. If WU flag is set, the ORFE bit will not be set. The ORFE bit is cleared by reading the status register, then reading the Receive Data Register, or by $\overline{RES}$.

Bit 7 **RDRF** Receiver Data Register Full – set by hardware when a transfer from the input shift register to the receiver data register is made. If WU flag is set, the RDRF bit will not be set. The RDRF bit is cleared by reading the status register, then reading the Receive Data Register, or by $\overline{RES}$.

Rate and Mode Control Register

The Rate and Mode Control register controls the following serial I/O variables:

- Baud rate
- format
- clocking source, and
- Port 2 bit 2 configuration

The register consists of 4 bits all of which are write-only and cleared on $\overline{RES}$. The 4 bits in the register may be considered as a pair of 2-bit fields. The two low order bits control the bit rate for internal clocking and the remaining two bits control the format and clock select logic. The register definition is as follows:

Rate and Mode Control Register

7	6	5	4	3	2	1	0	
X	X	X	X	CC1	CC0	SS1	SS0	ADDR : $0010

Bit 0 **SS0** Speed Select – These bits select the Baud rate for
Bit 1 **SS1** the internal clock. The four rates which may be selected are a function of the MPU ϕ_2 clock frequency. Table 5 lists the available Baud rates.

Bit 2 **CC0** Clock Control and Format Select – this 2-bit field
Bit 3 **CC1** controls the format and clock select logic. Table 6 defines the bit field.

Table 5 SCI Bit Times and Rates

	XTAL	2.4576 MHz	4.0 MHz	4.9152 MHz*
SS1 : SS0	E	614.4 kHz	1.0 MHz	1.2288 MHz
0 0	E ÷ 16	26 μs/38,400 Baud	16 μs/62,500 Baud	13 μs/76,800 Baud
0 1	E ÷ 128	208 μs/4,800 Baud	128 μs/7812.5 Baud	104.2 μs/9,600 Baud
1 0	E ÷ 1024	1.67 ms/600 Baud	1.024 ms/976.6 Baud	33.3 μs/1,200 Baud
1 1	E ÷ 4096	6.67 ms/150 Baud	4.096 ms/244.1 Baud	3.33 ms/300 Baud

*HD6801V5 Only

Table 6 SCI Format and Clock Source Control

CC1, CC0	Format	Clock Source	Port 2 Bit 2	Port 2 Bit 3	Port 2 Bit 4
00	–	–	–	**	**
01	NRZ	Internal	Not Used	**	**
10	NRZ	Internal	Output*	**	**
11	NRZ	External	Input	**	**

* Clock output is available regardless of values for bits RE and TE.
** Bit 3 is used for serial input if RE = "1" in TRCS; bit 4 is used for serial output if TE = "1" in TRCS.

Internally Generated Clock

If the user wishes for the serial I/O to furnish a clock, the following requirements are applicable:

- the values of RE and TE are immaterial.
- CC1, CC0 must be set to 10
- the maximum clock rate will be E ÷ 16.
- the clock will be at 1× the bit rate and will have a rising edge at mid-bit.

Externally Generated Clock

If the user wishes to provide an external clock for the serial I/O, the following requirements are applicable:

- the CC1, CC0, field in the Rate and Mode Control Register must be set to 11,
- the external clock must be set to 8 times (×8) the desired baud rate and
- the maximum external clock frequency is 1.0 MHz.

- **Serial Operations**

The serial I/O hardware should be initialized by the HD6801V software prior to operation. This sequence will normally consist of;

- writing the desired operation control bits to the Rate and Mode Control Register and
- writing the desired operational control bits in the Transmit/Receive Control and Status Register.

The Transmitter Enable (TE) and Receiver Enable (RE) bits may be left set for dedicated operations.

Transmit Operations

The transmit operation is enabled by the TE bit in the Transmit/Receive Control and Status Register. This bit when set, gates the output of the serial transmit shift register to Port 2 Bit 4 and takes unconditional control over the Data Direction Register value for Port 2, Bit 4.

Following a $\overline{RES}$ the user should configure both the Rate and Mode Control Register and the Transmit/Receive Control and Status Register for desired operation. Setting the TE bit during this procedure initiates the serial output by first transmitting a ten-bit preamble of 1's. Following the preamble, internal synchronization is established and the transmitter section is ready for operation.

At this point one of two situation exist:

1) if the Transmit Data Register is empty (TDRE = 1), a continuous string of ones will be sent indicating an idle line, or,
2) if data has been loaded into the Transmit Data Register (TDRE = 0), the word is transferred to the output shift register and transmission of the data word will begin.

During the transfer itself, the 0 start bit is first transmitted. Then the 8 data bits (beginning with bit 0) followed by the stop bit, are transmitted. When the Transmitter Data Register has been emptied, the hardware sets the TDRE flag bit.

If the HD6801V fails to respond to the flag within the proper time, (TDRE is still set when the next normal transfer from the parallel data register to the serial output register should occur) then a 1 will be sent (instead of a 0) at "Start" bit time, followed by more 1's until more data is supplied to the data register. No 0's will be sent while TDRE remains a 1.

Receive Operation

The receive operation is enabled by the RE bit which gates in the serial input through Port 2 Bit 3. The receiver section operation is conditioned by the contents of the Transmit/Receive Control and Status Register and the Rate and Mode Control Register.

The receiver bit interval is divided into 8 sub-intervals for internal synchronization. In the NRZ Mode, the received bit stream is synchronized by the first 0 (space) encountered.

The approximate center of each bit time is strobed during the next 10 bits. If the tenth bit is not a 1 (stop bit) a framing error is assumed, and bit ORFE is set. If the tenth bit as a 1, the data is transferred to the Receive Data Register, and interrupt flag RDRF is set. If RDRF is still set at the next tenth bit time, ORFE will be set, indicating an over-run has occurred. When the HD6801V responds to either flag (RDRF or ORFE) by reading the status register followed by reading the Data Register, RDRF (or ORFE) will be cleared.

■ RAM CONTROL REGISTER

This register, which is addressed at $0014, gives status information about the standby RAM. A 0 in the RAM enable bit (RAM E) will disable the standby RAM, thereby protecting it at power down if V_{CC} Standby is held greater than V_{SBB} volts, as explained previously in the signal description for V_{CC} Standby.

RAM Control Register

$0014	STBY PWR	RAME	X	X	X	X	X	X

Bit 0 Not used.
Bit 1 Not used.
Bit 2 Not used.
Bit 3 Not used.
Bit 4 Not used.
Bit 5 Not used.
Bit 6 **RAME** The RAM Enable control bit allows the user the ability to disable the standby RAM. This bit is set to a logic "1" by $\overline{RES}$ which enables the standby RAM and can be written to one or zero under program control. When the RAM is disabled, data is read from external memory.
Big 7 **STBY PWR** The Standby Power bit is cleared when the standby voltage is removed. This bit is a read/write status flag that the user can read which indicates that the standby RAM voltage has been applied, and the data in the standby RAM is valid.

■ GENERAL DESCRIPTION OF INSTRUCTION SET

The HD6801V is upward object code compatible with the HD6800 as it implements the full HMCS6800 instruction set. The execution times of key instructions have been reduced to increase throughout. In addition, new instructions have been added; these include 16-bit operations and a hardware multiply.

Included in the instruction set section are the following:

- MPU Programming Model (Figure 23)
- Addressing modes
- Accumulator and memory instructions – Table 7
- New instructions
- Index register and stack manipulations instructions – Table 8
- Jump and branch instructions – Table 9
- Condition code register manipulation instructions – Table 10
- Instructions Execution times in machine cycles – Table 11
- Summary of cycle by cycle operation – Table 12
- Summary of undefined instructions operation
- Op codes Map – Table 13

• MPU Programming Model

The programming model for the HD6801V is shown in Figure 23. The double (D) accumulator is physically the same as the Accumulator A concatenated with the Accumulator B so that any operation using accumulator D will destroy information in A and B.

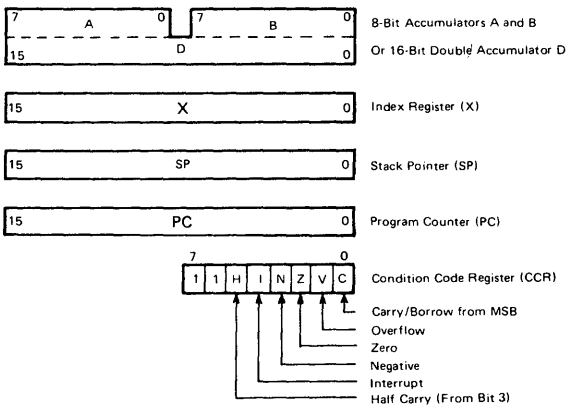


Figure 23 MCU Programming Model

• MPU Addressing Modes

The HD6801V eight-bit microcomputer unit has seven address modes that can be used by a programmer, with the addressing mode a function of both the type of instruction and the coding within the instruction. A summary of the addressing modes for a particular instruction can be found in Table 11 along with the associated instruction execution time that is given in machine cycles. With a clock frequency of 4 MHz, these times would be microseconds.

Accumulator (ACCX) Addressing

In accumulator only addressing, either accumulator A or accumulator B is specified. These are one-byte instructions.

Immediate Addressing

In immediate addressing, the operand is contained in the second byte of the instruction except LDS and LDX which have the operand in the second and third bytes of the instruction. The MCU addresses this location when it fetches the immediate instruction for execution. These are two or three-byte instructions.

Table 7 Accumulator & Memory Instructions

Operations	Mnemonic	Addressing Modes															Boolean/ Arithmetic Operation	Condition Code Register					
		IMMED.			DIRECT			INDEX			EXTEND			IMPLIED				5	4	3	2	1	0
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#		H	I	N	Z	V	C
Add	ADDA	8B	2	2	9B	3	2	AB	4	2	BB	4	3				A + B → A	↕	•	↕	↕	↕	↕
	ADDB	CB	2	2	DB	3	2	EB	4	2	FB	4	3				B + M → B	↕	•	↕	↕	↕	↕
Add Double	ADDD	C3	4	3	D3	5	2	E3	6	2	F3	6	3				A : B + M : M + 1 → A : B	•	•	↕	↕	↕	↕
Add Accumulators	ABA													1B	2	1	A + B → A	↕	•	↕	↕	↕	↕
Add With Carry	ADCA	89	2	2	99	3	2	A9	4	2	B9	4	3				A + M + C → A	↕	•	↕	↕	↕	↕
	ADCB	C9	2	2	D9	3	2	E9	4	2	F9	4	3				B + M + C → B	↕	•	↕	↕	↕	↕
AND	ANDA	84	2	2	94	3	2	A4	4	2	B4	4	3				A·M → A	•	•	↕	↕	R	•
	ANDB	C4	2	2	D4	3	2	E4	4	2	F4	4	3				B·M → B	•	•	↕	↕	R	•
Bit Test	BIT A	85	2	2	95	3	2	A5	4	2	B5	4	3				A·M	•	•	↕	↕	R	•
	BIT B	C5	2	2	D5	3	2	E5	4	2	F5	4	3				B·M	•	•	↕	↕	R	•
Clear	CLR							6F	6	2	7F	6	3				00 → M	•	•	R	S	R	R
	CLRA													4F	2	1	00 → A	•	•	R	S	R	R
	CLRB													5F	2	1	00 → B	•	•	R	S	R	R
Compare	CMPA	81	2	2	91	3	2	A1	4	2	B1	4	3				A − M	•	•	↕	↕	↕	↕
	CMPB	C1	2	2	D1	3	2	E1	4	2	F1	4	3				B − M	•	•	↕	↕	↕	↕
Compare Accumulators	CBA													11	2	1	A − B	•	•	↕	↕	↕	↕
Complement, 1's	COM							63	6	2	73	6	3				$\overline{M} \rightarrow M$	•	•	↕	↕	R	S
	COMA													43	2	1	$\overline{A} \rightarrow A$	•	•	↕	↕	R	S
	COMB													53	2	1	$\overline{B} \rightarrow B$	•	•	↕	↕	R	S
Complement, 2's (Negate)	NEG							60	6	2	70	6	3				00 − M → M	•	•	↕	↕	①	②
	NEGA													40	2	1	00 − A → A	•	•	↕	↕	①	②
	NEGB													50	2	1	00 − B → B	•	•	↕	↕	①	②
Decimal Adjust, A	DAA													19	2	1	Converts binary add of BCD characters into BCD format	•	•	↕	↕	↕	③
Decrement	DEC							6A	6	2	7A	6	3				M − 1 → M	•	•	↕	↕	④	•
	DECA													4A	2	1	A − 1 → A	•	•	↕	↕	④	•
	DECB													5A	2	1	B − 1 → B	•	•	↕	↕	④	•
Exclusive OR	EORA	88	2	2	98	3	2	A8	4	2	B8	4	3				A ⊕ M → A	•	•	↕	↕	R	•
	EORB	C8	2	2	D8	3	2	E8	4	2	F8	4	3				B ⊕ M → B	•	•	↕	↕	R	•
Increment	INC							6C	6	2	7C	6	3				M + 1 → M	•	•	↕	↕	⑤	•
	INCA													4C	2	1	A + 1 → A	•	•	↕	↕	⑤	•
	INCB													5C	2	1	B + 1 → B	•	•	↕	↕	⑤	•
Load Accumulator	LDAA	86	2	2	96	3	2	A6	4	2	B6	4	3				M → A	•	•	↕	↕	R	•
	LDAB	C6	2	2	D6	3	2	E6	4	2	F6	4	3				M → B	•	•	↕	↕	R	•
Load Double Accumulator	LDD	CC	3	3	DC	4	2	EC	5	2	FC	5	3				M + 1 → B, M → A	•	•	↕	↕	R	•
Multiply Unsigned	MUL													3D	10	1	A x B → A : B	•	•	•	•	•	⑪
OR, Inclusive	ORAA	8A	2	2	9A	3	2	AA	4	2	BA	4	3				A + M → A	•	•	↕	↕	R	•
	ORAB	CA	2	2	DA	3	2	EA	4	2	FA	4	3				B + M → B	•	•	↕	↕	R	•
Push Data	PSHA													36	3	1	A → Msp, SP − 1 → SP	•	•	•	•	•	•
	PSHB													37	3	1	B → Msp, SP − 1 → SP	•	•	•	•	•	•
Pull Data	PULA													32	4	1	SP + 1 → SP, Msp → A	•	•	•	•	•	•
	PULB													33	4	1	SP + 1 → SP, Msp → B	•	•	•	•	•	•
Rotate Left	ROL							69	6	2	79	6	3				M, A, B } C b7 b0	•	•	↕	↕	⑥	↕
	ROLA													49	2	1		•	•	↕	↕	⑥	↕
	ROLB													59	2	1		•	•	↕	↕	⑥	↕
Rotate Right	ROR							66	6	2	76	6	3				M, A, B } C b7 b0	•	•	↕	↕	⑥	↕
	RORA													46	2	1		•	•	↕	↕	⑥	↕
	RORB													56	2	1		•	•	↕	↕	6	↕

The Condition Code Register notes are listed after Table 10.

(Continued)

Table 7 Accumulator & Memory Instructions (Continued)

Operations	Mnemonic	Addressing Modes															Boolean/ Arithmetic Operation	Condition Code Register					
		IMMED.			DIRECT			INDEX			EXTEND			IMPLIED				5	4	3	2	1	0
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#		H	I	N	Z	V	C
Shift Left Arithmetic	ASL							68	6	2	78	6	3				M	•	•	↕	↕	⑥	↕
	ASLA													48	2	1	A	•	•	↕	↕	⑥	↕
	ASLB													58	2	1	B	•	•	↕	↕	⑥	↕
Double Shift Left, Arithmetic	ASLD													05	3	1	ACC A/ ACC B	•	•	↕	↕	⑥	↕
Shift Right Arithmetic	ASR							67	6	2	77	6	3				M	•	•	↕	↕	⑥	↕
	ASRA													47	2	1	A	•	•	↕	↕	⑥	↕
	ASRB													57	2	1	B	•	•	↕	↕	⑥	↕
Shift Right Logical	LSR							64	6	2	74	6	3				M	•	•	↕	↕	⑥	↕
	LSRA													44	2	1	A	•	•	↕	↕	⑥	↕
	LSRB													54	2	1	B	•	•	↕	↕	⑥	↕
Double Shift Right Logical	LSRD													04	3	1	0→ ACC A/ ACC B →C	•	•	R	↕	⑥	↕
Store Accumulator	STAA				97	3	2	A7	4	2	B7	4	3				A → M	•	•	↕	↕	R	•
	STAB				D7	3	2	E7	4	2	F7	4	3				B → M	•	•	↕	↕	R	•
Store Double Accumulator	STD				DD	4	2	ED	5	2	FD	5	3				A → M B → M + 1	•	•	↕	↕	R	•
Subtract	SUBA	80	2	2	90	3	2	A0	4	2	B0	4	3				A − M → A	•	•	↕	↕	↕	↕
	SUBB	C0	2	2	D0	3	2	E0	4	2	F0	4	3				B − M → B	•	•	↕	↕	↕	↕
Double Subtract	SUBD	83	4	3	93	5	2	A3	6	2	B3	6	3				A : B − M : M + 1 → A : B	•	•	↕	↕	↕	↕
Subtract Accumulators	SBA													10	2	1	A − B → A	•	•	↕	↕	↕	↕
Subtract With Carry	SBCA	82	2	2	92	3	2	A2	4	2	B2	4	3				A − M − C → A	•	•	↕	↕	↕	↕
	SBCB	C2	2	2	D2	3	2	E2	4	2	F2	4	3				B − M − C → B	•	•	↕	↕	↕	↕
Transfer Accumulators	TAB													16	2	1	A → B	•	•	↕	↕	R	•
	TBA													17	2	1	B → A	•	•	↕	↕	R	•
Test Zero or Minus	TST							6D	6	2	7D	6	3				M − 00	•	•	↕	↕	R	R
	TSTA													4D	2	1	A − 00	•	•	↕	↕	R	R
	TSTB													5D	2	1	B − 00	•	•	↕	↕	R	R

The Condition Code Register notes are listed after Table 10.

Direct Addressing

In direct addressing, the address of the operand is contained in the second byte of the instruction. Direct addressing allows the user to directly address the lowest 256 bytes in the machine i.e., locations zero through 255. Enhanced execution times are achieved by storing data in these locations. In most configurations, it should be a random access memory. These are two-byte instructions.

Extended Addressing

In extended addressing, the address contained in the second byte of the instruction is used as the higher eight-bits of the address of the operand. The third byte of the instruction is used as the lower eight-bits of the address for the operand. This is an absolute address in memory. These are three-byte instructions.

Indexed Addressing

In indexed addressing, the address contained in the second byte of the instruction is added to the index register's lowest eight bits in the MCU. The carry is then added to the higher order eight bits of the index register. This result is then used to address memory. The modified address is held in a temporary address register so there is no change to the index register. These are two-byte instructions.

Implied Addressing

In the implied addressing mode the instruction gives the address (i.e., stack pointer, index register, etc.). These are one-byte instructions.

Relative Addressing

In relative addressing, the address contained in the second byte of the instruction is added to the program counter's lowest eight bits plus two. The carry or borrow is then added to the high eight bits. This allows the user to address data within a range of -126 to +129 bytes of the present instruction. These are two-byte instructions.

- **New Instructions**

In addition to the existing 6800 Instruction Set, the following new instructions are incorporated in the HD6801V Microcomputer.

ABX Adds the 8-bit unsigned accumulator B to the 16-bit X-Register taking into account the possible carry out of the low order byte of the X-Register.

ADDD Adds the double precision ACCD* to the double precision value M:M+1 and places the results in ACCD.

ASLD Shifts all bits of ACCD one place to the left. Bit 0 is loaded with zero. The C bit is loaded from the most significant bit of ACCD.

LDD Loads the contents of double precision memory location into the double accumulator A:B. The condition codes are set according to the data.

LSRD Shifts all bits of ACCD one place to the right. Bit 15 is loaded with zero. The C bit is loaded from the least significant bit to ACCD.

MUL Multiplies the 8 bits in accumulator A with the 8 bits in accumulator B to obtain a 16-bit unsigned number in A:B, ACCA contains MSB of result.

PSHX The contents of the index register is pushed onto the stack at the address contained in the stack pointer. The stack pointer is decremented by 2.

PULX The index register is pulled from the stack beginning at the current address contained in the stack pointer +1. The stack pointer is incremented by 2 in total.

STD Stores the contents of double accumulator A:B in memory. The contents of ACCD remain unchanged.

SUBD Subtracts the contents of M:M + 1 from the contents of double accumulator AB and places the result in ACCD.

BRN Never branches. If effect, this instruction can be considered a two byte NOP (No operation) requiring three cycles for execution.

CPX Internal processing modified to permit its use with any conditional branch instruction.

*ACCD is the 16 bit register (A:B) formed by concatenating the A and B accumulators. The A-accumulator is the most significant byte.

Table 8 Index Register and Stack Manipulation Instructions

Pointer Operations	Mnemonic	Addressing Modes															Boolean/ Arithmetic Operation	Condition Code Register					
		IMMED.			DIRECT			INDEX			EXTND			IMPLIED				5	4	3	2	1	0
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#		H	I	N	Z	V	C
Compare Index Reg	CPX	8C	4	3	9C	5	2	AC	6	2	BC	6	3				X − M : M + 1	•	•	↕	↕	↕	↕
Decrement Index Reg	DEX													09	3	1	X − 1 → X	•	•	•	↕	•	•
Decrement Stack Pntr	DES													34	3	1	SP − 1 → SP	•	•	•	•	•	•
Increment Index Reg	INX													08	3	1	X + 1 → X	•	•	•	↕	•	•
Increment Stack Pntr	INS													31	3	1	SP + 1 → SP	•	•	•	•	•	•
Load Index Reg	LDX	CE	3	3	DE	4	2	EE	5	2	FE	5	3				M → X_H, (M + 1) → X_L	•	•	⑦	↕	R	•
Load Stack Pntr	LDS	8E	3	3	9E	4	2	AE	5	2	BE	5	3				M → SP_H, (M+1) → SP_L	•	•	⑦	↕	R	•
Store Index Reg	STX				DF	4	2	EF	5	2	FF	5	3				X_H → M, X_L → (M + 1)	•	•	⑦	↕	R	•
Store Stack Pntr	STS				9F	4	2	AF	5	2	BF	5	3				SP_H → M, SP_L → (M + 1)	•	•	⑦	↕	R	•
Index Reg → Stack Pntr	TXS													35	3	1	X − 1 → SP	•	•	•	•	•	•
Stack Pntr → Index Reg	TSX													30	3	1	SP + 1 → X	•	•	•	•	•	•
Add	ABX													3A	3	1	B + X → X	•	•	•	•	•	•
Push Data	PSHX													3C	4	1	X_L → M_{sp}, SP − 1 → SP X_H → M_{sp}, SP − 1 → SP	•	•	•	•	•	•
Pull Data	PULX													38	5	1	SP + 1 → SP, M_{sp} → X_H SP + 1 → SP, M_{sp} → X_L	•	•	•	•	•	•

The Condition Code Register notes are listed after Table 10.

Table 9 Jump and Branch Instructions

Operations	Mnemonic	Addressing Modes														Branch Test	Condition Code Register					
		RELATIVE			DIRECT			INDEX			EXTND			IMPLIED				5	4	3	2	1	0
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#		H	I	N	Z	V	C
Branch Always	BRA	20	3	2													None	•	•	•	•	•	•
Branch Never	BRN	21	3	2													None	•	•	•	•	•	•
Branch If Carry Clear	BCC	24	3	2													C = 0	•	•	•	•	•	•
Branch If Carry Set	BCS	25	3	2													C = 1	•	•	•	•	•	•
Branch If = Zero	BEQ	27	3	2													Z = 1	•	•	•	•	•	•
Branch If ≥ Zero	BGE	2C	3	2													N ⊕ V = 0	•	•	•	•	•	•
Branch If > Zero	BGT	2E	3	2													Z + (N ⊕ V) = 0	•	•	•	•	•	•
Branch If Higher	BHI	22	3	2													C + Z = 0	•	•	•	•	•	•
Branch If ≤ Zero	BLE	2F	3	2													Z + (N ⊕ V) = 1	•	•	•	•	•	•
Branch If Lower Or Same	BLS	23	3	2													C + Z = 1	•	•	•	•	•	•
Branch If < Zero	BLT	2D	3	2													N ⊕ V = 1	•	•	•	•	•	•
Branch If Minus	BMI	2B	3	2													N = 1	•	•	•	•	•	•
Branch If Not Equal Zero	BNE	26	3	2													Z = 0	•	•	•	•	•	•
Branch If Overflow Clear	BVC	28	3	2													V = 0	•	•	•	•	•	•
Branch If Overflow Set	BVS	29	3	2													V = 1	•	•	•	•	•	•
Branch If Plus	BPL	2A	3	2													N = 0	•	•	•	•	•	•
Branch To Subroutine	BSR	8D	6	2													See Special Operations	•	•	•	•	•	•
Jump	JMP							6E	3	2	7E	3	3					•	•	•	•	•	•
Jump To Subroutine	JSR				9D	5	2	AD	6	2	BD	6	3					•	•	•	•	•	•
No Operation	NOP													01	2	1	Advances Prog. Cntr. Only	•	•	•	•	•	•
Return From Interrupt	RTI													3B	10	1	See Special Operations	⑧					
Return From Subroutine	RTS													39	5	1		•	•	•	•	•	•
Software Interrupt	SWI													3F	12	1		•	S	•	•	•	•
Wait for Interrupt	WAI													3E	9	1		•	⑨	•	•	•	•

Table10 Condition Code Register Manipulation Instructions

Operations	Mnemonic	Addressing Modes			Boolean Operation	Condition Code Register					
		IMPLIED				5	4	3	2	1	0
		OP	~	#		H	I	N	Z	V	C
Clear Carry	CLC	0C	2	1	0 → C	•	•	•	•	•	R
Clear Interrupt Mask	CLI	0E	2	1	0 → I	•	R	•	•	•	•
Clear Overflow	CLV	0A	2	1	0 → V	•	•	•	•	R	•
Set Carry	SEC	0D	2	1	1 → C	•	•	•	•	•	S
Set Interrupt Mask	SEI	0F	2	1	1 → I	•	S	•	•	•	•
Set Overflow	SEV	0B	2	1	1 → V	•	•	•	•	S	•
Accumulator A → CCR	TAP	06	2	1	A → CCR	⑩					
CCR → Accumulator A	TPA	07	2	1	CCR → A	•	•	•	•	•	•

Condition Code Register Notes: (Bit set it test is true and cleared otherwise)

① (Bit V) Test: Result = 10000000?
② (Bit C) Test: Result = 00000000?
③ (Bit C) Test: Decimal value of most significant BCD Character greater than nine? (Not cleared if previously set)
④ (Bit V) Test: Operand = 10000000 prior to execution?
⑤ (Bit V) Test: Operand = 01111111 prior to execution?
⑥ (Bit V) Test: Set equal to result to N ⊕ C after shift has occurred.
⑦ (Bit N) Test: Result less than zero? (Bit 15 = 1)
⑧ (All) Load Condition Code Register from Stack. (See Special Operations)
⑨ (Bit I) Set when interrupt occurs. If previously set, a Non-Maskable Interrupt is required to exit the wait state.
⑩ (All) Set according to the contents of Accumulator A.
⑪ (Bit C) Set equal to result of Bit 7 (AccB)

Table 11 Instruction Execution Times in Machine Cycle

	ACCX	Immediate	Direct	Extended	Indexed	Implied	Relative		ACCX	Immediate	Direct	Extended	Indexed	Implied	Relative
ABA	•	•	•	•	•	2	•	INX	•	•	•	•	•	3	•
ABX	•	•	•	•	•	3	•	JMP	•	•	•	3	3	•	•
ADC	•	2	3	4	4	•	•	JSR	•	•	5	6	6	•	•
ADD	•	2	3	4	4	•	•	LDA	•	2	3	4	4	•	•
ADDD	•	4	5	6	6	•	•	LDD	•	3	4	5	5	•	•
AND	•	2	3	4	4	•	•	LDS	•	3	4	5	5	•	•
ASL	2	•	•	6	6	•	•	LDX	•	3	4	5	5	•	•
ASLD	•	•	•	•	•	3	•	LSR	2	•	•	6	6	•	•
ASR	2	•	•	6	6	•	•	LSRD	•	•	•	•	•	3	•
BCC	•	•	•	•	•	•	3	MUL	•	•	•	•	•	10	•
BCS	•	•	•	•	•	•	3	NEG	2	•	•	6	6	•	•
BEQ	•	•	•	•	•	•	3	NOP	•	•	•	•	•	2	•
BGE	•	•	•	•	•	•	3	ORA	•	2	3	4	4	•	•
BGT	•	•	•	•	•	•	3	PSH	3	•	•	•	•	•	•
BHI	•	•	•	•	•	•	3	PSHX	•	•	•	•	•	4	•
BIT	•	2	3	4	4	•	•	PUL	4	•	•	•	•	•	•
BLE	•	•	•	•	•	•	3	PULX	•	•	•	•	•	5	•
BLS	•	•	•	•	•	•	3	ROL	2	•	•	6	6	•	•
BLT	•	•	•	•	•	•	3	ROR	2	•	•	6	6	•	•
BMI	•	•	•	•	•	•	3	RTI	•	•	•	•	•	10	•
BNE	•	•	•	•	•	•	3	RTS	•	•	•	•	•	5	•
BPL	•	•	•	•	•	•	3	SBA	•	•	•	•	•	2	•
BRA	•	•	•	•	•	•	3	SBC	•	2	3	4	4	•	•
BRN	•	•	•	•	•	•	3	SEC	•	•	•	•	•	2	•
BSR	•	•	•	•	•	•	6	SEI	•	•	•	•	•	2	•
BVC	•	•	•	•	•	•	3	SEV	•	•	•	•	•	2	•
BVS	•	•	•	•	•	•	3	STA	•	•	3	4	4	•	•
CBA	•	•	•	•	•	2	•	STD	•	•	4	5	5	•	•
CLC	•	•	•	•	•	2	•	STS	•	•	4	5	5	•	•
CLI	•	•	•	•	•	2	•	STX	•	•	4	5	5	•	•
CLR	2	•	•	6	6	•	•	SUB	•	2	3	4	4	•	•
CLV	•	•	•	•	•	2	•	SUBD	•	4	5	6	6	•	•
CMP	•	2	3	4	4	•	•	SWI	•	•	•	•	•	12	•
COM	2	•	•	6	6	•	•	TAB	•	•	•	•	•	2	•
CPX	•	4	5	6	6	•	•	TAP	•	•	•	•	•	2	•
DAA	•	•	•	•	•	2	•	TBA	•	•	•	•	•	2	•
DEC	2	•	•	6	6	•	•	TPA	•	•	•	•	•	2	•
DES	•	•	•	•	•	3	•	TST	2	•	•	6	6	•	•
DEX	•	•	•	•	•	3	•	TSX	•	•	•	•	•	3	•
EOR	•	2	3	4	4	•	•	TXS	•	•	•	•	•	3	•
INC	2	•	•	6	6	•	•	WAI	•	•	•	•	•	9	•
INS	•	•	•	•	•	3	•								

- **Summary of Cycle by Cycle Operation**

Table 12 provides a detailed description of the information present on the Address Bus, Data Bus, and the Read/Write line (R/$\overline{W}$) during each cycle for each instruction.

This information is useful in comparing actual with expected results during debug of both software and hardware as the control program is executed. The information is categorized in groups according to addressing mode and number of cycles per instruction. (In general, instructions with the same addressing mode and number of cycles execute in the same manner; exceptions are indicated in the table).

Table 12 Cycle by Cycle Operation

Address Mode & Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
IMMEDIATE					
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	2	1 2	Op Code Address Op Code Address + 1	1 1	Op Code Operand Data
LDS LDX	3	1 2 3	Op Code Address Op Code Address + 1 Op Code Address + 2	1 1 1	Op Code Operand Data (High Order Byte) Operand Data (Low Order Byte)
CPX SUBD ADDD	4	1 2 3 4	Op Code Address Op Code Address + 1 Op Code Address + 2 Address Bus FFFF	1 1 1 1	Op Code Operand Data (High Order Byte) Operand Data (Low Order Byte) Low Byte of Restart Vector
DIRECT					
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	3	1 2 3	Op Code Address Op Code Address + 1 Address of Operand	1 1 1	Op Code Address of Operand Operand Data
STA	3	1 2 3	Op Code Address Op Code Address + 1 Destination Address	1 1 0	Op Code Destination Address Data from Accumulator
LDS LDX LDD	4	1 2 3 4	Op Code Address Op Code Address + 1 Address of Operand Operand Address + 1	1 1 1 1	Op Code Address of Operand Operand Data (High Order Byte) Operand Data (Low Order Byte)
STS STX STD	4	1 2 3 4	Op Code Address Op Code Address + 1 Address of Operand Address of Operand + 1	1 1 0 0	Op Code Address of Operand Register Data (High Order Byte) Register Data (Low Order Byte)
CPX SUBD ADDD	5	1 2 3 4 5	Op Code Address Op Code Address + 1 Operand Address Operand Address + 1 Address Bus FFFF	1 1 1 1 1	Op Code Address of Operand Operand Data (High Order Byte) Operand Data (Low Order Byte) Low Byte of Restart Vector
JSR	5	1 2 3 4 5	Op Code Address Op Code Address + 1 Subroutine Address Stack Pointer Stack Pointer + 1	1 1 1 0 0	Op Code Irrelevant Data First Subroutine Op Code Return Address (Low Order Byte) Return Address (High Order Byte)

(Continued)

Table 12 Cycle by Cycle Operation (Continued)

Address Mode & Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
INDEXED					
JMP	3	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
ADC EOR	4	1	Op Code Address	1	Op Code
ADD LDA		2	Op Code Address + 1	1	Offset
AND ORA		3	Address Bus FFFF	1	Low Byte of Restart Vector
BIT SBC		4	Index Register Plus Offset	1	Operand Data
CMP SUB					
STA	4	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	0	Operand Data
LDS	5	1	Op Code Address	1	Op Code
LDX		2	Op Code Address + 1	1	Offset
LDD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	1	Operand Data (High Order Byte)
		5	Index Register Plus Offset + 1	1	Operand Data (Low Order Byte)
STS	5	1	Op Code Address	1	Op Code
STX		2	Op Code Address + 1	1	Offset
STD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	0	Operand Data (High Order Byte)
		5	Index Register Plus Offset + 1	0	Operand Data (Low Order Byte)
ASL LSR	6	1	Op Code Address	1	Op Code
ASR NEG		2	Op Code Address + 1	1	Offset
CLR ROL		3	Address Bus FFFF	1	Low Byte of Restart Vector
COM ROR		4	Index Register Plus Offset	1	Current Operand Data
DEC TST*		5	Address Bus FFFF	1	Low Byte of Restart Vector
INC		6	Index Register Plus Offset	0	New Operand Data
CPX	6	1	Op Code Address	1	Op Code
SUBD		2	Op Code Address + 1	1	Offset
ADDD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register + Offset	1	Operand Data (High Order Byte)
		5	Index Register + Offset + 1	1	Operand Data (Low Order Byte)
		6	Address Bus FFFF	1*	Low Byte of Restart Vector
JSR	6	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register + Offset	1	First Subroutine Op Code
		5	Stack Pointer	0	Return Address (Low Order Byte)
		6	Stack Pointer – 1	0	Return Address (High Order Byte)

* In the TST instruction, R/$\overline{W}$ line of the sixth cycle is "1" level, and AB=FFFF, DB=Low Byte of Reset Vector. (Continued)

Table 12 Cycle by Cycle Operation (Continued)

Address Mode & Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
EXTENDED					
JMP	3	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Jump Address (High Order Byte)
		3	Op Code Address + 2	1	Jump Address (Low Order Byte)
ADC EOR	4	1	Op Code Address	1	Op Code
ADD LDA		2	Op Code Address + 1	1	Address of Operand
AND ORA		3	Op Code Address + 2	1	Address of Operand (Low Order Byte)
BIT SBC		4	Address of Operand	1	Operand Data
CMP SUB					
STA A	4	1	Op Code Address	1	Op Code
STA B		2	Op Code Address + 1	1	Destination Address (High Order Byte)
		3	Op Code Address + 2	1	Destination Address (Low Order Byte)
		4	Operand Destination Address	0	Data from Accumulator
LDS	5	1	Op Code Address	1	Op Code
LDX		2	Op Code Address + 1	1	Address of Operand (High Order Byte)
LDD		3	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	Address of Operand	1	Operand Data (High Order Byte)
		5	Address of Operand + 1	1	Operand Data (Low Order Byte)
STS	5	1	Op Code Address	1	Op Code
STX		2	Op Code Address + 1	1	Address of Operand (High Order Byte)
STD		3	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	Address of Operand	0	Operand Data (High Order Byte)
		5	Address of Operand + 1	0	Operand Data (Low Order Byte)
ASL LSR	6	1	Op Code Address	1	Op Code
ASR NEG		2	Op Code Address + 1	1	Address of Operand (High Order Byte)
CLR ROL		3	Op Code Address + 2	1	Address of Operand (Low Order Byte)
COM ROR		4	Address of Operand	1	Current Operand Data
DEC TST*		5	Address Bus FFFF	1	Low Byte of Restart Vector
INC		6	Address of Operand	0	New Operand Data
CPX	6	1	Op Code Address	1	Op Code
SUBD		2	Op Code Address + 1	1	Operand Address (High Order Byte)
ADDD		3	Op Code Address + 2	1	Operand Address (Low Order Byte)
		4	Operand Address	1	Operand Data (High Order Byte)
		5	Operand Address + 1	1	Operand Data (Low Order Byte)
		6	Address Bus FFFF	1	Low Byte of Restart Vector
JSR	6	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Address of Subroutine (High Order Byte)
		3	Op Code Address + 2	1	Address of Subroutine (Low Order Byte)
		4	Subroutine Starting Address	1	Op Code of Next Instruction
		5	Stack Pointer	0	Return Address (Low Order Byte)
		6	Stack Pointer – 1	0	Address of Operand (High Order Byte)

* In the TST instruction, R/$\overline{W}$ line of the sixth cycle is "1" level, and AB = FFFF, DB = Low Byte of Reset Vector.

(Continued)

Table 12 Cycle by Cycle Operation (Continued)

Address Mode & Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
IMPLIED					
ABA DAA SEC ASL DEC SEI ASR INC SEV CBA LSR TAB CLC NEG TAP CLI NOP TBA CLR ROL TPA CLV ROR TST COM SBA	2	1 2	Op Code Address Op Code Address + 1	1 1	Op Code Op Code of Next Instruction
ABX	3	1 2 3	Op Code Address Op Code Address + 1 Address Bus FFFF	1 1 1	Op Code Irrelevant Data Low Byte of Restart Vector
ASLD LSRD	3	1 2 3	Op Code Address Op Code Address + 1 Address Bus FFFF	1 1 1	Op Code Irrelevant Data Low Byte of Restart Vector
DES INS	3	1 2 3	Op Code Address Op Code Address + 1 Previous Register Contents	1 1 1	Op Code Op Code of Next Instruction Irrelevant Data
INX DEX	3	1 2 3	Op Code Address Op Code Address + 1 Address Bus FFFF	1 1 1	Op Code Op Code of Next Instruction Low Byte of Restart Vector
PSHA PSHB	3	1 2 3	Op Code Address Op Code Address + 1 Stack Pointer	1 1 0	Op Code Op Code of Next Instruction Accumulator Data
TSX	3	1 2 3	Op Code Address Op Code Address + 1 Stack Pointer	1 1 1	Op Code Op Code of Next Instruction Irrelevant Data
TXS	3	1 2 3	Op Code Address Op Code Address + 1 Address Bus FFFF	1 1 1	Op Code Op Code of Next Instruction Low Byte of Restart Vector
PULA PULB	4	1 2 3 4	Op Code Address Op Code Address + 1 Stack Pointer Stack Pointer + 1	1 1 1 1	Op Code Op Code of Next Instruction Irrelevant Data
PSHX	4	1 2 3 4	Op Code Address Op Code Address + 1 Stack Pointer Stack Pointer – 1	1 1 0 0	Op Code Irrelevant Data Index Register (Low Order Byte) Index Register (High Order Byte)
PULX	5	1 2 3 4 5	Op Code Address Op Code Address + 1 Stack Pointer Stack Pointer + 1 Stack Pointer +2	1 1 1 1 1	Op Code Irrelevant Data Irrelevant Data Index Register (High Order Byte) Index Register (Low Order Byte)
RTS	5	1 2 3 4 5	Op Code Address Op Code Address + 1 Stack Pointer Stack Pointer + 1 Stack Pointer + 2	1 1 1 1 1	Op Code Irrelevant Data Irrelevant Data Address of Next Instruction (High Order Byte) Address of Next Instruction (Low Order Byte)
WAI**	9	1 2 3 4	Op Code Address Cp Code Address + 1 Stack Pointer Stack Pointer – 1	1 1 0 0	Op Code Op Code of Next Instruction Return Address (Low Order Byte) Return Address (High Order Byte)

(Continued)

Table 12 Cycle by Cycle Operation (Continued)

Address Mode & Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
WAI**		5	Stack Pointer – 2	0	Index Register (Low Order Byte)
		6	Stack Pointer – 3	0	Index Register (High Order Byte)
		7	Stack Pointer – 4	0	Contents of Accumulator A
		8	Stack Pointer – 5	0	Contents of Accumulator B
		9	Stack Pointer – 6	0	Contents of Cond. Code Register
MUL	10	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Irrelevant Data
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Address Bus FFFF	1	Low Byte of Restart Vector
		5	Address Bus FFFF	1	Low Byte of Restart Vector
		6	Address Bus FFFF	1	Low Byte of Restart Vector
		7	Address Bus FFFF	1	Low Byte of Restart Vector
		8	Address Bus FFFF	1	Low Byte of Restart Vector
		9	Address Bus FFFF	1	Low Byte of Restart Vector
		10	Address Bus FFFF	1	Low Byte of Restart Vector
RTI	10	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Irrelevant Data
		3	Stack Pointer	1	Irrelevant Data
		4	Stack Pointer + 1	1	Contents of Cond. Code Reg. from Stack
		5	Stack Pointer + 2	1	Contents of Accumulator B from Stack
		6	Stack Pointer + 3	1	Contents of Accumulator A from Stack
		7	Stack Pointer + 4	1	Index Register from Stack (High Order Byte)
		8	Stack Pointer + 5	1	Index Register from Stack (Low Order Byte)
		9	Stack Pointer + 6	1	Next Instruction Address from Stack (High Order Byte)
		10	Stack Pointer + 7	1	Next Instruction Address from Stack (Low Order Byte)
SWI	12	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Irrelevant Data
		3	Stack Pointer	0	Return Address (Low Order Byte)
		4	Stack Pointer – 1	0	Return Address (High Order Byte)
		5	Stack Pointer – 2	0	Index Register (Low Order Byte)
		6	Stack Pointer – 3	0	Index Register (High Order Byte)
		7	Stack Pointer – 4	0	Contents of Accumulator A
		8	Stack Pointer – 5	0	Contents of Accumulator B
		9	Stack Pointer – 6	0	Contents of Cond. Code Register
		10	Stack Pointer – 7	1	Irrelevant Data
		11	Vector Address FFFA (Hex)	1	Address of Subroutine (High Order Byte)
		12	Vector Address FFFB (Hex)	1	Address of Subroutine (Low Order Byte)

**While the MPU is in the "Wait" state, its bus state will appear as a series of MPU reads of an address which is seven locations less than the original contents of the Stack Pointer. Contrary to the HD6800, none of the ports are driven to the high impedance state by a WAI instruction.

(Continued)

Table 12 Cycle by Cycle Operation (Continued)

Address Mode & Instruction	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
RELATIVE					
BCC BHT BNE BCS BLE BPL BEQ BLS BRA BGE BLT BVC BGT BMT BVS BRN	3	1 2 3	Op Code Address Op Code Address + 1 Address Bus FFFF	1 1 1	Op Code Branch Offset Low Byte of Restart Vector
BSR	6	1 2 3 4 5 6	Op Code Address Op Code Address + 1 Address Bus FFFF Subroutine Starting Address Stack Pointer Stack Pointer – 1	1 1 1 1 0 0	Op Code Branch Offset Low Byte of Restart Vector Op Code of Next Instruction Return Address (Low Order Byte) Return Address (High Order Byte)

- **Summary of Undefined Instruction Operations**

The HD6801V has 36 undefined instructions. When these are carried out, the contents of Register and Memory in MPU change at random.

When the op codes (4E, 5E) are used to execute, the MPU continues to increase the program counter and it will not stop until the Reset signal enters. These op codes are used to test the LSI.

Table 13 Op codes Map

HD6801V MICROCOMPUTER INSTRUCTIONS

OP CODE						ACC A	ACC B	IND	EXT	ACCA or SP				ACCB or X				
										IMM	DIR	IND	EXT	IMM	DIR	IND	EXT	
LO \ HI		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111	
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0000	0		SBA	BRA	TSX	NEG				SUB								0
0001	1	NOP	CBA	BRN	INS					CMP								1
0010	2			BHI	PULA (+1)					SBC								2
0011	3			BLS	PULB (+1)	COM				*	SUBD (+2)			*	ADDD (+2)			3
0100	4	LSRD (+1)		BCC	DES	LSR				AND								4
0101	5	ASLD (+1)		BCS	TXS					BIT								5
0110	6	TAP	TAB	BNE	PSHA	ROR				LDA								6
0111	7	TPA	TBA	BEQ	PSHB	ASR					STA				STA			7
1000	8	INX (+1)		BVC	PULX (+2)	ASL				EOR								8
1001	9	DEX (+1)	DAA	BVS	RTS (+2)	ROL				ADC								9
1010	A	CLV		BPL	ABX	DEC				ORA								A
1011	B	SEV	ABA	BMI	RTI (+7)					ADD								B
1100	C	CLC		BGE	PSHX (+1)	INC				*	CPX (+2)			*	LDD (+1)			C
1101	D	SEC		BLT	MUL (+7)	TST				BSR (+4)	JSR (+2)			* (+1)	STD (+1)			D
1110	E	CLI		BGT	WAI (+6)	**		JMP (-3)		*	LDS (+1)			*	LDX (+1)			E
1111	F	SEI		BLE	SWI (+9)	CLR				* (+1)	STS (+1)			* (+1)	STX (+1)			F
BYTE/CYCLE		1/2	1/2	2/3	1/3	1/2	1/2	2/6	3/6	2/2	2/3	2/4	3/4	2/2	2/3	2/4	3/4	

[NOTES] 1) Undefined Op codes are marked with ▨.

2) ( ) indicate that the number in parenthesis must be added to the cycle count for that instruction.

3) The instructions shown below are all 3 bytes and are marked with "*".
Immediate addressing mode of SUBD, CPX, LDS, ADDD, LDD and LDX instructions, and undefined op codes (8F, CD, CF).

4) The Op codes (4E, 5E) are 1 byte/∞ cycles instructions, and are marked with "**".

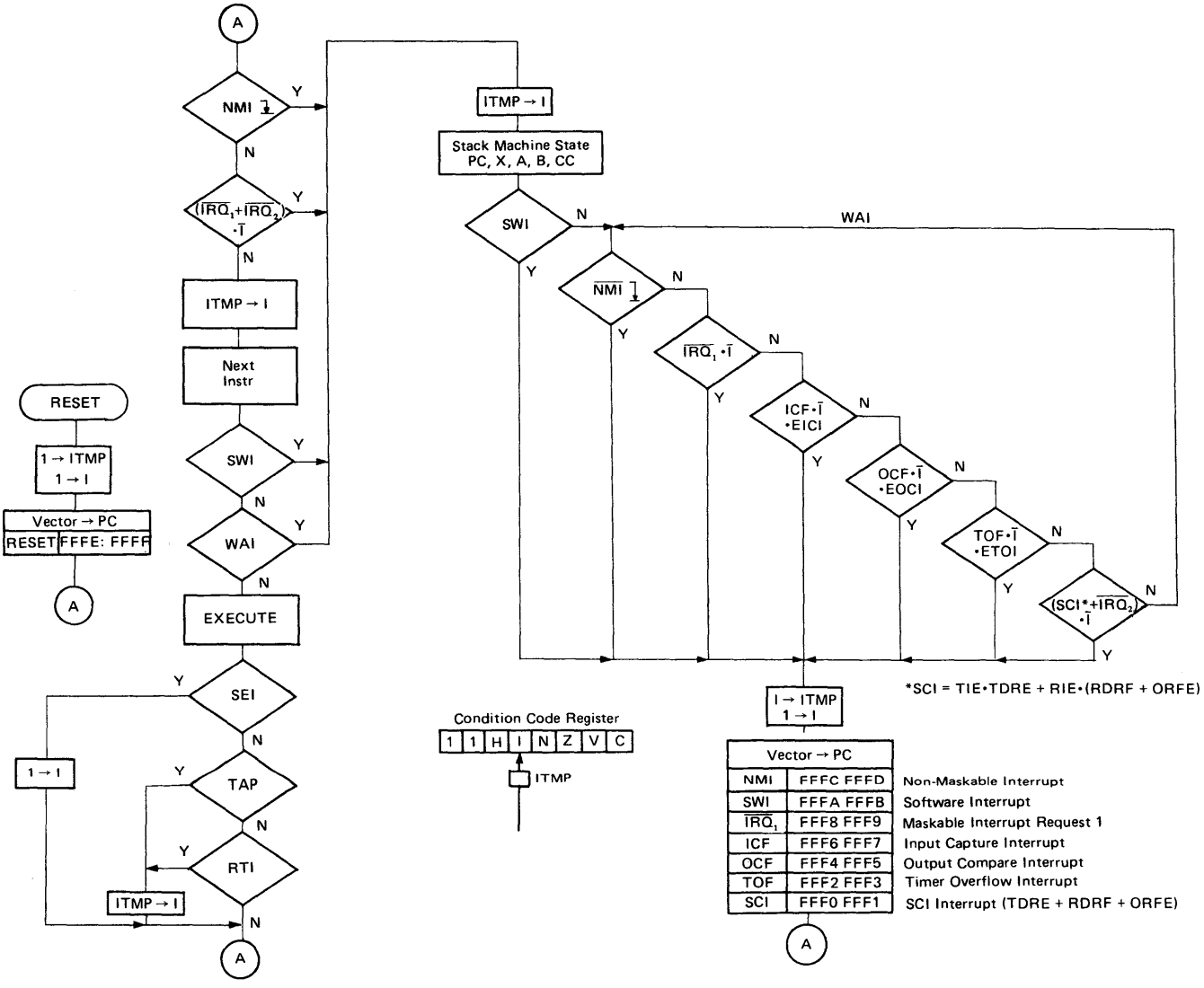


Figure 24 Interrupt Flowchart

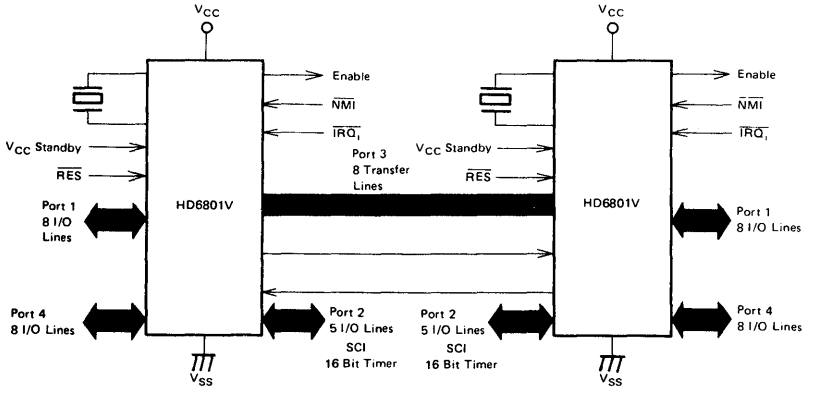


Figure 25 HD6801V MCU Single-Chip Dual Processor Configuration

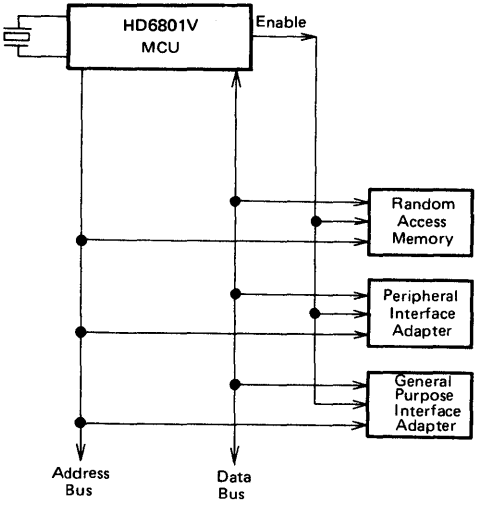


Figure 26 HD6801V MCU Expanded Non-Multiplexed Mode

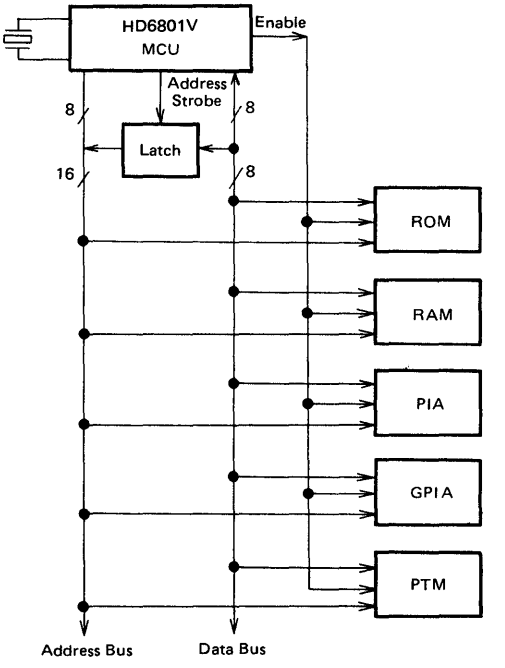


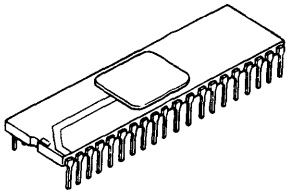
Figure 27 HD6801V MCU Expanded Multiplexed Mode

HD6803, HD6803-1

MPU (Micro Processing Unit)

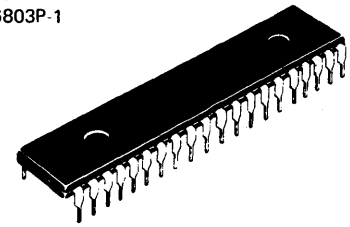
The HD6803 MPU is an 8-bit microcomputer system which is compatible with the HMCS6800 family of parts. The HD6803 MPU is object code compatible with the HD6800 with improved execution times of key instructions plus several new 16-bit and 8-bit instructions including an 8 x 8 unsigned multiply with 16-bit result. The HD6803 MPU can be expanded to 65k words. The HD6803 MPU is TTL compatible and requires one +5.0 volt power supply. The HD6803 MPU has 128 bytes of RAM, Serial Communications interface (S.C.I.), and parallel I/O as well as a three function 16-bit timer. Features and Block diagram of the HD6803 include the following:

HD6803C



(DC-40)

HD6803P
HD6803P-1

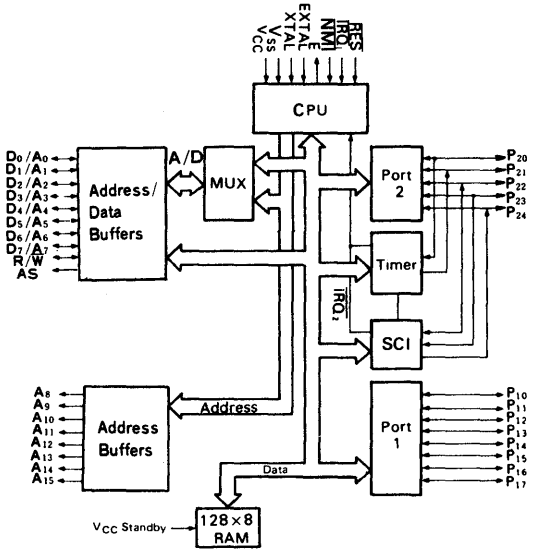


(DP-40)

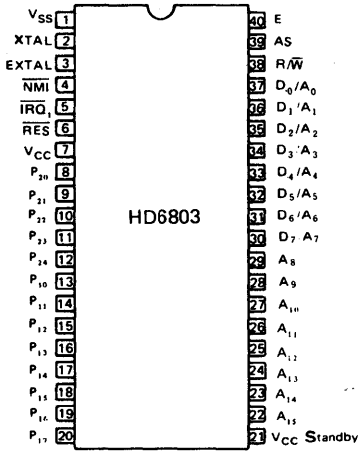
■ FEATURES

- Expanded HMCS6800 Instruction Set
- 8 x 8 Multiply
- On-Chip Serial Communications Interface (S.C.I.)
- Object Code Compatible With The HD6800 MPU
- 16-Bit Timer
- Expandable to 65k Words
- Multiplexed Address and Data
- 128 Bytes Of RAM (64 Bytes Retainable On Power Down)
- 13 Parallel I/O Lines
- Internal Clock/Divided-By-Four
- TTL Compatible Inputs And Outputs
- Interrupt Capability
- Compatible with MC6803

■ BLOCK DIAGRAM



■ PIN ARRANGEMENT



(Top View)

■ TYPE OF PRODUCTS

Type No.	Bus Timing
HD6803	1.0MHz
HD6803-1	1.25MHz

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC} *	-0.3 ~ +7.0	V
Input Voltage	V_{in} *	-0.3 ~ +7.0	V
Operating Temperature	T_{opr}	0 ~ +70	°C
Storage Temperature	T_{stg}	-55 ~ +150	°C

* With respect to V_{SS} (SYSTEM GND)

[NOTE] Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ ELECTRICAL CHARACTERISTICS

• DC CHARACTERISTICS (V_{CC} =5.0V±5%, V_{SS} = 0V, Ta = 0~+70°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Input "High" Voltage	$\overline{RES}$	V_{IH}		4.0	–	V_{CC}	V
	Other Inputs*			2.0	–	V_{CC}	
Input "Low" Voltage	All Inputs*	V_{IL}		-0.3	–	0.8	V
Input Load Current	EXTAL	$\|I_{in}\|$	V_{in} = 0 ~ V_{CC}	–	–	0.8	mA
Input Leakage Current	$\overline{NMI}$, $\overline{IRQ_1}$, $\overline{RES}$	$\|I_{in}\|$	V_{in} = 0 ~ 5.25V	–	–	2.5	μA
Three State (Offset) Leakage Current	P_{10} ~ P_{17}	$\|I_{TSI}\|$	V_{in} = 0.5 ~ 2.4V	–	–	10	μA
	P_{20} ~ P_{24}			–	–	100	
Output "High" Voltage	D_0/A_0 ~ D_7/A_7	V_{OH}	I_{LOAD} = -205 μA	2.4	–	–	V
	A_8 ~ A_{15}, E, R/$\overline{W}$, AS		I_{LOAD} = -145 μA	2.4	–	–	
	Other Outputs		I_{LOAD} = -100 μA	2.4	–	–	
Output "Low" Voltage	All Outputs	V_{OL}	I_{LOAD} = 1.6 mA	–	–	0.5	V
Darlington Drive Current	P_{10} ~ P_{17}	$-I_{OH}$	V_{out} = 1.5V	1.0	–	10.0	mA
Power Dissipation		P_D		–	–	1200	mW
Input Capacitance		C_{in}	V_{in} = 0V, Ta = 25°C, f = 1.0 MHz	–	–	10.0	pF
V_{CC} Standby	Powerdown	V_{SBB}		4.0	–	5.25	V
	Operating	V_{SB}		4.75	–	5.25	
Standby Current	Powerdown	I_{SBB}	V_{SBB} = 4.0V	–	–	8.0	mA

*Except Mode Programming Levels.

- **AC CHARACTERISTICS**

BUS TIMING (V_{CC} = 5.0V ± 5%, V_{SS} = 0V, Ta = 0 ~ +70°C, unless otherwise noted.)

Item		Symbol	Test Condition	HD6803 min	HD6803 typ	HD6803 max	HD6803-1 min	HD6803-1 typ	HD6803-1 max	Unit
Cycle Time		t_{cyc}	Fig. 1	1	–	10	0.8	–	10	μs
Address Strobe Pulse Width "High"		PW_{ASH}		200	–	–	150	–	–	ns
Address Strobe Rise Time		t_{ASr}		5	–	50	5	–	50	ns
Address Strobe Fall Time		t_{ASf}		5	–	50	5	–	50	ns
Address Strobe Delay Time		t_{ASD}		60	–	–	30	–	–	ns
Enable Rise Time		t_{Er}		5	–	50	5	–	50	ns
Enable Fall Time		t_{Ef}		5	–	50	5	–	50	ns
Enable Pulse Width "High" Time		PW_{EH}		450	–	–	340	–	–	ns
Enable Pulse Width "Low" Time		PW_{EL}		450	–	–	350	–	–	ns
Address Strobe to Enable Delay Time		t_{ASED}		60	–	–	30	–	–	ns
Address Delay Time		t_{AD}		–	–	260	–	–	260	ns
Address Delay Time for Latch		t_{ADL}		–	–	270	–	–	260	ns
Data Set-up Write Time		t_{DSW}		225	–	–	115	–	–	ns
Data Set-up Read Time		t_{DSR}		80	–	–	70	–	–	ns
Data Hold Time	Read	t_{HR}		10	–	–	10	–	–	ns
	Write	t_{HW}		20	–	–	20	–	–	
Address Set-up Time for Latch		t_{ASL}		60	–	–	50	–	–	ns
Address Hold Time for Latch		t_{AHL}		20	–	–	20	–	–	ns
Address Hold Time		t_{AH}		20	–	–	20	–	–	ns
Peripheral Read Access Time (Multiplexed Bus)		(t_{ACCM})		–	–	(600)	–	–	(410)	ns
Oscillator stabilization Time		t_{RC}	Fig. 7	100	–	–	100	–	–	ms
Processor Control Set-up Time		t_{PCS}	Fig. 8	200	–	–	200	–	–	ns

PERIPHERAL PORT TIMING (V_{CC} = 5.0V ± 5%, V_{SS} = 0V, Ta = 0 ~ +70°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Peripheral Data Setup Time	Port 1, 2	t_{PDSU}	Fig. 2	200	–	–	ns
Peripheral Data Hold Time	Port 1, 2	t_{PDH}	Fig. 2	200	–	–	ns
Delay Time, Enable Negative Transition to Peripheral Data Valid	Port 1, 2*	t_{PWD}	Fig. 3	–	–	400	ns

* Except P_{21}

TIMER, SCI TIMING (V_{CC} = 5.0V ±5%, V_{SS} = 0V, Ta = 0 ~ +70°C, unless otherwise noted.)

Item	Symbol	Test Condition	min	typ	max	Unit
Timer Input Pulse Width	t_{PWT}		$2t_{cyc}$+200	–	–	ns
Delay Time, Enable Positive Transition to Timer Out	t_{TOD}	Fig. 4	–	–	600	ns
SCI Input Clock Cycle	t_{Scyc}		1	–	–	t_{cyc}
SCI Input Clock Pulse Width	t_{PWSCK}		0.4	–	0.6	t_{Scyc}

MODE PROGRAMMING (V_{CC} = 5.0V ±5%, V_{SS} = 0V, Ta = 0 ~ +70°C, unless otherwise noted.)

Item	Symbol	Test Condition	min	typ	max	Unit
Mode Programming Input "Low" Voltage	V_{MPL}	Fig. 5	–	–	1.7	V
Mode Programming Input "High" Voltage	V_{MPH}		4.0	–	–	V
$\overline{RES}$ "Low" Pulse Width	PW_{RSTL}		3.0	–	–	t_{cyc}
Mode Programming Set-up Time	t_{MPS}		2.0	–	–	t_{cyc}
Mode Programming Hold Time	t_{MPH}		150	–	–	ns

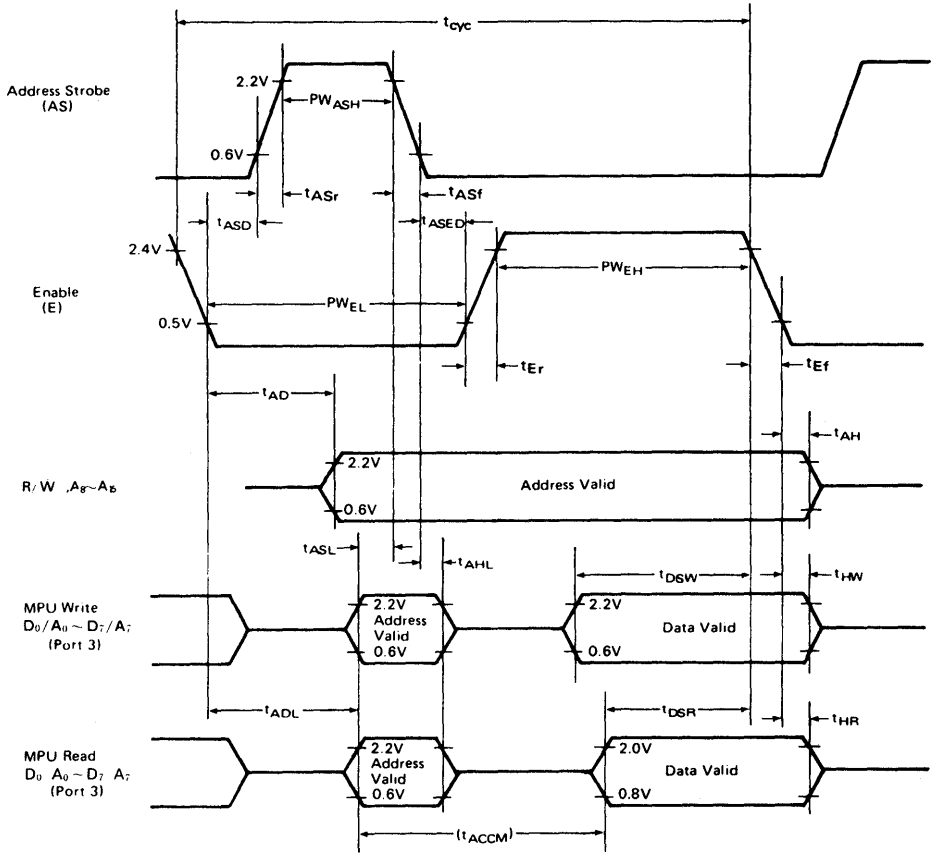


Figure 1 Expanded Multiplexed Bus Timing

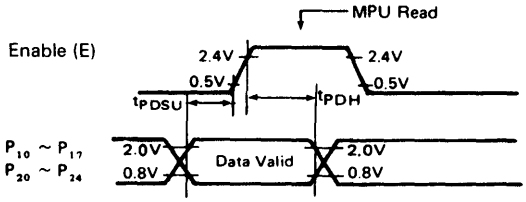


Figure 2 Data Set-up and Hold Times (MPU Read)

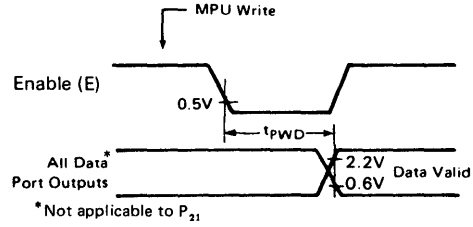


Figure 3 Port Data Delay Timing (MPU Write)

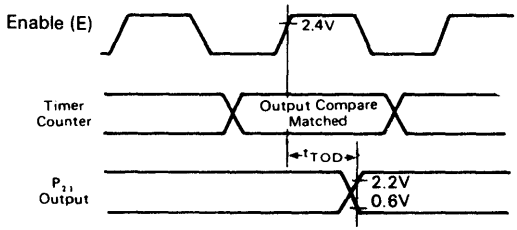


Figure 4 Timer Output Timing

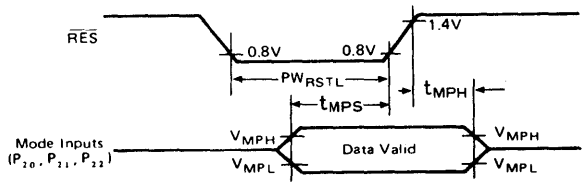
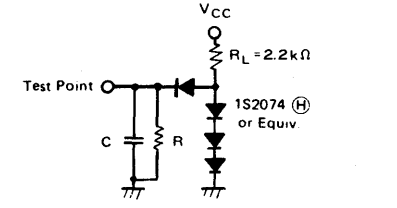


Figure 5 Mode Programming Timing



C = 90 pF for $D_0/A_0 \sim D_7/A_7$, $A_8 \sim A_{15}$, E, AS, R/$\overline{W}$
= 30 pF for $P_{10} \sim P_{17}$, $P_{20} \sim P_{24}$
R = 12 kΩ for $D_0/A_0 \sim D_7/A_7$, $A_8 \sim A_{15}$, E, AS, R/$\overline{W}$
= 24 kΩ for $P_{10} \sim P_{17}$, $P_{20} \sim P_{24}$

TTL Load

Figure 6 Bus Timing Test Loads

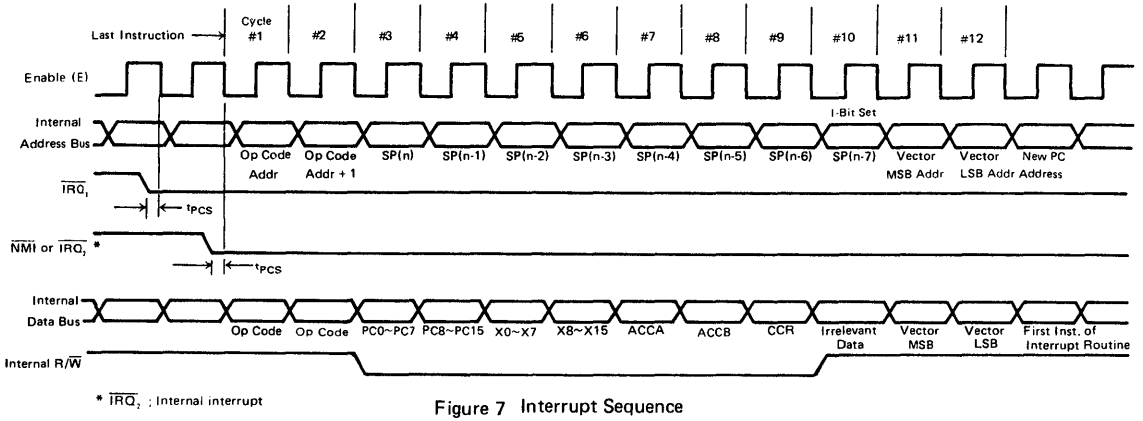


Figure 7 Interrupt Sequence

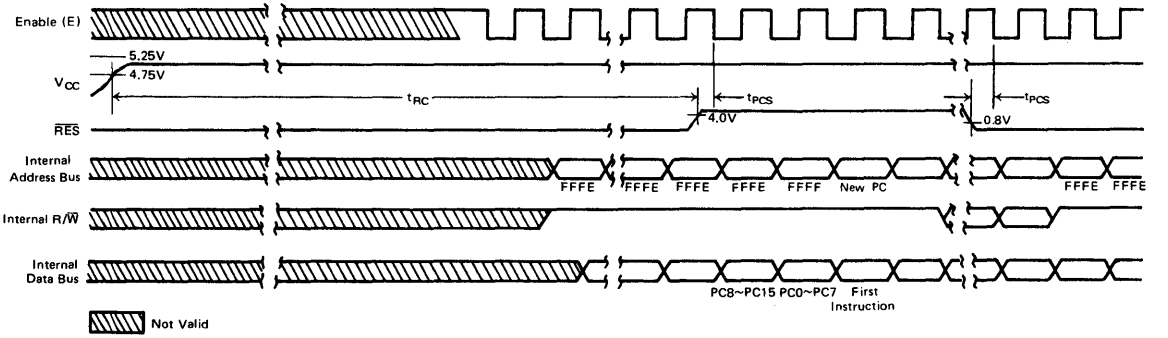


Figure 8 Reset Timing

■ SIGNAL DESCRIPTIONS

• VCC and VSS

These two pins are used to supply power and ground to the chip. The voltage supplied will be +5 volts ±5%.

• XTAL and EXTAL

These connections are for a parallel resonant fundamental crystal, AT cut. Devided by 4 circuitry is included with the internal clock, so a 4 MHz crystal may be used to run the system at 1 MHz. The devide by 4 circuitry allows for use of the inexpensive 3.58 MHz Color TV crystal for non-time critical applications. Two 22pF capacitors are needed from the two crystal pins to ground to insure reliable operation. EXTAL may be driven by an external TTL compatible source with a 50% (±10%) duty cycle. It will devided by 4 any frequency less than or equal to 5 MHz. XTAL must be grounded if an external clock is used. The following are the recommended crystal parameters:

Nominal Crystal Parameter

Item \ Crystal	4 MHz	5 MHz
C_O	7pF max.	4.7pF max.
R_S	60Ω max.	30Ω typ.

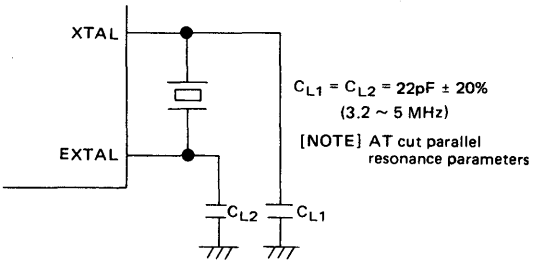


Figure 9 Crystal Interface

- **V_{CC} Standby**

This pin will supply +5 volts ±5% to the standby RAM on the chip. The first 64 bytes of RAM will be maintained in the power down mode with 8 mA current max. The circuit of figure 13 can be utilized to assure that V_{CC} Standby does not go below V_{SBB} during power down.

To retain information in the RAM during power down the following procedure is necessary:

1) Write "0" into the RAM enable bit, RAM E. RAM E is bit 6 of the RAM Control Register at location $0014. This disables the standby RAM, thereby protecting it at power down.
2) Keep V_{CC} Standby greater than V_{SBB}.

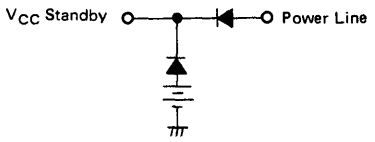


Figure 10 Battery Backup for V_{CC} Standby

- **Reset ($\overline{RES}$)**

This input is used to reset and start the MPU from a power down condition, resulting from a power failure or an initial startup of the processor. On power up, the reset must be held "Low" for at least 100 ms. During operation, $\overline{RES}$, when brought "Low", must be held "Low" at least 3 clock cycles.

When a "High" level is detected, the MPU does the following:

1) All the higher order address lines will be forced "High".
2) I/O Port 2 bits, 2, 1, and 0 are latched into programmed control bits PC2, PC1 and PC0.
3) The last two ($FFFE, $FFFF) locations in memory will be used to load the program addressed by the program counter.
4) The interrupt mask bit is set, must be cleared before the MPU can recognize maskable interrupts.

- **Enable (E)**

This supplies the external clock for the rest of the system when the internal oscillator is used. It is a single phase, TTL compatible clock, and will be the divide by 4 result of the crystal frequency. It will drive one TTL load and 90 pF.

- **Non-Maskable Interrupt ($\overline{NMI}$)**

A low-going edge on this input requests that a non-maskable-interrupt sequence be generated within the processor. As with interrupt Request signal, the processor will complete the current instruction that is being executed before it recognizes the $\overline{NMI}$ signal. The interrupt mask bit in the Condition Code Register has no effect on $\overline{NMI}$.

In response to an $\overline{NMI}$ interrupt, the Index Register, Program Counter, Accumulators, and Condition Code Register are stored on the stack. At the end of the sequence, a 16-bit address will be loaded that points to a vectoring address located in memory locations $FFFC and $FFFD. An address loaded at these locations causes the MPU to branch to a non-maskable interrupt service routine in memory.

A 3.3 kΩ external resistor to V_{CC} should be used for wire-OR and optimum control of interrupts.

Inputs $\overline{IRQ_1}$ and $\overline{NMI}$ are hardware interrupt lines that are sampled during E and will start the interrupt routine on the $\overline{E}$ following the completion of an instruction.

- **Interrupt Request ($\overline{IRQ_1}$)**

This level sensitive input requests that an interrupt sequence be generated within the machine. The processor will wait until it completes the current instruction that it being executed before it recognizes the request. At that time, if the interrupt mask bit in the Condition Code Register is not set, the machine will begin an interrupt sequence. The Index Register, Program Counter, Accumulators, and Condition Code Register are stored on the stack. Next the MPU will respond to the interrupt request by setting the interrupt mask bit "High" so that no further maskable interrupts may occur. At the end of the cycle, a 16-bit address will be loaded that points to a vectoring address which is located in memory locations $FFF8 and $FFF9. An address loaded at these locations causes the MPU to branch to an interrupt routine in memory.

The $\overline{IRQ_1}$ requires a 3.3 kΩ external resister to V_{CC} which should be used for wire-OR and optimum control of interrupts. Internal Interrupts will use an internal interrupt line ($\overline{IRQ_2}$). This interrupt will operate the same as $\overline{IRQ_1}$ except that it will use the vector address of $FFF0 through $FFF7. $\overline{IRQ_1}$ will have priority over $\overline{IRQ_2}$ if both occur at the same time. The Interrupt Mask Bit in the condition code register masks both interrupts (See Table 1).

Table 1 Interrupt Vector Location

	Vector		Interrupt
	MSB	LSB	
Highest Priority	FFFE	FFFF	$\overline{RES}$
	FFFC	FFFD	$\overline{NMI}$
	FFFA	FFFB	Software Interrupt (SWI)
	FFF8	FFF9	$\overline{IRQ_1}$
	FFF6	FFF7	ICF (Input Capture)
	FFF4	FFF5	OCF (Output Compare)
	FFF2	FFF3	TOF (Timer Overflow)
Lowest Priority	FFF0	FFF1	SCI (RDRF + ORFE + TDRE)

- **Read/Write (R/$\overline{W}$)**

This TTL compatible output signals the peripherals and memory devices whether the MPU is in a Read ("High") or a Write ("Low") state. The normal standby state of this signal is Read ("High"). This output is capable of driving one TTL load and 90 pF.

- **Address Strobe (AS)**

In the expanded multiplexed mode of operation address strobe is output on this pin. This signal is used to latch the 8 LSB's of address which are multiplexed with data on Port 3. An 8-bit latch is utilized in conjunction with Address Strobe, as shown in figure 11. Expanded Multiplexed Mode. Address Strobe signals the latch when it is time to latch the address lines so the lines can become data bus lines during the E pulse. The timing for this singal is shown in Figure 1 of Bus Timing. This signal is also used to disable the address from the multiplexed bus allowing a deselect time, t_{ASD} before the data is enabled to the bus.

- **PORTS**

There are two I/O ports on the HD6803 MPU; one 8-bit port and one 5-bit port. Each port has an associated write

only Data Direction Register which allows each I/O line to be programmed to act as an input or an output*. A "1" in the corresponding Data Direction Register bit will cause that I/O line to be an output. A "0" in the corresponding Data Direction Register bit will cause that I/O line to be an input. There are two ports: Port 1, Port 2. Their addresses and the addresses of their Data Direction registers are given in Table 2.

* The only exception is bit 1 of Port 2, which can either be data input or Timer output.

Table 2 Port and Data Direction Register Addresses

Ports	Port Address	Data Direction Register Address
I/O Port 1	$0002	$0000
I/O Port 2	$0003	$0001

- **I/O Port 1**

This is an 8-bit port whose individual bits may be defined as inputs or outputs by the corresponding bit in its data direction register. The 8 output buffers have three-state capability, allowing them to enter a high impedance state when the peripheral data lines are used as inputs. In order to be read properly, the voltage on the input lines must be greater than 2.0 V for a logic "1" and less than 0.8 V for a logic "0". As outputs, these lines are TTL compatible and may also be used as a source of up to 1 mA at 1.5 V to directly drive a Darlington base. After Reset, the I/O lines are configured as inputs.

- **I/O Port 2**

This port has five lines that may be defined as inputs or outputs by its data direction register. The 5 output buffers have three-state capability, allowing them to enter a high impedance state when used as an input. In order to be read properly, the voltage on the input lines must be greater than 2.0 V for a logic "1" and less than 0.8 V for a logic "0". As outputs, this port has no internal pullup resistors but will drive TTL inputs directly. For driving CMOS inputs, external pullup resistors are required. After Reset, the I/O lines are configured as inputs. Three pins on Port 2 (pin 8, 9 and 10 of the chip) are requested to set following values (Table 3) during reset. The values of above three pins during reset are latched into the three MSBs (Bit 5, 6 and 7) of Port 2 which are read only.

Port 2 can be configured as I/O and provides access to the Serial Communications Interface and the Timer. Bit 1 is the only pin restricted to data input or Timer output.

Table 3 The Values of three pins

Pin Number	Value
8	L
9	H
10	L

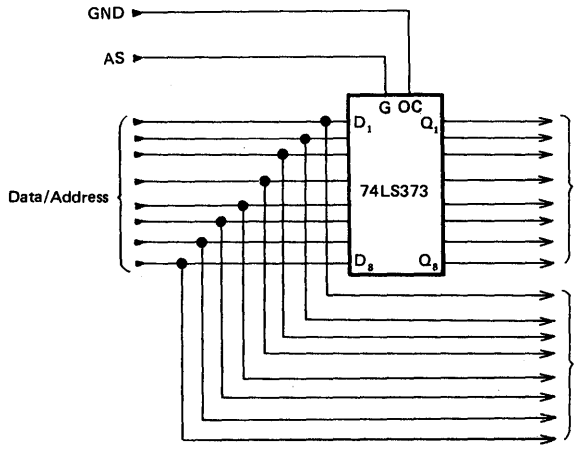
[NOTES] L; Logical "0"
H; Logical "1"

- **Data/Address (Lower Order Address Bus Latches)**

Since the data bus is multiplexed with the lower order address bus in Data/Address, latches are required to latch those address bits. The 74LS373 Transparent Octal D-type latch can be used with the HD6803 to latch the least significant address byte. Figure 11 shows how to connect the latch to the HD6803. The output control to the 74LS373 may be connected to ground.

■ INTERRUPT FLOWCHART

The Interrupt flowchart is depicted in Figure 16 and is common to every interrupt excluding reset.



Function Table

Output Control	Enable G	Enable D	Output Q
L	H	H	H
L	H	L	L
L	L	X	Q_0
H	X	X	Z

Figure 11 Latch Connection

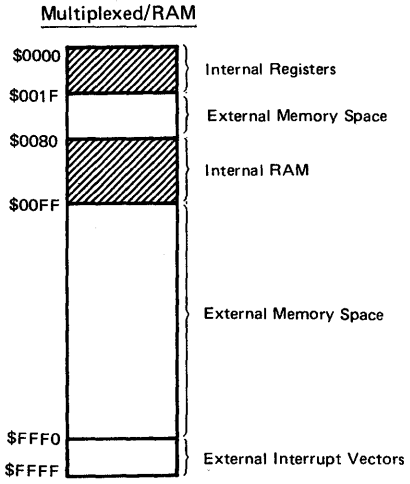
■ MEMORY MAP

The MPU can provide up to 65k byte address space. A memory map is shown in Figure 12. The first 32 locations are reserved for the MPU's internal register area, as shown in Table 4 with exceptions as indicated.

Table 4 Internal Register Area

Register	Address
Port 1 Data Direction Register**	00
Port 2 Data Direction Register**	01
Port 1 Data Register	02
Port 2 Data Register	03
Not Used	04*
Not Used	05*
Not Used	06*
Not Used	07*
Timer Control and Status Register	08
Counter (High Byte)	09
Counter (Low Byte)	0A
Output Compare Register (High Byte)	0B
Output Compare Register (Low Byte)	0C
Input Capture Register (High Byte)	0D
Input Capture Register (Low Byte)	0E
Not Used	0F*
Rate and Mode Control Register	10
Transmit/Receive Control and Status Register	11
Receive Data Register	12
Transmit Data Register	13
RAM Control Register	14
Reserved	15-1F

* External Address
** 1; Output, 0; Input



[NOTE]
Excludes the following addresses which may be used externally: $04, $05, $06, $07, and $0F.

Figure 12 HD6803 Memory Map

■ PROGRAMMABLE TIMER

The HD6803 contains an on-chip 16-bit programmable timer which may be used to perform measurements on an input waveform while independently generating an output waveform. Pulse widths for both input and output signals may vary from a few microseconds to many seconds. The timer hardware consists of

- an 8-bit control and status register,
- a 16-bit free running counter,
- a 16-bit output compare register, and
- a 16-bit input capture register

A block diagram of the timer registers is shown in Figure 13.

● Free Running Counter ($0009:000A)

The key element in the programmable timer is a 16-bit free running counter which is driven to increasing values by E (Enable). The counter value may be read by the MPU software at any time. The counter is cleared to zero on $\overline{RES}$ and may be considered a read-only register with one exception. Any MPU write to the counter's address ($09) will always result in preset value of $FFF8 being loaded into the counter regardless of the value involved in the write. This preset figure is intended for testing operation of the part, but may be of value in some applications.

● Output Compare Register ($000B:000C)

The Output Compare Register is a 16-bit read/write register which is used to control an output waveform. The contents of this register are constantly compared with the current value of the free running counter. When a match is found, a flag is set (OCF) in the Timer Control and Status Register (TCSR) and the current value of the Output Level bit (OLVL) in the TCSR is clocked to the Output Level Register. Providing the Data Direction Register for Port 2, Bit 1 contains a "1" (Output), the output level register value will appear on the pin for Port 2 Bit 1. The values in the Output Compare Register and Output Level bit may then be changed to control the output level on the next compare value. The Output Compare Register is set to $FFFF during $\overline{RES}$. The Compare function is inhibited for one cycle following a write to the high byte of the Output Compare Register to insure a valid 16-bit value is in the register before a compare is made.

● Input Capture Register ($000D:000E)

The Input Capture Register is a 16-bit read-only register used to store the current value of the free running counter when the proper transition of an external input signal occurs. The input transition change required to trigger the counter transfer is controlled by the input Edge bit (IEDG) in the TCSR. The Data Direction Register bit for Port 2 Bit 0, should* be clear (zero) in order to gate in the external input signal to the edge detect unit in the timer.

* With Port 2 Bit 0 configured as an output and set to "1", the external input will still be seen by the edge detect unit.

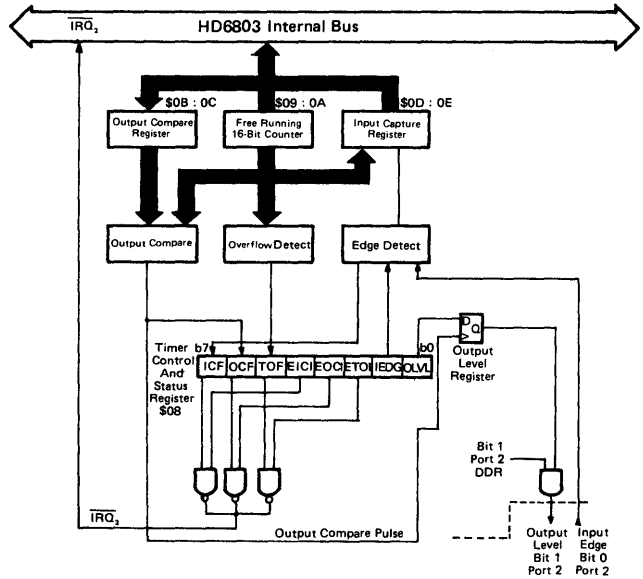


Figure 13 Block Diagram of Programmable Timer

Timer Control and Status Register

7	6	5	4	3	2	1	0	
ICF	OCF	TOF	EICI	EOCI	ETOI	IEDG	OLVL	$0008

- **Timer Control and Status Register (TCSR) ($0008)**

The Timer Control and Status Register consists of an 8-bit register of which all 8 bits are readable but only the low order 5 bits may be written. The upper three bits contain read-only timer status information and indicate that:

- a proper transition has taken place on the input pin with a subsequent transfer of the current counter value to the input capture register.
- a match has been found between the value in the free running counter and the output compare register, and when $0000 is in the free running counter.

Each of the flags may be enabled onto the HD6803 internal bus ($\overline{IRQ_2}$) with an individual Enable bit in the TCSR. If the I-bit in the HD6803 Condition Code register has been cleared, a priority vectored interrupt will occur corresponding to the flag bit(s) set. A description for each bit follows:

Bit 0 **OLVL** Output Level – This value is clocked to the output level register on a successful output compare. If the DDR for Port 2 bit 1 is set, the value will appear on the output pin.

Bit 1 **IEDG** Input Edge – This bit controls which transition of an input will trigger a transfer of the counter to the input capture register. The DDR for Port 2 Bit 0 must be clear for this function to operate. IEDG = 0 Transfer takes place on a negative edge ("High"-to-"Low" transition).
IEDG = 1 Transfer takes place on a positive edge ("Low"-to-"High" transition).

Bit 2 **ETOI** Enable Timer Overflow Interrupt – When set, this bit enables $\overline{IRQ_2}$ to occur on the internal bus for a TOF interrupt; when clear the interrupt is inhibited.

Bit 3 **EOCI** Enable Output Compare Interrupt – When set, this bit enables $\overline{IRQ_2}$ to appear on the internal bus for an output compare interrupt; when clear the interrupt is inhibited.

Bit 4 **EICI** Enable input Capture Interrupt – When set, this bit enables $\overline{IRQ_2}$ to occur on the internal bus for an input capture interrupt; when clear the interrupt is inhibited.

Bit 5 **TOF** Timer Overflow Flag – This read-only bit is set when the counter contains $0000. It is cleared by a read of the TCSR (with TOF set) followed by an MPU read of the Counter ($09).

Bit 6 **OCF** Output Compare Flag – This read-only bit is set when a match is found between the output compare register and the free running counter. It is cleared by a read of the TCSR (with OCF set) followed by an MPU write to the output compare register ($0B or $0C).

Bit 7 **ICF** Input Capture Flag – This read-only status bit is set by a proper transition on the input; it is cleared by a read of the TCSR (with ICF set) followed by an MPU read of the Input Capture Register ($0D).

■ SERIAL COMMUNICATIONS INTERFACE

The HD6803 contains a full-duplex asynchronous serial communications interface (SCI) on chip. The controller comprises a transmitter and a receiver which operate independently or each other but in the same data format and at the same data rate. Both transmitter and receiver communicate with the MPU via the data bus and with the outside world via pins 2, 3, and 4 of Port 2. The hardware, software, and registers are explained in the following paragraphs.

• Wake-Up Feature

In a typical multi-processor application, the software protocol will usually contain a destination address in the initial byte(s) of the message. In order to permit non-selected MPU's to ignore the remainder of the message, a wake-up feature is included whereby all further interrupt processing may be optionally inhibited until the beginning of the next message. When the next message appears, the hardware re-enables (or "wakes-up") for the next message. The "wake-up" is automatically triggered by a string of ten consecutive 1's which indicates an idle transmit line. The software protocol must provide for the short idle period between any two consecutive messages.

• Programmable Options

The following features of the HD6803 serial I/O section are programmable:

- format – standard mark/space (NRZ)
- Clock – external or internal
- baud rate – one of 4 per given MPU ϕ_2 clock frequency or external clock ×8 input
- wake-up feature – enabled or disabled
- Interrupt requests – enabled or masked individually for transmitter and receiver data registers
- clock output – internal clock enabled or disabled to Port 2 (Bit 2)
- Port 2 (bits 3 and 4) – dedicated or not dedicated to serial I/O individually for transmitter and receiver.

• Serial Communications Hardware

The serial communications hardware is controlled by 4 registers as shown in Figure 14. The registers include:

- an 8-bit control and status register
- a 4-bit rate and mode control register (write only)
- an 8-bit read only receive data register and
- an 8-bit write only transmit data register.

In addition to the four registers, the serial I/O section utilizes bit 3 (serial input) and bit 4 (serial output) of Port 2. Bit 2 of Port 2 is utilized if the internal-clock-out or external-clock-in options are selected.

Transmit/Receive Control and Status (TRCS) Register

The TRCS register consists of an 8-bit register of which all 8 bits may be read while only bits 0~4 may be written. The register is initialized to $20 on $\overline{RES}$. The bits in the TRCS register are defined as follows:

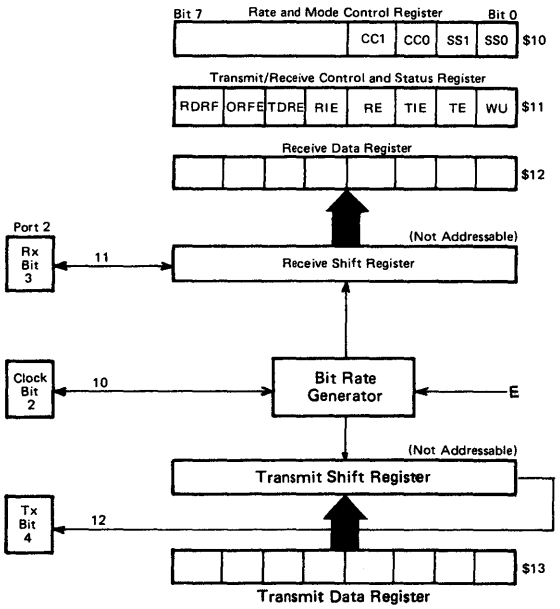


Figure 14 Serial I/O Registers

Bit 0 **WU** "Wake-up" on Next Message – set by HD6803 software and cleared by hardware on receipt of ten consecutive 1's or reset of RE flag. It should be noted that RE flag should be set in advance of MPU set of WU flag.

Bit 1 **TE** Transmit Enable – set by HD6803 to produce preamble of nine consecutive 1's and to enable gating of transmitter output to Port 2, bit 4 regardless of the DDR value corresponding to this bit; when clear, serial I/O has no effect on Port 2 bit 4.

TE set should be after at least one bit time of data transmit rate from the set-up of transmit data rate and mode.

Bit 2 **TIE** Transmit Interrupt Enable – when set, will permit an $\overline{IRQ_2}$ interrupt to occur when bit 5 (TDRE) is set; when clear, the TDRE value is masked from the bus.

Bit 3 **RE** Receiver Enable – when set, gates Port 2 bit 3 to input of receiver regardless of DDR value for this bit; when clear, serial I/O has no effect on Port 2 bit 3.

Bit 4 **RIE** Receiver Interrupt Enable – when set, will permit an $\overline{IRQ_2}$ interrupt to occur when bit 7 (RDRF) or bit 6 (ORFE) is set; when clear, the interrupt is masked.

Transmit/Receive Control and Status Register

7	6	5	4	3	2	1	0	
RDRF	ORFE	TDRE	RIE	RE	TIE	TE	WU	ADDR : $0011

Bit 5 **TDRE** Transmit Data Register Empty – set by hardware when a transfer is made from the transmit data register to the output shift register. The TDRE bit is cleared by reading the status register, then writing a new byte into the transmit data register, TDRE is initialized to 1 by $\overline{RES}$.

Bit 6 **ORFE** Over-Run-Framing Error – set by hardware when an overrun or framing error occurs (receive only)

Rate and Mode Control Register

7	6	5	4	3	2	1	0
X	X	X	X	CC1	CC0	SS1	SS0

ADDR : $0010

An overrun is defined as a new byte received with last byte still in Dat Register/Buffer. A framing error has occured when the byte boundaries in bit stream are not synchronized to bit counter. If WU-flag is set, the ORFE bit will not be set. The ORFE bit is cleard by reading the status register, then reading the Receive Data Register, or by $\overline{RES}$.

Bit 7 **RDRF** Receiver Data Register Full-set by hardware when a transfer from the input shift register to the receiver data register is made. If WU-flag is set, the RDRF bit will not be set. The RDRF bit is cleared by reading the status register, then reading the Receive Data Register, or by $\overline{RES}$.

Rate and Mode Control Register

The Rate and Mode Control register controls the following serial I/O variables:

- Baud rate
- format
- clocking source, and
- Port 2 bit 2 configuration

The register consists of 4 bits all of which are write-only and cleared on $\overline{RES}$. The 4 bits in the register may be considered as a pair of 2-bit fields. The two low order bits control the bit rate for internal clocking and the remaining two bits control the format and clock select logic. The register definition is as follows:

Bit 0 **SS0** Speed Select – These bits select the Baud rate for
Bit 1 **SS1** the internal clock. The four rates which may be selected are a function of the MPU ϕ_2 clock frequency. Table 5 lists the available Baud rates.

Bit 2 **CC0** Clock Control and Format Select – this 2-bit field
Bit 3 **CC1** controls the format and clock select logic. Table 6 defines the bit field.

Table 5 SCI Bit Times and Rates

SS1 : SS0	XTAL E	2.4576 MHz 614.4 kHz	4.0 MHz 1.0 MHz	4.9152 MHz* 1.2288 MHz
0 0	E ÷ 16	26 μs/38,400 Baud	16 μs/62,500 Baud	13.0 μs/76,800 Baud
0 1	E ÷ 128	208 μs/4,800 Baud	128 μs/7812.5 Baud	104.2 μs/9,600 Baud
1 0	E ÷ 1024	1.67 ms/600 Baud	1.024 ms/976.6 Baud	833.3 μs/1,200 Baud
1 1	E ÷ 4096	6.67 ms/150 Baud	4.096 ms/244.1 Baud	3.33 ms/300 Baud

* HD6803-1 Only

Table 6 SCI Format and Clock Source Control

CC1, CC0	Format	Clock Source	Port 2 Bit 2	Port 2 Bit 3	Port 2 Bit 4
00	–	–	–	**	**
01	NRZ	Internal	Not Used	**	**
10	NRZ	Internal	Output*	**	**
11	NRZ	External	Input	**	**

* Clock output is available regardless of values for bits RE and TE.
** Bit 3 is used for serial input if RE = "1" in TRCS; bit 4 is used for serial output if TE = "1" in TRCS.

Internally Generated Clock

If the user wishes for the serial I/O to furnish a clock, the following requirements are applicable:

- the values of RE and TE are immaterial.
- CC1, CC0 must be set to 10
- the maximum clock rate will be E ÷ 16.
- the clock will be at 1× the bit rate and will have a rising edge at mid-bit.

Externally Generated Clock

If the user wishes to provide an external clock for the serial I/O, the following requirements are applicable:

- the CC1, CC0, field in the Rate and Mode Control Register must be set to 11,
- the external clock must be set to 8 times (×8) the desired baud rate and
- the maximum external clock frequency is 1.0 MHz.

- **Serial Operations**

The serial I/O hardware should be initialized by the HD6803 software prior to operation. This sequence will normally consist of;

- writing the desired operation control bits to the Rate and Mode Control Register and
- writing the desired operational control bits in the Transmit/Receive Control and Status Register.

The Transmitter Enable (TE) and Receiver Enable (RE) bits may be left set for dedicated operations.

Transmit Operations

The transmit operation is enabled by the TE bit in the Transmit/Receive Control and Status Register. This bit when set, gates the output of the serial transmit shift register to Port 2 Bit 4 and takes unconditional control over the Data Direction Register value for Port 2, Bit 4.

Following a $\overline{RES}$ the user should configure both the Rate and Mode Control Register and the Transmit/Receive Control and Status Register for desired operation. Setting the TE bit during this procedure initiates the serial output by first transmitting a ten-bit preamble of 1's. Following the preamble, internal synchronization is established and the transmitter section is ready for operation.

At this point one of two situation exist:

1) if the Transmit Data Register is empty (TDRE = 1), a continuous string of ones will be sent indicating an idle line, or,
2) if data has been loaded into the Transmit Data Register (TDRE = 0), the word is transferred to the output shift register and transmission of the data word will begin.

During the transfer itself, the 0 start bit is first transmitted. Then the 8 data bits (beginning with bit 0) followed by the stop bit, are transmitted. When the Transmitter Data Register has been emptied, the hardware sets the TDRE flag bit.

If the HD6803 fails to respond to the flag within the proper time, (TDRE is still set when the next normal transfer from the parallel data register to the serial output register should occur) then a 1 will be sent (instead of a 0) at "Start" bit time, followed by more 1's until more data is supplied to the data register. No 0's will be sent while TDRE remains a 1.

Receive Operation

The receive operation is enabled by the RE bit which gates in the serial input through Port 2, Bit 3. The receiver section operation is conditioned by the contents of the Transmit/Receive Control and Status Register and the Rate and Mode Control Register.

The receiver bit interval is divided into 8 sub-intervals for internal synchronization. In the NRZ Mode, the received bit stream is synchronized by the first 0 (space) encountered.

The approximate center of each bit time is strobed during the next 10 bits. If the tenth bit is not a 1 (stop bit) a framing error is assumed, and bit ORFE is set. If the tenth bit as a 1, the data is transferred to the Receive Data Register, and interrupt flag RDRF is set. If RDRF is still set at the next tenth bit time, ORFE will be set, indicating an overrun has occurred. When the HD6803 responds to either flag (RDRF or ORFE) by reading the status register followed by reading the Data Register, RDRF (or ORFE) will be cleared.

■ RAM CONTROL REGISTER

This register, which is addressed at $0014, gives status information about the standby RAM. A 0 in the RAM enable bit (RAM E) will disable the standby RAM, thereby protecting it at power down if V_{CC} Standby is held greater than V_{SBB} volts, as explained previously in the signal description for V_{CC} Standby.

RAM Control Register

$0014	STBY PWR	RAME	X	X	X	X	X	X

Bit 0 Not used.
Bit 1 Not used.
Bit 2 Not used.
Bit 3 Not used.
Bit 4 Not used.
Bit 5 Not used.
Bit 6 **RAME** The RAM Enable control bit allows the user the ability to disable the standby RAM. This bit is set to a logic "1" by $\overline{RES}$ which enables the standby RAM and can be written to one or zero under program control. When the RAM is disabled, data is read from external memory.
Bit 7 **STBY PWR** The Standby Power bit is cleared when the standby voltage is removed. This bit is a read/write status flag that the user can read which indicates that the standby RAM voltage has been applied, and the data in the standby RAM is valid.

■ GENERAL DESCRIPTION OF INSTRUCTION SET

The HD6803 is upward object code compatible with the HD6800 as it implements the full HMCS6800 instruction set. The execution times of key instructions have been reduced to increase throughout. In addition, new instructions have been added; these include 16-bit operations and a hardware multiply.

Included in the instruction set section are the following:

- MPU Programming Model (Figure 15)
- Addressing modes
- Accumulator and memory instructions – Table 7
- New instructions
- Index register and stack manipulations instructions – Table 8
- Jump and branch instructions – Table 9
- Condition code register manipulation instructions – Table 10
- Instructions Execution times in machine cycles – Table 11
- Summary of cycle by cycle operation – Table 12
- Summary of undefined instructions – Table 13

- **MPU Programming Model**

The programming model for the HD6803 is shown in Figure 15. The double (D) accumulator is physically the same as the Accumulator A concatenated with the Accumulator B so that any operation using accumulator D will destroy information in A and B.

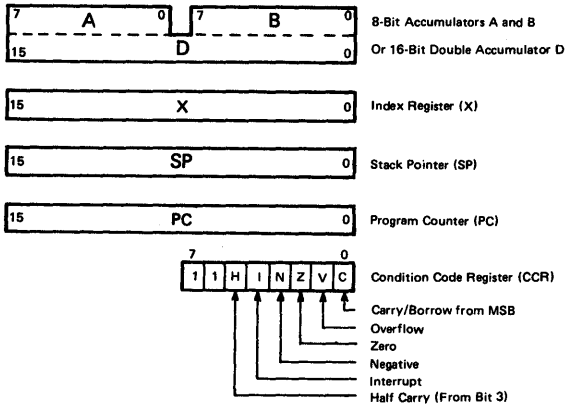


Figure 15 MPU Programming Model

- **MPU Addressing Modes**

The HD6803 8-bit micro processing unit has seven address modes that can be used by a programmer, with the addressing mode a function of both the type of instruction and the coding within the instruction. A summary of the addressing modes for a particular instruction can be found in Table 11 along with the associated instruction execution time that is given in machine cycles. With a clock frequency of 4 MHz, these times would be microseconds.

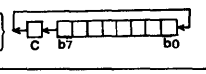
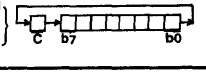
Accumulator (ACCX) Addressing

In accumulator only addressing, either accumulator A or accumulator B is specified. These are one-byte instructions.

Immediate Addressing

In immediate addressing, the operand is contained in the second byte of the instruction except LDS and LDX which have the operand in the second and third bytes of the instruction. The MPU addresses this location when it fetches the immediate instruction for execution. These are two or three-byte instructions.

Table 7 Accumulator & Memory Instructions

Operations	Mnemonic	Addressing Modes														Boolean/ Arithmetic Operation	Condition Code Register					
		IMMED.			DIRECT			INDEX			EXTEND			IMPLIED				5	4	3	2	1	0
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#		H	I	N	Z	V	C
Add	ADDA	8B	2	2	9B	3	2	AB	4	2	BB	4	3				A + M → A	‡	•	‡	‡	‡	‡
	ADDB	CB	2	2	DB	3	2	EB	4	2	FB	4	3				B + M → B	‡	•	‡	‡	‡	‡
Add Double	ADDD	C3	4	3	D3	5	2	E3	6	2	F3	6	3				A : B + M : M + 1 → A : B	•	•	‡	‡	‡	‡
Add Accumulators	ABA													1B	2	1	A + B → A	‡	•	‡	‡	‡	‡
Add With Carry	ADCA	89	2	2	99	3	2	A9	4	2	B9	4	3				A + M + C → A	‡	•	‡	‡	‡	‡
	ADCB	C9	2	2	D9	3	2	E9	4	2	F9	4	3				B + M + C → B	‡	•	‡	‡	‡	‡
AND	ANDA	84	2	2	94	3	2	A4	4	2	B4	4	3				A·M → A	•	•	‡	‡	R	•
	ANDB	C4	2	2	D4	3	2	E4	4	2	F4	4	3				B·M → B	•	•	‡	‡	R	•
Bit Test	BIT A	85	2	2	95	3	2	A5	4	2	B5	4	3				A·M	•	•	‡	‡	R	•
	BIT B	C5	2	2	D5	3	2	E5	4	2	F5	4	3				B·M	•	•	‡	‡	R	•
Clear	CLR							6F	6	2	7F	6	3				00 → M	•	•	R	S	R	R
	CLRA													4F	2	1	00 → A	•	•	R	S	R	R
	CLRB													5F	2	1	00 → B	•	•	R	S	R	R
Compare	CMPA	81	2	2	91	3	2	A1	4	2	B1	4	3				A − M	•	•	‡	‡	‡	‡
	CMPB	C1	2	2	D1	3	2	E1	4	2	F1	4	3				B − M	•	•	‡	‡	‡	‡
Compare Accumulators	CBA													11	2	1	A − B	•	•	‡	‡	‡	‡
Complement, 1's	COM							63	6	2	73	6	3				$\overline{M} \rightarrow M$	•	•	‡	‡	R	S
	COMA													43	2	1	$\overline{A} \rightarrow A$	•	•	‡	‡	R	S
	COMB													53	2	1	$\overline{B} \rightarrow B$	•	•	‡	‡	R	S
Complement, 2's (Negate)	NEG							60	6	2	70	6	3				00 − M → M	•	•	‡	‡	①	②
	NEGA													40	2	1	00 − A → A	•	•	‡	‡	①	②
	NEGB													50	2	1	00 − B → B	•	•	‡	‡	①	②
Decimal Adjust, A	DAA													19	2	1	Converts binary add of BCD characters into BCD format	•	•	‡	‡	‡	③
Decrement	DEC							6A	6	2	7A	6	3				M − 1 → M	•	•	‡	‡	④	•
	DECA													4A	2	1	A − 1 → A	•	•	‡	‡	④	•
	DECB													5A	2	1	B − 1 → B	•	•	‡	‡	④	•
Exclusive OR	EORA	88	2	2	98	3	2	A8	4	2	B8	4	3				A ⊕ M → A	•	•	‡	‡	R	•
	EORB	C8	2	2	D8	3	2	E8	4	2	F8	4	3				B ⊕ M → B	•	•	‡	‡	R	•
Increment	INC							6C	6	2	7C	6	3				M + 1 → M	•	•	‡	‡	⑤	•
	INCA													4C	2	1	A + 1 → A	•	•	‡	‡	⑤	•
	INCB													5C	2	1	B + 1 → B	•	•	‡	‡	⑤	•
Load Accumulator	LDAA	86	2	2	96	3	2	A6	4	2	B6	4	3				M → A	•	•	‡	‡	R	•
	LDAB	C6	2	2	D6	3	2	E6	4	2	F6	4	3				M → B	•	•	‡	‡	R	•
Load Double Accumulator	LDD	CC	3	3	DC	4	2	EC	5	2	FC	5	3				M + 1 → B, M → A	•	•	‡	‡	R	•
Multiply Unsigned	MUL													3D	10	1	A x B → A : B	•	•	•	•	•	⑪
OR, Inclusive	ORAA	8A	2	2	9A	3	2	AA	4	2	BA	4	3				A + M → A	•	•	‡	‡	R	•
	ORAB	CA	2	2	DA	3	2	EA	4	2	FA	4	3				B + M → B	•	•	‡	‡	R	•
Push Data	PSHA													36	3	1	A → Msp, SP − 1 → SP	•	•	•	•	•	•
	PSHB													37	3	1	B → Msp, SP − 1 → SP	•	•	•	•	•	•
Pull Data	PULA													32	4	1	SP + 1 → SP, Msp → A	•	•	•	•	•	•
	PULB													33	4	1	SP + 1 → SP, Msp → B	•	•	•	•	•	•
Rotate Left	ROL							69	6	2	79	6	3					•	•	‡	‡	⑥	‡
	ROLA													49	2	1		•	•	‡	‡	⑥	‡
	ROLB													59	2	1		•	•	‡	‡	⑥	‡
Rotate Right	ROR							66	6	2	76	6	3					•	•	‡	‡	⑥	‡
	RORA													46	2	1		•	•	‡	‡	⑥	‡
	RORB													56	2	1		•	•	‡	‡	⑥	‡

The Condition Code Register notes are listed after Table 10.

(Continued)

Table 7 Accumulator & Memory Instructions (Continued)

Operations	Mnemonic	Addressing Modes														Boolean/ Arithmetic Operation	Condition Code Register					
		IMMED.			DIRECT			INDEX			EXTEND			IMPLIED				5	4	3	2	1	0
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#		H	I	N	Z	V	C
Shift Left Arithmetic	ASL							68	6	2	78	6	3				M, A, B ← 0 (C b7 b0)	•	•	↕	↕	⑥	↕
	ASLA													48	2	1		•	•	↕	↕	⑥	↕
	ASLB													58	2	1		•	•	↕	↕	⑥	↕
Double Shift Left, Arithmetic	ASLD													05	3	1	C ← ACC A/ ACC B ← 0 (A7 A0 B7 B0)	•	•	↕	↕	⑥	↕
Shift Right Arithmetic	ASR							67	6	2	77	6	3				M, A, B → (b7 b0 C)	•	•	↕	↕	⑥	↕
	ASRA													47	2	1		•	•	↕	↕	⑥	↕
	ASRB													57	2	1		•	•	↕	↕	⑥	↕
Shift Right Logical	LSR							64	6	2	74	6	3				0 → M, A, B → (b7 b0 C)	•	•	↕	↕	⑥	↕
	LSRA													44	2	1		•	•	↕	↕	⑥	↕
	LSRB													54	2	1		•	•	↕	↕	⑥	↕
Double Shift Right Logical	LSRD													04	3	1	0 → ACC A/ ACC B → C (A7 A0 B7 B0)	•	•	R	↕	⑥	↕
Store Accumulator	STAA				97	3	2	A7	4	2	B7	4	3				A → M	•	•	↕	↕	R	•
	STAB				D7	3	2	E7	4	2	F7	4	3				B → M	•	•	↕	↕	R	•
Store Double Accumulator	STD				DD	4	2	ED	5	2	FD	5	3				A → M B → M + 1	•	•	↕	↕	R	•
Subtract	SUBA	80	2	2	90	3	2	A0	4	2	B0	4	3				A – M → A	•	•	↕	↕	↕	↕
	SUBB	C0	2	2	D0	3	2	E0	4	2	F0	4	3				B – M → B	•	•	↕	↕	↕	↕
Double Subtract	SUBD	83	4	3	93	5	2	A3	6	2	B3	6	3				A : B – M : M + 1 → A : B	•	•	↕	↕	↕	↕
Subtract Accumulators	SBA													10	2	1	A – B → A	•	•	↕	↕	↕	↕
Subtract With Carry	SBCA	82	2	2	92	3	2	A2	4	2	B2	4	3				A – M – C → A	•	•	↕	↕	↕	↕
	SBCB	C2	2	2	D2	3	2	E2	4	2	F2	4	3				B – M – C → B	•	•	↕	↕	↕	↕
Transfer Accumulators	TAB													16	2	1	A → B	•	•	↕	↕	R	•
	TBA													17	2	1	B → A	•	•	↕	↕	R	•
Test Zero or Minus	TST							6D	6	2	7D	6	3				M – 00	•	•	↕	↕	R	R
	TSTA													4D	2	1	A – 00	•	•	↕	↕	R	R
	TSTB													5D	2	1	B – 00	•	•	↕	↕	R	R

The Condition Code Register notes are listed after Table 10.

Direct Addressing

In direct addressing, the address of the operand is contained in the second byte of the instruction. Direct addressing allows the user to directly address the lowest 256 bytes in the machine i.e., locations zero through 255. Enhanced execution times are achieved by storing data in these locations. In most configurations, it should be a random access memory. These are two-byte instructions.

Extended Addressing

In extended addressing, the address contained in the second byte of the instruction is used as the higher 8-bits of the address of the operand. The third byte of the instruction is used as the lower 8-bits of the address for the operand. This is an absolute address in memory. These are three-byte instructions.

Indexed Addressing

In indexed addressing, the address contained in the second byte of the instruction is added to the index register's lowest 8-bits in the MPU. The carry is then added to the higher order 8-bits of the index register. This result is then used to address memory. The modified address is held in a temporary address register so there is no change to the index register. These are two-byte instructions.

Implied Addressing

In the implied addressing mode the instruction gives the address (i.e., stack pointer, index register, etc.). These are one-byte instructions.

Relative Addressing

In relative addressing, the address contained in the second byte of the instruction is added to the program counter's lowest 8-bits plus two. The carry or borrow is then added to the high 8-bits. This allows the user to address data within a range of -126 to +129 bytes of the present instruction. These are two-byte instructions.

- **New Instructions**

In addition to the existing 6800 Instruction Set, the following new instructions are incorporated in the HD6803 Microcomputer.

ABX Adds the 8-bit unsigned accumulator B to the 16-bit X-Register taking into account the possible carry out of the low order byte of the X-Register.

ADDD Adds the double precision ACCD* to the double precision value M:M+1 and places the results in ACCD.

ASLD Shifts all bits of ACCD one place to the left. Bit 0 is loaded with zero. The C bit is loaded from the most significant bit of ACCD.

LDD Loads the contents of double precision memory location into the double accumulator A:B. The condition codes are set according to the data.

LSRD Shifts all bits of ACCD one place to the right. Bit 15 is loaded with zero. The C bit is loaded from the least significant bit to ACCD.

MUL Multiplies the 8 bits in accumulator A with the 8 bits in accumulator B to obtain a 16-bit unsigned number in A:B, ACCA contains MSB of result.

PSHX The contents of the index register is pushed onto the stack at the address contained in the stack pointer. The stack pointer is decremented by 2.

PULX The index register is pulled from the stack beginning at the current address contained in the stack pointer +1. The stack pointer is incremented by 2 in total.

STD Stores the contents of double accumulator A:B in memory. The contents of ACCD remain unchanged.

SUBD Subtracts the contents of M:M + 1 from the contents of double accumulator AB and places the result in ACCD.

BRN Never branches. If effect, this instruction can be considered a two byte NOP (No operation) requiring three cycles for execution.

CPX Internal processing modified to permit its use with any conditional branch instruction.

*ACCD' is the 16 bit register (A:B) formed by concatenating the A and B accumulators. The A-accumulator is the most significant byte.

Table 8 Index Register and Stack Manipulation Instructions

Pointer Operations	Mnemonic	Addressing Modes															Boolean/ Arithmetic Operation	Condition Code Register					
		IMMED.			DIRECT			INDEX			EXTND			IMPLIED				5	4	3	2	1	0
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#		H	I	N	Z	V	C
Compare Index Reg	CPX	8C	4	3	9C	5	2	AC	6	2	BC	6	3				X − M : M + 1	•	•	↕	↕	↕	↕
Decrement Index Reg	DEX													09	3	1	X − 1 → X	•	•	•	↕	•	•
Decrement Stack Pntr	DES													34	3	1	SP − 1 → SP	•	•	•	•	•	•
Increment Index Reg	INX													08	3	1	X + 1 → X	•	•	•	↕	•	•
Increment Stack Pntr	INS													31	3	1	SP + 1 → SP	•	•	•	•	•	•
Load Index Reg	LDX	CE	3	3	DE	4	2	EE	5	2	FE	5	3				M → X_H, (M+1) → X_L	•	•	⑦	↕	R	•
Load Stack Pntr	LDS	8E	3	3	9E	4	2	AE	5	2	BE	5	3				M → SP_H, (M+1) → SP_L	•	•	⑦	↕	R	•
Store Index Reg	STX				DF	4	2	EF	5	2	FF	5	3				X_H → M, X_L → (M + 1)	•	•	⑦	↕	R	•
Store Stack Pntr	STS				9F	4	2	AF	5	2	BF	5	3				SP_H → M, SP_L → (M+1)	•	•	⑦	↕	R	•
Index Reg → Stack Pntr	TXS													35	3	1	X − 1 → SP	•	•	•	•	•	•
Stack Pntr → Index Reg	TSX													30	3	1	SP + 1 → X	•	•	•	•	•	•
Add	ABX													3A	3	1	B + X → X	•	•	•	•	•	•
Push Data	PSHX													3C	4	1	X_L → M_{sp}, SP − 1 → SP X_H → M_{sp}, SP − 1 → SP	•	•	•	•	•	•
Pull Data	PULX													38	5	1	SP + 1 → SP, M_{sp} → X_H SP + 1 → SP, M_{sp} → X_L	•	•	•	•	•	•

The Condition Code Register notes are listed after Table 10.

Table 9 Jump and Branch Instructions

Operations	Mnemonic	Addressing Modes															Branch Test	Condition Code Register					
		RELATIVE			DIRECT			INDEX			EXTND			IMPLIED				5	4	3	2	1	0
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#		H	I	N	Z	V	C
Branch Always	BRA	20	3	2													None	•	•	•	•	•	•
Branch Never	BRN	21	3	2													None	•	•	•	•	•	•
Branch If Carry Clear	BCC	24	3	2													C = 0	•	•	•	•	•	•
Branch If Carry Set	BCS	25	3	2													C = 1	•	•	•	•	•	•
Branch If = Zero	BEQ	27	3	2													Z = 1	•	•	•	•	•	•
Branch If ≥ Zero	BGE	2C	3	2													N ⊕ V = 0	•	•	•	•	•	•
Branch If > Zero	BGT	2E	3	2													Z + (N ⊕ V) = 0	•	•	•	•	•	•
Branch If Higher	BHI	22	3	2													C + Z = 0	•	•	•	•	•	•
Branch If ≤ Zero	BLE	2F	3	2													Z + (N ⊕ V) = 1	•	•	•	•	•	•
Branch If Lower Or Same	BLS	23	3	2													C + Z = 1	•	•	•	•	•	•
Branch If < Zero	BLT	2D	3	2													N ⊕ V = 1	•	•	•	•	•	•
Branch If Minus	BMI	2B	3	2													N = 1	•	•	•	•	•	•
Branch If Not Equal Zero	BNE	26	3	2													Z = 0	•	•	•	•	•	•
Branch If Overflow Clear	BVC	28	3	2													V = 0	•	•	•	•	•	•
Branch If Overflow Set	BVS	29	3	2													V = 1	•	•	•	•	•	•
Branch If Plus	BPL	2A	3	2													N = 0	•	•	•	•	•	•
Branch To Subroutine	BSR	8D	6	2													See Special Operations	•	•	•	•	•	•
Jump	JMP							6E	3	2	7E	3	3					•	•	•	•	•	•
Jump To Subroutine	JSR				9D	5	2	AD	6	2	BD	6	3					•	•	•	•	•	•
No Operation	NOP													01	2	1	Advances Prog. Cntr. Only	•	•	•	•	•	•
Return From Interrupt	RTI													3B	10	1	See Special Operations	——			⑧		——
Return From Subroutine	RTS													39	5	1		•	•	•	•	•	•
Software Interrupt	SWI													3F	12	1		•	S	•	•	•	•
Wait for Interrupt	WAI													3E	9	1		•	⑨	•	•	•	•

Table10 Condition Code Register Manipulation Instructions

Operations	Mnemonic	AddressingModes			Boolean Operation	Condition Code Register					
		IMPLIED				5	4	3	2	1	0
		OP	~	#		H	I	N	Z	V	C
Clear Carry	CLC	0C	2	1	0 → C	•	•	•	•	•	R
Clear Interrupt Mask	CLI	0E	2	1	0 → I	•	R	•	•	•	•
Clear Overflow	CLV	0A	2	1	0 → V	•	•	•	•	R	•
Set Carry	SEC	0D	2	1	1 → C	•	•	•	•	•	S
Set Interrupt Mask	SEI	0F	2	1	1 → I	•	S	•	•	•	•
Set Overflow	SEV	0B	2	1	1 → V	•	•	•	•	S	•
Accumulator A → CCR	TAP	06	2	1	A → CCR	——			⑩		——
CCR → Accumulator A	TPA	07	2	1	CCR → A	•	•	•	•	•	•

Condition Code Register Notes: (Bit set it test is true and cleared otherwise)

① (Bit V) Test: Result = 10000000?
② (Bit C) Test: Result = 00000000?
③ (Bit C) Test: Decimal value of most significant BCD Character greater than nine? (Not cleared if previously set)
④ (Bit V) Test: Operand = 10000000 prior to execution?
⑤ (Bit V) Test: Operand = 01111111 prior to execution?
⑥ (Bit V) Test: Set equal to result of N ⊕ C after shift has occurred.
⑦ (Bit N) Test: Result less than zero? (Bit 15 = 1)
⑧ (All) Load Condition Code Register from Stack. (See Special Operations)
⑨ (Bit I) Set when interrupt occurs. If previously set, a Non-Maskable Interrupt is required to exit the wait state.
⑩ (All) Set according to the contents of Accumulator A.
⑪ (Bit C) Set equal to result of Bit 7 (AccB)

Table 11 Instruction Execution Times in Machine Cycle

	ACCX	Immediate	Direct	Extended	Indexed	Implied	Relative		ACCX	Immediate	Direct	Extended	Indexed	Implied	Relative
ABA	•	•	•	•	•	2	•	INX	•	•	•	•	•	3	•
ABX	•	•	•	•	•	3	•	JMP	•	•	•	3	3	•	•
ADC	•	2	3	4	4	•	•	JSR	•	•	5	6	6	•	•
ADD	•	2	3	4	4	•	•	LDA	•	2	3	4	4	•	•
ADDD	•	4	5	6	6	•	•	LDD	•	3	4	5	5	•	•
AND	•	2	3	4	4	•	•	LDS	•	3	4	5	5	•	•
ASL	2	•	•	6	6	•	•	LDX	•	3	4	5	5	•	•
ASLD	•	•	•	•	•	3	•	LSR	2	•	•	6	6	•	•
ASR	2	•	•	6	6	•	•	LSRD	•	•	•	•	•	3	•
BCC	•	•	•	•	•	•	3	MUL	•	•	•	•	•	10	•
BCS	•	•	•	•	•	•	3	NEG	2	•	•	6	6	•	•
BEQ	•	•	•	•	•	•	3	NOP	•	•	•	•	•	2	•
BGE	•	•	•	•	•	•	3	ORA	•	2	3	4	4	•	•
BGT	•	•	•	•	•	•	3	PSH	3	•	•	•	•	•	•
BHI	•	•	•	•	•	•	3	PSHX	•	•	•	•	•	4	•
BIT	•	2	3	4	4	•	•	PUL	4	•	•	•	•	•	•
BLE	•	•	•	•	•	•	3	PULX	•	•	•	•	•	5	•
BLS	•	•	•	•	•	•	3	ROL	2	•	•	6	6	•	•
BLT	•	•	•	•	•	•	3	ROR	2	•	•	6	6	•	•
BMI	•	•	•	•	•	•	3	RTI	•	•	•	•	•	10	•
BNE	•	•	•	•	•	•	3	RTS	•	•	•	•	•	5	•
BPL	•	•	•	•	•	•	3	SBA	•	•	•	•	•	2	•
BRA	•	•	•	•	•	•	3	SBC	•	2	3	4	4	•	•
BRN	•	•	•	•	•	•	3	SEC	•	•	•	•	•	2	•
BSR	•	•	•	•	•	•	6	SEI	•	•	•	•	•	2	•
BVC	•	•	•	•	•	•	3	SEV	•	•	•	•	•	2	•
BVS	•	•	•	•	•	•	3	STA	•	•	3	4	4	•	•
CBA	•	•	•	•	•	2	•	STD	•	•	4	5	5	•	•
CLC	•	•	•	•	•	2	•	STS	•	•	4	5	5	•	•
CLI	•	•	•	•	•	2	•	STX	•	•	4	5	5	•	•
CLR	2	•	•	6	6	•	•	SUB	•	2	3	4	4	•	•
CLV	•	•	•	•	•	2	•	SUBD	•	4	5	6	6	•	•
CMP	•	2	3	4	4	•	•	SWI	•	•	•	•	•	12	•
COM	2	•	•	6	6	•	•	TAB	•	•	•	•	•	2	•
CPX	•	4	5	6	6	•	•	TAP	•	•	•	•	•	2	•
DAA	•	•	•	•	•	2	•	TBA	•	•	•	•	•	2	•
DEC	2	•	•	6	6	•	•	TPA	•	•	•	•	•	2	•
DES	•	•	•	•	•	3	•	TST	2	•	•	6	6	•	•
DEX	•	•	•	•	•	3	•	TSX	•	•	•	•	•	3	•
EOR	•	2	3	4	4	•	•	TXS	•	•	•	•	•	3	•
INC	2	•	•	6	6	•	•	WAI	•	•	•	•	•	9	•
INS	•	•	•	•	•	3	•								

- **Summary of Cycle by Cycle Operation**

Table 12 provides a detailed description of the information present on the Address Bus, Data Bus, and the Read/Write line (R/$\overline{W}$) during each cycle for each instruction.

This information is useful in comparing actual with expected results during debug of both software and hardware as the control program is executed. The information is categorized in groups according to addressing mode and number of cycles per instruction. (In general, instructions with the same addressing mode and number of cycles execute in the same manner; exceptions are indicated in the table).

Table 12 Cycle by Cycle Operation

Address Mode & Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
IMMEDIATE					
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	2	1 2	Op Code Address Op Code Address + 1	1 1	Op Code Operand Data
LDS LDX	3	1 2 3	Op Code Address Op Code Address + 1 Op Code Address + 2	1 1 1	Op Code Operand Data (High Order Byte) Operand Data (Low Order Byte)
CPX SUBD ADDD	4	1 2 3 4	Op Code Address Op Code Address + 1 Op Code Address + 2 Address Bus FFFF	1 1 1 1	Op Code Operand Data (High Order Byte) Operand Data (Low Order Byte) Low Byte of Restart Vector
DIRECT					
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	3	1 2 3	Op Code Address Op Code Address + 1 Address of Operand	1 1 1	Op Code Address of Operand Operand Data
STA	3	1 2 3	Op Code Address Op Code Address + 1 Destination Address	1 1 0	Op Code Destination Address Data from Accumulator
LDS LDX LDD	4	1 2 3 4	Op Code Address Op Code Address + 1 Address of Operand Operand Address + 1	1 1 1 1	Op Code Address of Operand Operand Data (High Order Byte) Operand Data (Low Order Byte)
STS STX STD	4	1 2 3 4	Op Code Address Op Code Address + 1 Address of Operand Address of Operand + 1	1 1 0 0	Op Code Address of Operand Register Data (High Order Byte) Register Data (Low Order Byte)
CPX SUBD ADDD	5	1 2 3 4 5	Op Code Address Op Code Address + 1 Operand Address Operand Address + 1 Address Bus FFFF	1 1 1 1 1	Op Code Address of Operand Operand Data (High Order Byte) Operand Data (Low Order Byte) Low Byte of Restart Vector
JSR	5	1 2 3 4 5	Op Code Address Op Code Address + 1 Subroutine Address Stack Pointer Stack Pointer + 1	1 1 1 0 0	Op Code Irrelevant Data First Subroutine Op Code Return Address (Low Order Byte) Return Address (High Order Byte)

(Continued)

Table 12 Cycle by Cycle Operation (Continued)

Address Mode & Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
INDEXED					
JMP	3	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
ADC EOR	4	1	Op Code Address	1	Op Code
ADD LDA		2	Op Code Address + 1	1	Offset
AND ORA		3	Address Bus FFFF	1	Low Byte of Restart Vector
BIT SBC		4	Index Register Plus Offset	1	Operand Data
CMP SUB					
STA	4	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	0	Operand Data
LDS	5	1	Op Code Address	1	Op Code
LDX		2	Op Code Address + 1	1	Offset
LDD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	1	Operand Data (High Order Byte)
		5	Index Register Plus Offset + 1	1	Operand Data (Low Order Byte)
STS	5	1	Op Code Address	1	Op Code
STX		2	Op Code Address + 1	1	Offset
STD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	0	Operand Data (High Order Byte)
		5	Index Register Plus Offset + 1	0	Operand Data (Low Order Byte)
ASL LSR	6	1	Op Code Address	1	Op Code
ASR NEG		2	Op Code Address + 1	1	Offset
CLR ROL		3	Address Bus FFFF	1	Low Byte of Restart Vector
COM ROR		4	Index Register Plus Offset	1	Current Operand Data
DEC TST*		5	Address Bus FFFF	1	Low Byte of Restart Vector
INC		6	Index Register Plus Offset	0	New Operand Data
CPX	6	1	Op Code Address	1	Op Code
SUBD		2	Op Code Address + 1	1	Offset
ADDD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register + Offset	1	Operand Data (High Order Byte)
		5	Index Register + Offset + 1	1	Operand Data (Low Order Byte)
		6	Address Bus FFFF	1	Low Byte of Restart Vector
JSR	6	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register + Offset	1	First Subroutine Op Code
		5	Stack Pointer	0	Return Address (Low Order Byte)
		6	Stack Pointer – 1	0	Return Address (High Order Byte)

* In the TST instruction, R/$\overline{W}$ line of the sixth cycle is "1" level, and AB = FFFF, DB = Low Byte of Reset Vector.

(Continued)

Table 12 Cycle by Cycle Operation (Continued)

Address Mode & Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
EXTENDED					
JMP	3	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Jump Address (High Order Byte)
		3	Op Code Address + 2	1	Jump Address (Low Order Byte)
ADC EOR	4	1	Op Code Address	1	Op Code
ADD LDA		2	Op Code Address + 1	1	Address of Operand
AND ORA		3	Op Code Address + 2	1	Address of Operand (Low Order Byte)
BIT SBC		4	Address of Operand	1	Operand Data
CMP SUB					
STA A	4	1	Op Code Address	1	Op Code
STA B		2	Op Code Address + 1	1	Destination Address (High Order Byte)
		3	Op Code Address + 2	1	Destination Address (Low Order Byte)
		4	Operand Destination Address	0	Data from Accumulator
LDS	5	1	Op Code Address	1	Op Code
LDX		2	Op Code Address + 1	1	Address of Operand (High Order Byte)
LDD		3	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	Address of Operand	1	Operand Data (High Order Byte)
		5	Address of Operand + 1	1	Operand Data (Low Order Byte)
STS	5	1	Op Code Address	1	Op Code
STX		2	Op Code Address + 1	1	Address of Operand (High Order Byte)
STD		3	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	Address of Operand	0	Operand Data (High Order Byte)
		5	Address of Operand + 1	0	Operand Data (Low Order Byte)
ASL LSR	6	1	Op Code Address	1	Op Code
ASR NEG		2	Op Code Address + 1	1	Address of Operand (High Order Byte)
CLR ROL		3	Op Code Address + 2	1	Address of Operand (Low Order Byte)
COM ROR		4	Address of Operand	1	Current Operand Data
DEC TST*		5	Address Bus FFFF	1	Low Byte of Restart Vector
INC		6	Address of Operand	0	New Operand Data
CPX	6	1	Op Code Address	1	Op Code
SUBD		2	Op Code Address + 1	1	Operand Address (High Order Byte)
ADDD		3	Op Code Address + 2	1	Operand Address (Low Order Byte)
		4	Operand Address	1	Operand Data (High Order Byte)
		5	Operand Address + 1	1	Operand Data (Low Order Byte)
		6	Address Bus FFFF	1	Low Byte of Restart Vector
JSR	6	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Address of Subroutine (High Order Byte)
		3	Op Code Address + 2	1	Address of Subroutine (Low Order Byte)
		4	Subroutine Starting Address	1	Op Code of Next Instruction
		5	Stack Pointer	0	Return Address (Low Order Byte)
		6	Stack Pointer – 1	0	Address of Operand (High Order Byte)

* In the TST instruction, R/$\overline{W}$ line of the sixth cycle is "1" level, and AB = FFFF, DB = Low Byte of Reset Vector.

(Continued)

Table 12 Cycle by Cycle Operation (Continued)

Address Mode & Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
IMPLIED					
ABA DAA SEC ASL DEC SEI ASR INC SEV CBA LSR TAB CLC NEG TAP CLI NOP TBA CLR ROL TPA CLV ROR TST COM SBA	2	1 2	Op Code Address Op Code Address + 1	1 1	Op Code Op Code of Next Instruction
ABX	3	1 2 3	Op Code Address Op Code Address + 1 Address Bus FFFF	1 1 1	Op Code Irrelevant Data Low Byte of Restart Vector
ASLD LSRD	3	1 2 3	Op Code Address Op Code Address + 1 Address Bus FFFF	1 1 1	Op Code Irrelevant Data Low Byte of Restart Vector
DES INS	3	1 2 3	Op Code Address Op Code Address + 1 Previous Register Contents	1 1 1	Op Code Op Code of Next Instruction Irrelevant Data
INX DEX	3	1 2 3	Op Code Address Op Code Address + 1 Address Bus FFFF	1 1 1	Op Code Op Code of Next Instruction Low Byte of Restart Vector
PSHA PSHB	3	1 2 3	Op Code Address Op Code Address + 1 Stack Pointer	1 1 0	Op Code Op Code of Next Instruction Accumulator Data
TSX	3	1 2 3	Op Code Address Op Code Address + 1 Stack Pointer	1 1 1	Op Code Op Code of Next Instruction Irrelevant Data
TXS	3	1 2 3	Op Code Address Op Code Address + 1 Address Bus FFFF	1 1 1	Op Code Op Code of Next Instruction Low Byte of Restart Vector
PULA PULB	4	1 2 3 4	Op Code Address Op Code Address + 1 Stack Pointer Stack Pointer + 1	1 1 1 1	Op Code Op Code of Next Instruction Irrelevant Data
PSHX	4	1 2 3 4	Op Code Address Op Code Address + 1 Stack Pointer Stack Pointer – 1	1 1 0 0	Op Code Irrelevant Data Index Register (Low Order Byte) Index Register (High Order Byte)
PULX	5	1 2 3 4 5	Op Code Address Op Code Address + 1 Stack Pointer Stack Pointer + 1 Stack Pointer +2	1 1 1 1 1	Op Code Irrelevant Data Irrelevant Data Index Register (High Order Byte) Index Register (Low Order Byte)
RTS	5	1 2 3 4 5	Op Code Address Op Code Address + 1 Stack Pointer Stack Pointer + 1 Stack Pointer + 2	1 1 1 1 1	Op Code Irrelevant Data Irrelevant Data Address of Next Instruction (High Order Byte) Address of Next Instruction (Low Order Byte)
WAI**	9	1 2 3 4	Op Code Address Op Code Address + 1 Stack Pointer Stack Pointer – 1	1 1 0 0	Op Code Op Code of Next Instruction Return Address (Low Order Byte) Return Address (High Order Byte)

(Continued)

Table 12 Cycle by Cycle Operation (Continued)

Address Mode & Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
WAI**		5	Stack Pointer – 2	0	Index Register (Low Order Byte)
		6	Stack Pointer – 3	0	Index Register (High Order Byte)
		7	Stack Pointer – 4	0	Contents of Accumulator A
		8	Stack Pointer – 5	0	Contents of Accumulator B
		9	Stack Pointer – 6	0	Contents of Cond. Code Register
MUL	10	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Irrelevant Data
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Address Bus FFFF	1	Low Byte of Restart Vector
		5	Address Bus FFFF	1	Low Byte of Restart Vector
		6	Address Bus FFFF	1	Low Byte of Restart Vector
		7	Address Bus FFFF	1	Low Byte of Restart Vector
		8	Address Bus FFFF	1	Low Byte of Restart Vector
		9	Address Bus FFFF	1	Low Byte of Restart Vector
		10	Address Bus FFFF	1	Low Byte of Restart Vector
RTI	10	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Irrelevant Data
		3	Stack Pointer	1	Irrelevant Data
		4	Stack Pointer + 1	1	Contents of Cond. Code Reg. from Stack
		5	Stack Pointer + 2	1	Contents of Accumulator B from Stack
		6	Stack Pointer + 3	1	Contents of Accumulator A from Stack
		7	Stack Pointer + 4	1	Index Register from Stack (High Order Byte)
		8	Stack Pointer + 5	1	Index Register from Stack (Low Order Byte)
		9	Stack Pointer + 6	1	Next Instruction Address from Stack (High Order Byte)
		10	Stack Pointer + 7	1	Next Instruction Address from Stack (Low Order Byte)
SWI	12	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Irrelevant Data
		3	Stack Pointer	0	Return Address (Low Order Byte)
		4	Stack Pointer – 1	0	Return Address (High Order Byte)
		5	Stack Pointer – 2	0	Index Register (Low Order Byte)
		6	Stack Pointer – 3	0	Index Register (High Order Byte)
		7	Stack Pointer – 4	0	Contents of Accumulator A
		8	Stack Pointer – 5	0	Contents of Accumulator B
		9	Stack Pointer – 6	0	Contents of Cond. Code Register
		10	Stack Pointer – 7	1	Irrelevant Data
		11	Vector Address FFFA (Hex)	1	Address of Subroutine (High Order Byte)
		12	Vector Address FFFB (Hex)	1	Address of Subroutine (Low Order Byte)

(Continued)

** While the MPU is in the "Wait" state, its bus state will appear as a series of MPU reads of an address which is seven locations less than the original contents of the Stack Pointer. Contrary to the HD6800, none of the ports are driven to the high impedance state by a WAI instruction.

Table 12 Cycle by Cycle Operation (Continued)

RELATIVE

Address Mode & Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
BCC BHT BNE BCS BLE BPL BEQ BLS BRA BGE BLT BVC BGT BMT BVS BRN	3	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Branch Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
BSR	6	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Branch Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Subroutine Starting Address	1	Op Code of Next Instruction
		5	Stack Pointer	0	Return Address (Low Order Byte)
		6	Stack Pointer − 1	0	Return Address (High Order Byte)

- **Summary of Undefined Instruction Operations**

The HD6803 has 36 underfined instructions. When these are carried out, the contents of Register and Memory in MPU change at random.

When the op codes (4E, 5E) are used to execute, the MPU continues to increase the program counter and it will not stop until the Reset signal enters. These op codes are used to test the LSI.

Table 13 Op codes Map

HD6803 MICROCOMPUTER INSTRUCTIONS

OP CODE						ACC A	ACC B	IND	EXT	ACCA or SP				ACCB or X				
										IMM	DIR	IND	EXT	IMM	DIR	IND	EXT	
LO \ HI		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111	
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0000	0		SBA	BRA	TSX	NEG				SUB								0
0001	1	NOP	CBA	BRN	INS					CMP								1
0010	2			BHI	PULA (+1)					SBC								2
0011	3			BLS	PULB (+1)	COM				*	SUBD (+2)			*	ADDD (+2)			3
0100	4	LSRD (+1)		BCC	DES	LSR				AND								4
0101	5	ASLD (+1)		BCS	TXS					BIT								5
0110	6	TAP	TAB	BNE	PSHA	ROR				LDA								6
0111	7	TPA	TBA	BEQ	PSHB	ASR					STA				STA			7
1000	8	INX (+1)		BVC	PULX (+2)	ASL				EOR								8
1001	9	DEX (+1)	DAA	BVS	RTS (+2)	ROL				ADC								9
1010	A	CLV		BPL	ABX	DEC				ORA								A
1011	B	SEV	ABA	BMI	RTI (+7)					ADD								B
1100	C	CLC		BGE	PSHX (+1)	INC				*	CPX (+2)			*	LDD (+1)			C
1101	D	SEC		BLT	MUL (+7)	TST				BSR (+4)	JSR (+2)			* (+1)	STD (+1)			D
1110	E	CLI		BGT	WAI (+6)	**		JMP (−3)		*	LDS (+1)			*	LDX (+1)			E
1111	F	SEI		BLE	SWI (+9)	CLR				* (+1)	STS (+1)			* (+1)	STX (+1)			F
BYTE/CYCLE		1/2	1/2	2/3	1/3	1/2	1/2	2/6	3/6	2/2	2/3	2/4	3/4	2/2	2/3	2/4	3/4	

[NOTES] 1) Undefined Op codes are marked with ▱.

2) ( ) indicate that the number in parenthesis must be added to the cycle count for that instruction.

3) The instructions shown below are all 3 bytes and are marked with "*".
Immediate addressing mode of SUBD, CPX, LDS, ADDD, LDD and LDX instructions, and undefined op codes (8F, CD, CF).

4) The Op codes (4E, 5E) are 1 byte/∞ cycles instructions, and are marked with "**"

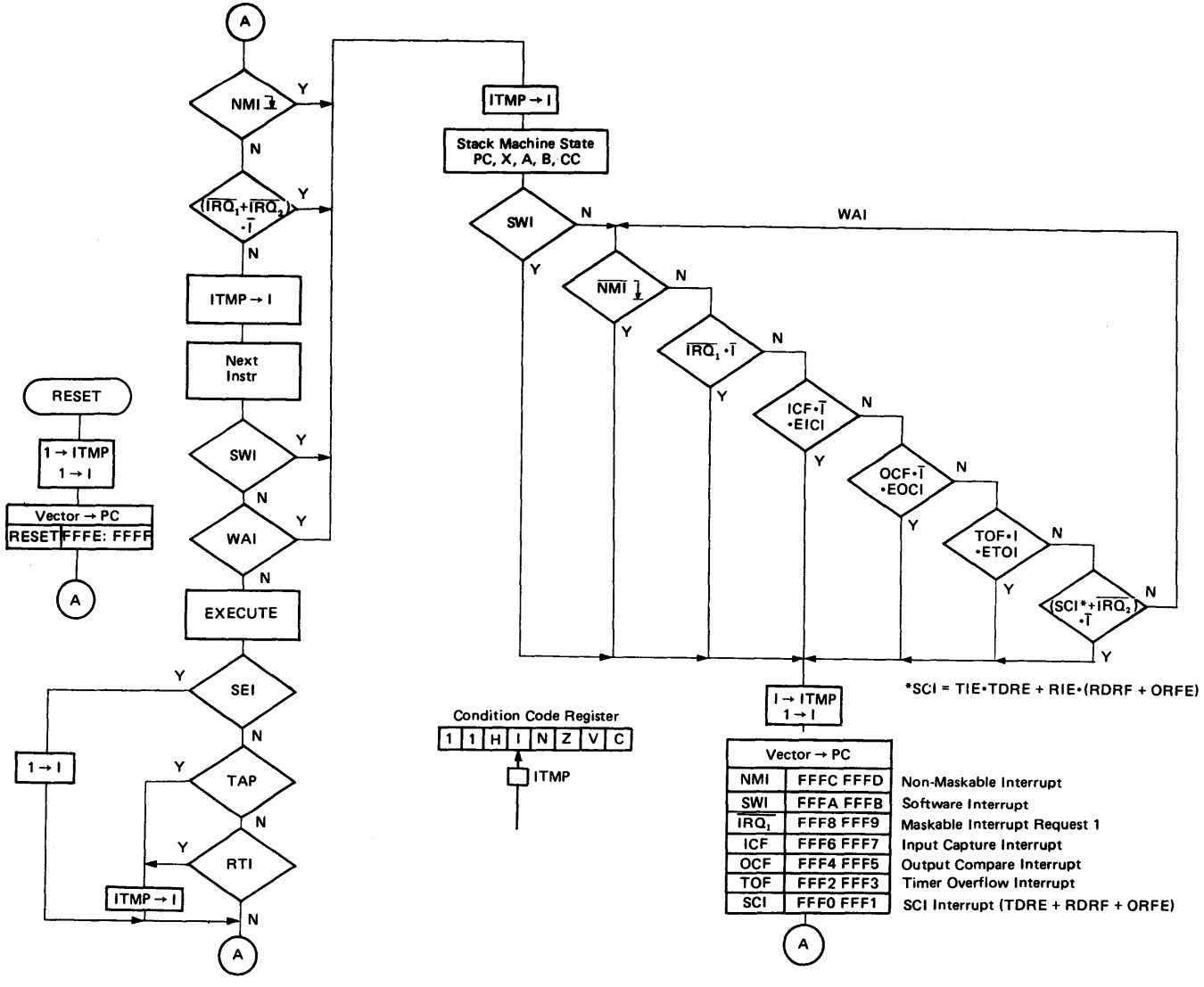


Figure 16 Interrupt Flowchart

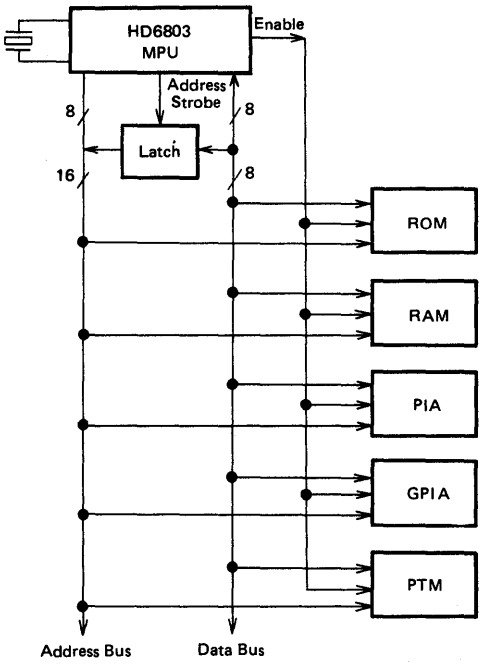


Figure 17 HD6803 MPU Expanded Multiplexed Bus

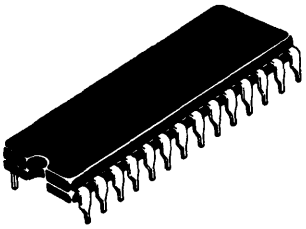
HD6805S1

MCU (Microcomputer Unit)

The HD6805S1 is the 8-bit Microcomputer Unit (MCU) which contains a CPU, on-chip clock, ROM, RAM, I/O and timer. It is designed for the user who needs an economical microcomputer with the proven capabilities of the HD 6800-based instruction set.

The following are some of the hardware and software highlights of the MCU.

HD6805S1P



(DP-28)

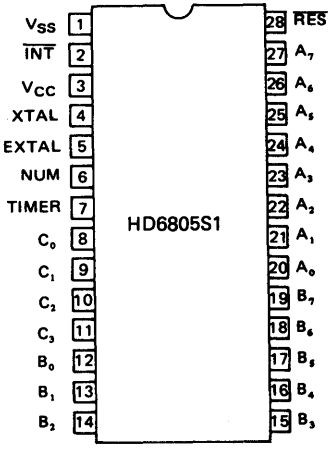
■ HARDWARE FEATURES

- 8-Bit Architecture
- 64 Bytes of RAM
- Memory Mapped I/O
- 1100 Bytes of User ROM
- Internal 8-Bit Timer with 7-Bit Prescaler
- Vectored Interrupts – External and Timer
- 20 TTL/CMOS Compatible I/O Lines; 8 Lines LED Compatible
- On-Chip Clock Circuit
- Self-Check Mode
- Master Reset
- Low Voltage Inhibit
- Complete Development System Support by Evaluation kit
- 5 Vdc Single Supply
- Compatible with MC6805P2

■ SOFTWARE FEATURES

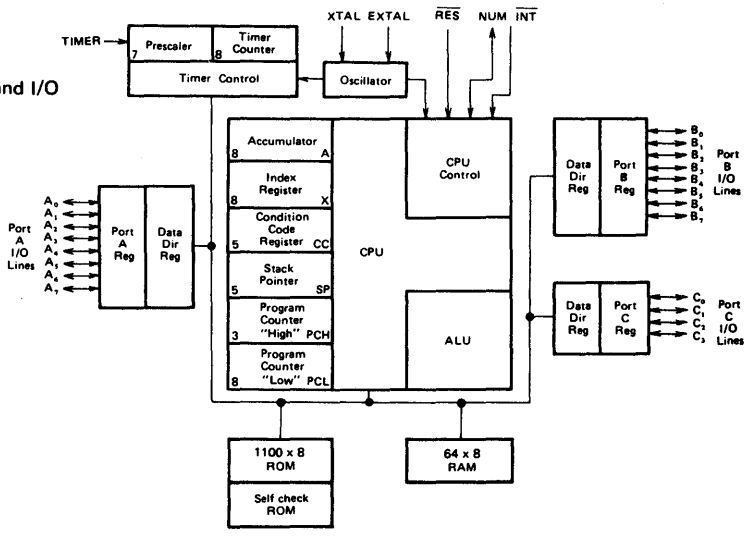
- Similar to HD6800
- Byte Efficient Instruction Set
- Easy to Program
- True Bit Manipulation
- Bit Test and Branch Instructions
- Versatile Interrupt Handing
- Powerful Indexed Addressing for Tables
- Full Set of Conditional Branches
- Memory Usable as Registers/Flags
- Single Instruction Memory Examine/Change
- 10 Powerful Addressing Modes
- All Addressing Modes Apply to ROM, RAM and I/O
- Compatible with MC6805P2

■ PIN ARRANGEMENT



(Top View)

■ BLOCK DIAGRAM



■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC} *	-0.3 ~ +7.0	V
Input Voltage (EXCEPT TIMER)	V_{in} *	-0.3 ~ +7.0	V
Input Voltage (TIMER)		-0.3 ~ +12.0	V
Operating Temperature	T_{opr}	0 ~ +70	°C
Storage Temperature	T_{stg}	- 55 ~ +150	°C

* With respect to V_{SS} (SYSTEM GND)

(NOTE) Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ ELECTRICAL CHARACTERISTICS

• DC CHARACTERISTICS (V_{CC}=5.25V ± 0.5V, V_{SS}=GND, Ta=0~+70°C, unless otherwise noted.)

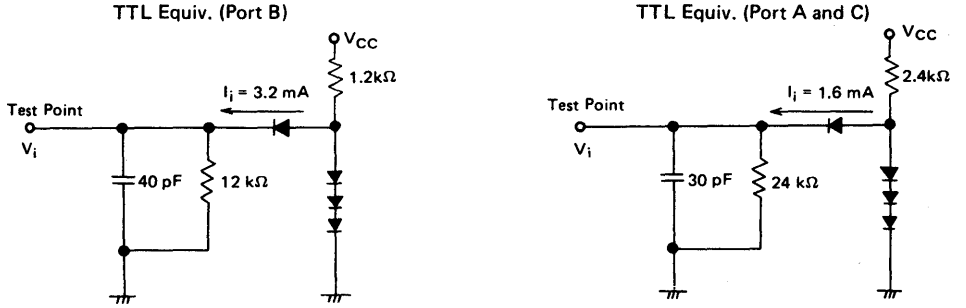
Item		Symbol	Test Condition	min	typ	max	Unit
Input "High" Voltage	$\overline{RES}$	V_{IH}		4.0	–	V_{CC}	V
	$\overline{INT}$			3.0	–	V_{CC}	V
	All Other			2.0	–	V_{CC}	V
Input "High" Voltage Timer	Timer Mode			2.0	–	V_{CC}	V
	Self-Check Mode			9.0	–	11.0	V
Input "Low" Voltage	$\overline{RES}$	V_{IL}		-0.3	–	0.8	V
	$\overline{INT}$			-0.3	–	0.8	V
	XTAL(Crystal Mode)			-0.3	–	0.6	V
	All Other			-0.3	–	0.8	V
Power Dissipation		P_D		–	400	700	mW
Low Voltage Recover		LVR		–	–	4.75	V
Low Voltage Inhibit		LVI		–	4.0	–	V
Input Leak Current	TIMER	I_{IL}	V_{in}=0.4V~V_{CC}	-20	–	20	μA
	$\overline{INT}$			-50	–	50	μA
	XTAL(Crystal Mode)			-1200	–	0	μA

• AC CHARACTERISTICS (V_{CC}=5.25V ± 0.5V, V_{SS}=GND, Ta=0 ~ +70°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Clock Frequency		f_{cl}		0.4	–	4.0	MHz
Cycle Time		t_{cyc}		1.0	–	10	μs
Oscillation Frequency (External Resister Mode)		f_{EXT}	R_{CP}=15.0kΩ±1%	2.7	–	4.0	MHz
$\overline{INT}$ Pulse Width		t_{IWL}		$\frac{t_{cyc}}{250}$+	–	–	ns
$\overline{RES}$ Pulse Width		t_{RWL}		$\frac{t_{cyc}}{250}$+	–	–	ns
TIMER Pulse Width		t_{TWL}		$\frac{t_{cyc}}{250}$+	–	–	ns
Oscillation Start-up Time (Crystal Mode)		t_{OSC}	C_L=22pF±20%, R_S=60Ω max.	–	–	100	ms
Delay Time Reset		t_{RHL}	External Cap. = 2.2 μF	100	–	–	ms
Input Capacitance	EXTAL	C_{in}	V_{in}=0V	–	25	35	pF
	All Other			–	6	10	pF

• PORT ELECTRICAL CHARACTERISTICS (V_{CC} = 5.25V ± 0.5V, V_{SS} = GND, Ta = 0 ~ +70°C unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Output "High" Voltage	Port A	V_{OH}	I_{OH} = −10 μA	3.5	–	–	V
			I_{OH} = −100 μA	2.4	–	–	V
	Port B		I_{OH} = −200 μA	2.4	–	–	V
			I_{OH} = −1 mA	1.5	–	–	V
	Port C		I_{OH} = −100 μA	2.4	–	–	V
Output "Low" Voltage	Port A and C	V_{OL}	I_{OL} = 1.6 mA	–	–	0.4	V
	Port B		I_{OL} = 3.2 mA	–	–	0.4	V
			I_{OL} = 10 mA	–	–	1.0	V
Input "High" Voltage	Port A, B, C	V_{IH}		2.0	–	V_{CC}	V
Input "Low" Voltage		V_{IL}		−0.3	–	0.8	V
Input Leak Current	Port A	I_{IL}	V_{in} = 0.8V	−500	–	–	μA
			V_{in} = 2V	−300	–	–	μA
	Port B, C		V_{in} = 0.4V ~ V_{CC}	− 20	–	20	μA



(NOTE) 1. Load capacitance includes the floating capacitance of the probe and the jig etc.
2. All diodes are 1S2074 Ⓗ or equivalent.

Figure 1 Bus Timing Test Loads

■ SIGNAL DESCRIPTION

The input and output signals for the MCU shown in PIN ARRANGEMENT are described in the following paragraphs.

• V_{CC} and V_{SS}

Power is supplied to the MCU using these two pins. V_{CC} is +5.25 V ±0.5 V. V_{SS} is the ground connection.

• $\overline{INT}$

This pin provides the capability for applying an external interrupt to the MCU Refer to INTERRUPTS for additional information

• XTAL and EXTAL

These pins provide control input for the on-chip clock circuit. A crystal (AT cut, 4 MHz maximum) or a resistor can be connected to these pins to provide the internal oscillator with varying degrees of stability. Refer to INTERNAL OSCILLATOR OPTIONS for recommendations about these inputs.

• TIMER

This pin allows an external input to be used to decrement the internal timer circuitry. Refer to TIMER for additional information about the timer circuitry.

• $\overline{RES}$

This pin allows resetting of the MCU at times other than the automatic resetting capability already in the MCU. Refer to RESETS for additional information.

• NUM

This pin is not for user application and should be connected to ground.

• Input/Output Lines (A_0 ~ A_7, B_0 ~ B_7, C_0 ~ C_3)

These 20 lines are arranged into tow 8-bit ports (A and B) and one 4-bit port (C). All lines are programmable as either inputs or outputs under software control of the data direction registers. Refer to INPUTS/OUTPUTS for additional information.

■ MEMORY

The MCU memory is configured as shown in Figure 2. During the processing of an interrupt, the contents of the MCU registers are pushed onto the stack in the order shown in Figure 3. Since the stack pointer decrements during pushes, the low order byte (PCL) of the program counter is stacked first; then the high order three bits (PCH) are stacked. This ensures that the program counter is loaded correctly as the stack pointer increments when it pulls data from the stack. A subroutine call will cause only the program counter (PCH, PCL) contents to be pushed onto the stack.

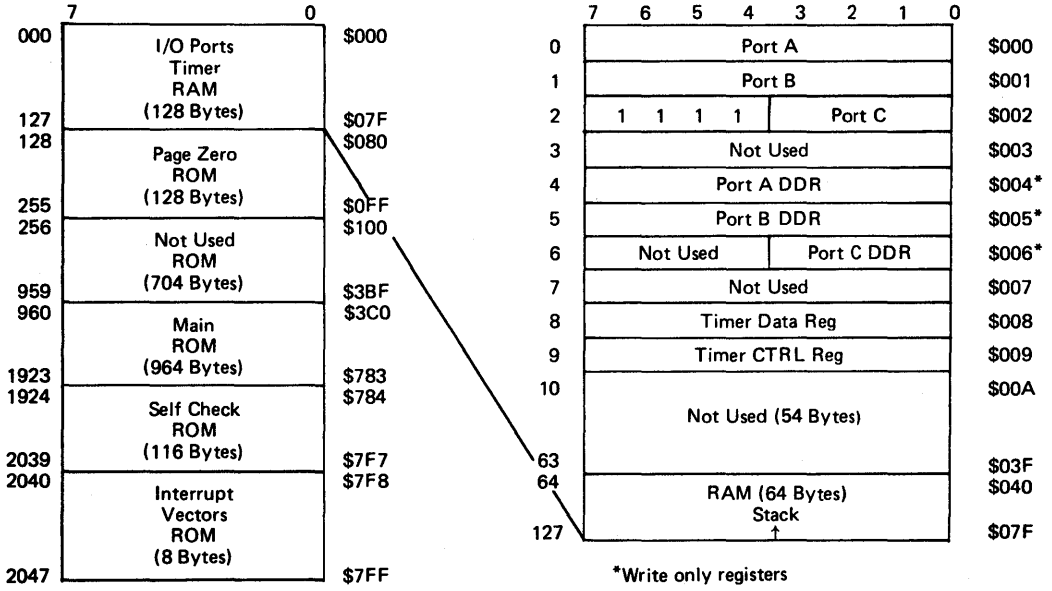


Figure 2 MCU Memory Configuration

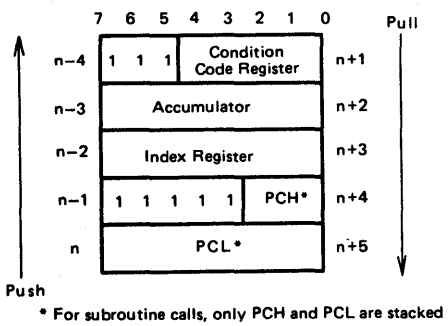


Figure 3 Interrupt Stacking Order

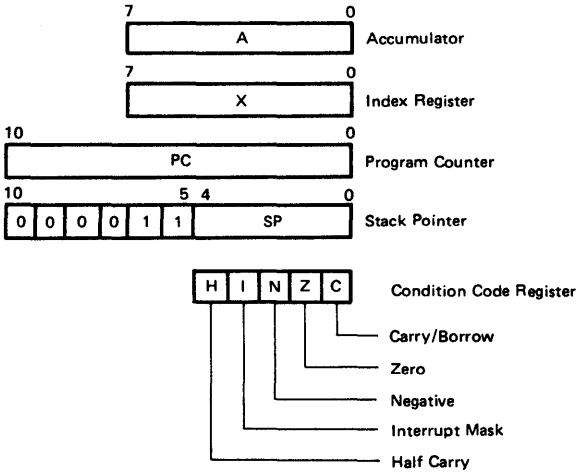


Figure 4 Programming Model

■ REGISTERS

The MCU has five registers available to the programmer. They are shown in Figure 4 and are explained in the following paragraphs.

- **Accumulator (A)**

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.

- **Index Register (X)**

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit address that may be added to an offset value to create an effective address. The index register can also be used for limited calculations and data manipulations when using read/modify/write instructions. When not required by a code sequence being executed, the index register can be used as a temporary storage area.

- **Program Counter (PC)**

The program counter is an 11-bit register that contains the address of the next instruction to be executed.

- **Stack Pointer (SP)**

The stack pointer is an 11-bit register that contains the address of the next free location on the stack. Initially, the stack pointer is set to location $07F and is decremented as data is being pushed onto the stack and incremented as data is being pulled from the stack. The six most significant bits of the stack pointer are permanently set to 000011. During a MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set to location $07F. Subroutines and interrupts may be nested down to location $061 which allows the programmer to use up to 15 levels of subroutine calls.

- **Condition Code Register (CC)**

The condition code register is a 5-bit register in which each bit is used to indicate or flag the results of the instruction just executed. These bits can be individually tested by a program and specific action taken as a result of their state. Each individual condition code register bit is explained in the following paragraphs.

- **Half Carry (H)**

Used during arithmetic operations (ADD and ADC) to indicate that a carry occurred between bits 3 and 4.

- **Interrupt (I)**

This bit is set to mask the timer and external interrupt ($\overline{INT}$). If an interrupt occurs while this bit is set it is latched and will be processed as soon as the interrupt bit is reset.

- **Negative (N)**

Used to indicate that the result of the last arithmetic, logical or data manipulation was negative (bit 7 in result equal to a logical one).

- **Zero (Z)**

Used to indicate that the result of the last arithmetic, logical or data manipulation was zero.

- **Carry/Borrow (C)**

Used to indicate that a carry or borrow out of the arithmetic logic unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions, shifts, and rotates.

■ TIMER

The MCU timer circuitry is shown in Figure 5. The 8-bit counter is loaded under program control and counts down toward zero as soon as the clock input is applied. When the timer reaches zero the timer interrupt request bit (bit 7) in the timer control register is set. The MCU responds to this interrupt by saving the present MCU state in the stack, fetching the timer interrupt vector from locations $7F8 and $7F9 and executing the interrupt routine. The timer interrupt can be masked by setting the timer interrupt mask bit (bit 6) in the timer control register. The interrupt bit (1 bit) in the condition code register will also prevent a timer interrupt from being processed.

The clock input to the timer can be from an external source applied to the TIMER input pin or it can be the internal ϕ_2 signal. Note that when the ϕ_2 signal is used as the source it can be gated by an input applied to the TIMER input pin allowing the user to easily perform pulse-width measurements. The

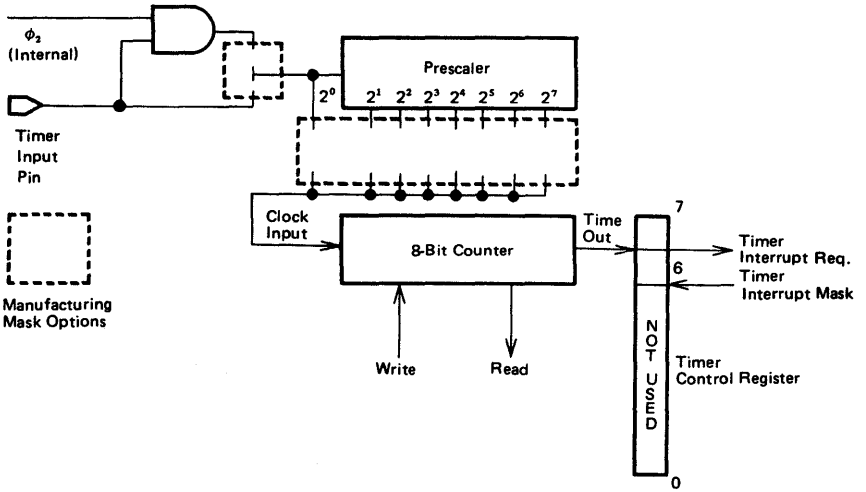


Figure 5 Timer Block Diagram

source of the clock input is one of the options that has to be specified before manufacture of the MCU. A prescaler option can be applied to the clock input that extends the timing interval up to a maximum of 128 counts before being applied to the counter. This prescaling option must also be specified before manufacturing begins. The timer continues to count past zero and its present count can be monitored at any time by monitoring the timer data register. This allows a program to determine the length of time since a timer interrupt has occured and not disturb the counting process.

At power up or reset the prescaler and counter are initialized with all logical ones; the timer interrupt request bit (bit 7) is cleared and the timer interrupt request mask bit (bit 6) is set.

■ SELF CHECK

The self check capability of the MCU provides an internal check to determine if the part is functional. Connect the MCU as shown in Figure 6 and monitor the output of port C bit 3 for an oscillation of approximately three hertz.

■ RESETS

The MCU can be reset three ways: by initial powerup, by the external reset input ($\overline{RES}$) and by an internal low voltage detect circuit, (mask option) see Figure 7. All the I/O port are initialized to Input mode (DDR's are cleared) during RESET.

Upon power up, a minimum of 100 milliseconds is needed before allowing the reset input to go "High".

This time allows the internal crystal oscillator to stabilize. Connecting a capacitor to the $\overline{RES}$ input as shown in Figure 8 will provide sufficient delay.

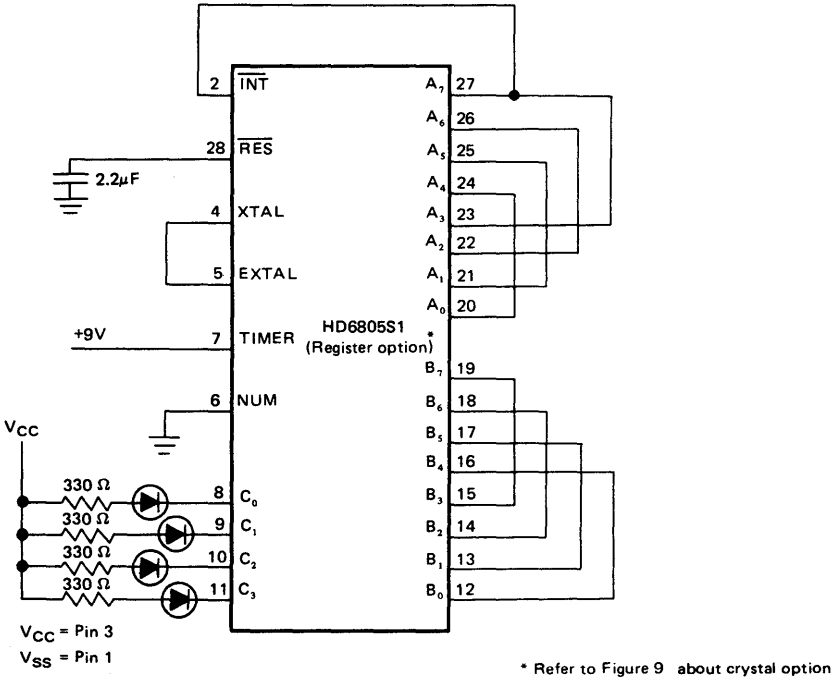


Figure 6 Self Check Connections

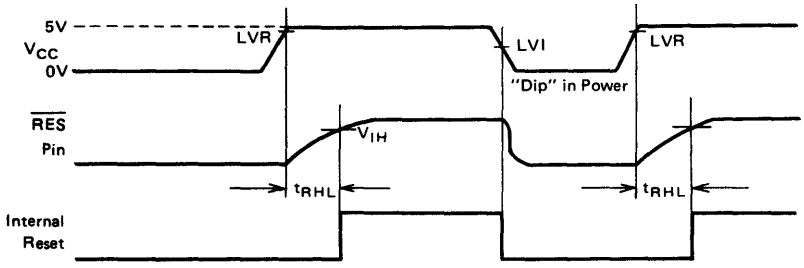


Figure 7 Power Up and $\overline{RES}$ Timing

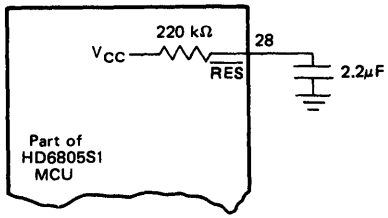


Figure 8 Power Up Reset Delay Circuit

■ INTERNAL OSCILLATOR OPTIONS

The internal oscillator circuit has been designed to require a minimum of external components. The use of a crystal (AT cut, 4 MHz max) or a resistor is sufficient to drive the internal oscillator with varying degrees of stability. A manufacturing mask option is available to provide better matching between the external components and the internal oscillator. The different connection methods are shown in Figure 9. Crystal specifications are given in Figure 10. A resistor selection graph is given in Figure 11.

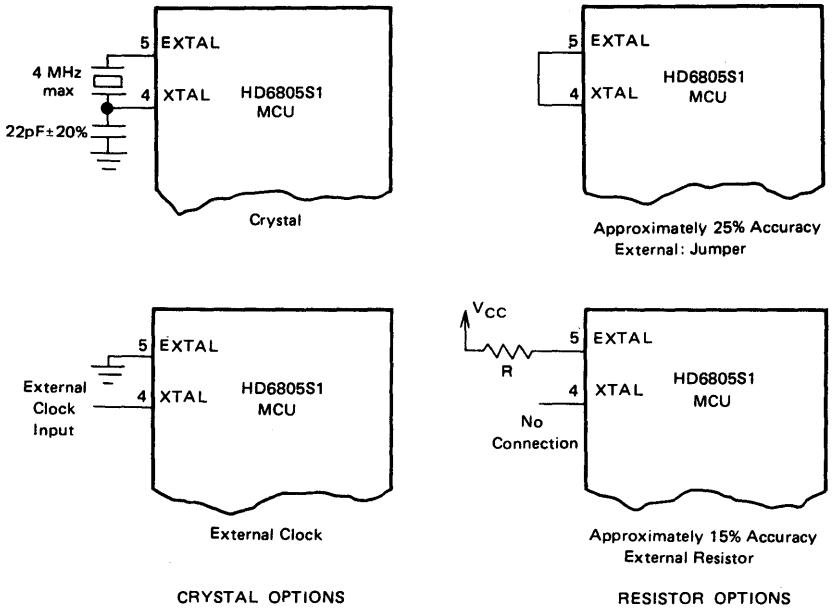
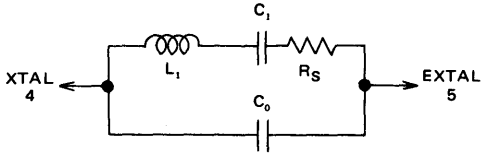


Figure 9 Internal Oscillator Options



AT – Cut Parallel Resonance Crystal
C_0 = 7 pF max.
f = 4 MHz (C_1 =22pF±20%)
R_S = 60 Ω max.

Figure 10 Crystal Parameters

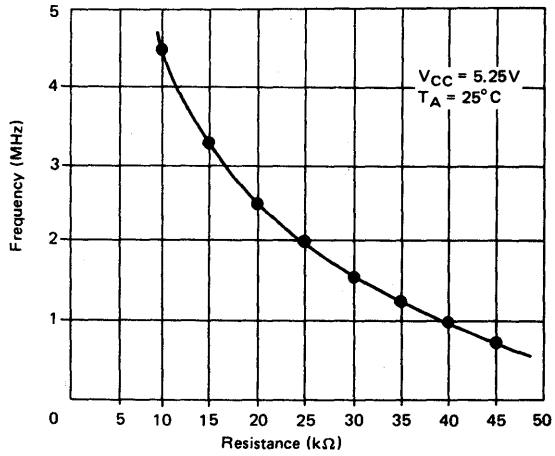


Figure 11 Typical Resistor Selection Graph

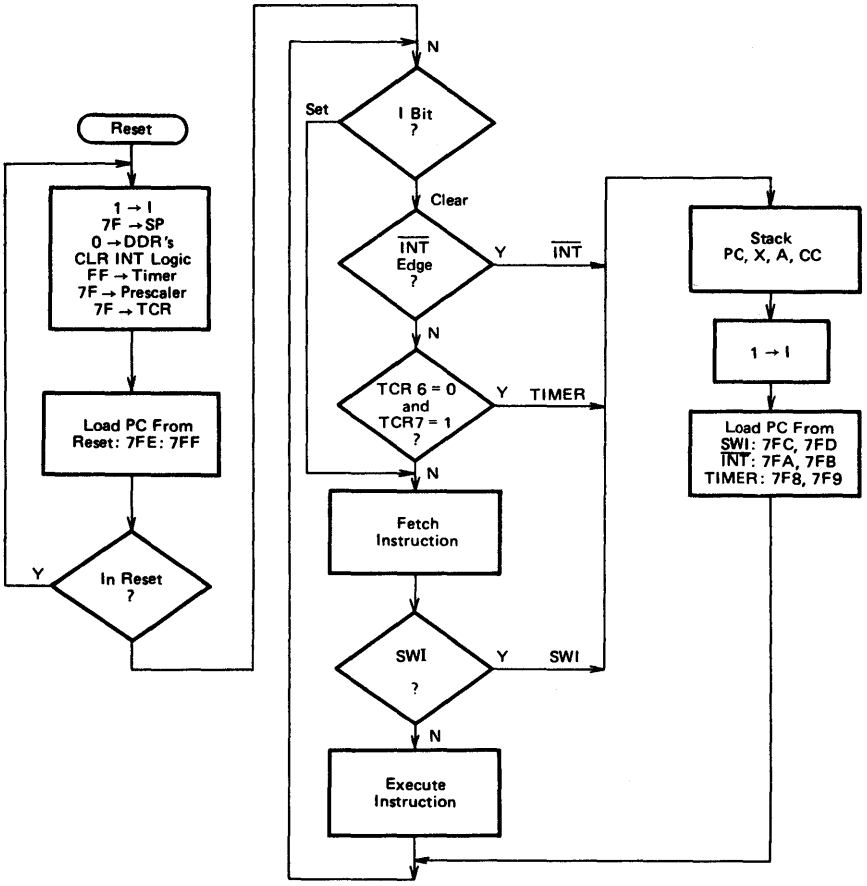
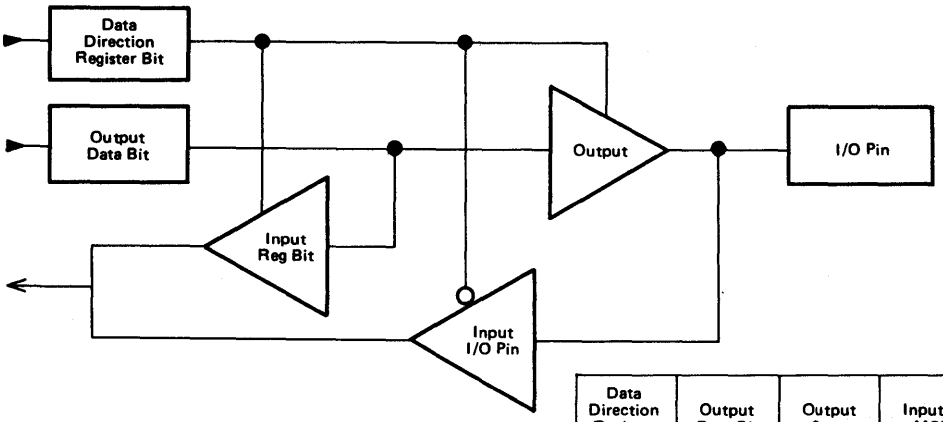


Figure 12 Interrupt Processing Flowchart



Data Direction Register Bit	Output Data Bit	Output State	Input to MCU
1	0	0	0
1	1	1	1
0	x	3-State	Pin

Figure 13 Typical Port I/O Circuitry

■ INTERRUPTS

The MCU can be interrupted three different ways: through the external interrupt ($\overline{INT}$) input pin, the internal timer interrupt request, and a software interrupt instruction (SWI). When any interrupt occurs, processing is suspended, the present MCU state is pushed onto the stack, the interrupt bit (I) in the condition code register is set, the address of the interrupt routine is obtained from the appropriate interrupt vector address, and the interrupt routine is executed. The interrupt service routines normally end with a return from interrupt (RTI) instruction which allows the MCU to resume processing of the program prior to the interrupt. Table 1 provides a listing of the interrupts, their priority, and the vector address that contain-the starting address of the appropriate interrupt routine.

A flowchart of the interrupt processing sequence is given in Figure 12.

Table 1 Interrupt Priorities

Interrupt	Priority	Vector Address
$\overline{RES}$	1	$7FE and $7FF
SWI	2	$7FC and $7FD
$\overline{INT}$	3	$7FA and $7FB
TIMER	4	$7F8 and $7F9

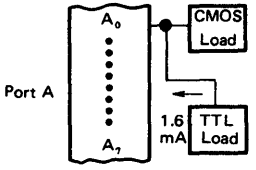
■ INPUT/OUTPUT

There are 20 input/output pins. All pins are programmable as either inputs or outputs under software control of the data direction registers. When programmed as outputs, all I/O pins read latched output data regardless of the logic level at the output pin due to output loading (see Figure 13). When port B is programmed for outputs, it is capable of sinking 10 millamperes on each pin (V_{OL} = 1V max). All input/output lines are TTL compatible as both inputs and outputs. Port A lines are CMOS compatible as outputs while port B and C lines are CMOS compatible as inputs. Figure 14 provides some examples of port connections.

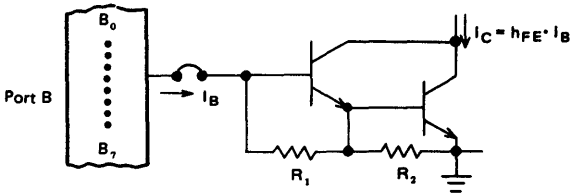
■ BIT MANIPULATION

The MCU has the ability to set or clear any single random access memory or input/output bit (except the data direction registers) with a single instruction (BSET, BCLR). Any bit in the page zero read only memory can be tested, using the BRSET and BRCLR instructions, and the program branches as a result of its state. This capability to work with any bit in RAM, ROM or I/O allows the user to have individual flags in RAM or to handle single I/O bits as control lines. The example in Figure 15 illustrates the usefulness of the bit manipulation and test instructions. Assume that bit 0 of port A is connected to a zero crossing detector circuit and that bit 1 of port A is connected to the trigger of a TRIAC which powers the controlled hardware.

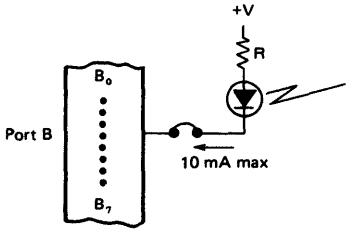
This program, which uses only seven ROM locations, provides turn-on of the TRIAC within 14 microseconds of the zero crossing. The timer could also be incorporated to provide turn-on at some later time which would permit pulse-width modulation of the controlled power.



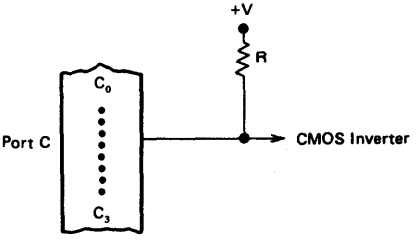
Port A Programmed as output(s) driving CMOS and TTL Load directly.
(a)



Port B Programmed as output(s) driving Darlington base directly.
(b)



Port B Programmed as output(s) driving LED(s) directly.
(c)



Port C Programmed as output(s) driving CMOS using external pull-up resistors.
(d)

Figure 14 Typical Port Connections

```
        .
        .
        .
SELF 1  BRCLR 0, PORT A, SELF 1
        BSET 1, PORT A
        BCLR 1, PORT A
        .
        .
        .
```

Figure 15 Bit Manipulation Example

■ ADDRESSING MODES

The MCU has ten addressing modes available for use by the programmer. They are explained and illustrated briefly in the following paragraphs.

- **Immediate**

Refer to Figure 16. The immediate addressing mode accesses constants which do not change during program execution. Such instructions are two bytes long. The effective address (EA) is the PC and the operand is fetched from the byte following the opcode.

- **Direct**

Refer to Figure 17. In direct addressing, the address of the operand is contained in the second byte of the instruction. Direct addressing allows the user to directly address the lowest 256 bytes in memory. All RAM space, I/O registers and 128 bytes of ROM are located in page zero to take advantage of this efficient memory addressing mode.

- **Extended**

Refer to Figure 18. Extended addressing is used to reference any location in memory space. The EA is the contents of the two bytes following the opcode. Extended addressing instructions are three bytes long.

- **Relative**

Refer to Figure 19. The relative addressing mode applies only to the branch instructions. In this mode the contents of the byte following the opcode is added to the program counter when the branch is taken. EA=(PC)+2+Rel. Rel is the contents of the location following the instruction opcode with bit 7 being the sign bit. If the branch is not taken Rel=0, when a branch takes place, the program goes to somewhere within the range of +129 bytes to -127 of the present instruction. These instructions are two bytes long.

- **Indexed (No Offset)**

Refer to Figure 20. This mode of addressing accesses the lowest 256 bytes of memory. These instructions are one byte long and their EA is the contents of the index register.

- **Indexed (8-bit Offset)**

Refer to Figure 21. The EA is calculated by adding the contents of the byte following the opcode to the contents of the index register. In this mode, 511 low memory locations are accessable. These instructions occupy two bytes.

- **Indexed (16-bit Offset)**

Refer to Figure 22. This addressing mode calculates the EA by adding the contents of the two bytes following the opcode to the index register. Thus, the entire memory space may be accessed. Instructions which use this addressing mode are three bytes long.

- **Bit Set/Clear**

Refer to Figure 23. This mode of addressing applies to instructions which can set or clear any bit on page zero. The lower three bits in the opcode specify the bit to be set or cleared while the byte following the opcode specifies the address in page zero.

- **Bit Test and Branch**

Refer to Figure 24. This mode of addressing applies to instructions which can test any bit in the first 256 locations ($00-$FF) and branch to any location relative to the PC. The byte to be tested is addressed by the byte following the opcode. The individual bit within that byte to be tested is addressed by the lower three bits of the opcode. The third byte is the relative address to be added to the program counter if the branch condition is met. These instructions are three bytes long. The value of the bit tested is written to the carry bit in the condition code register.

- **Implied**

Refer to Figure 25. The implied mode of addressing has no EA. All the information necessary to execute an instruction is contained in the opcode. Direct operations on the accumulator and the index register are included in this mode of addressing. In addition, control instructions such as SWI, RTI belong to this group. All implied addressing instructions are one byte long.

■ INSTRUCTION SET

The MCU has a set of 59 basic instructions. They can be divided into five different types: register/memory, read/modify/write, branch, bit manipulation, and control. The following paragraphs briefly explain each type. All the instructions within a given type are presented in individual tables.

- **Register/Memory Instructions**

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to Table 2.

- **Read/Modify/Write Instructions**

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read/modify/write instructions since it does not perform the write. Refer to Table 3.

- **Branch Instructions**

The branch instructions cause a branch from the program when a certain condition is met. Refer to Table 4.

- **Bit Manipulation Instructions**

These instructions are used on any bit in the first 256 bytes of the memory. One group either sets or clears. The other group performs the bit test and branch operations. Refer to Table 5.

- **Control Instructions**

The control instructions control the MCU operations during program execution. Refer to Table 6.

- **Alphabetical Listing**

The complete instruction set is given in alphabetical order in Table 7.

- **Opcode Map**

Table 8 is an opcode map for the instructions used on the MCU.

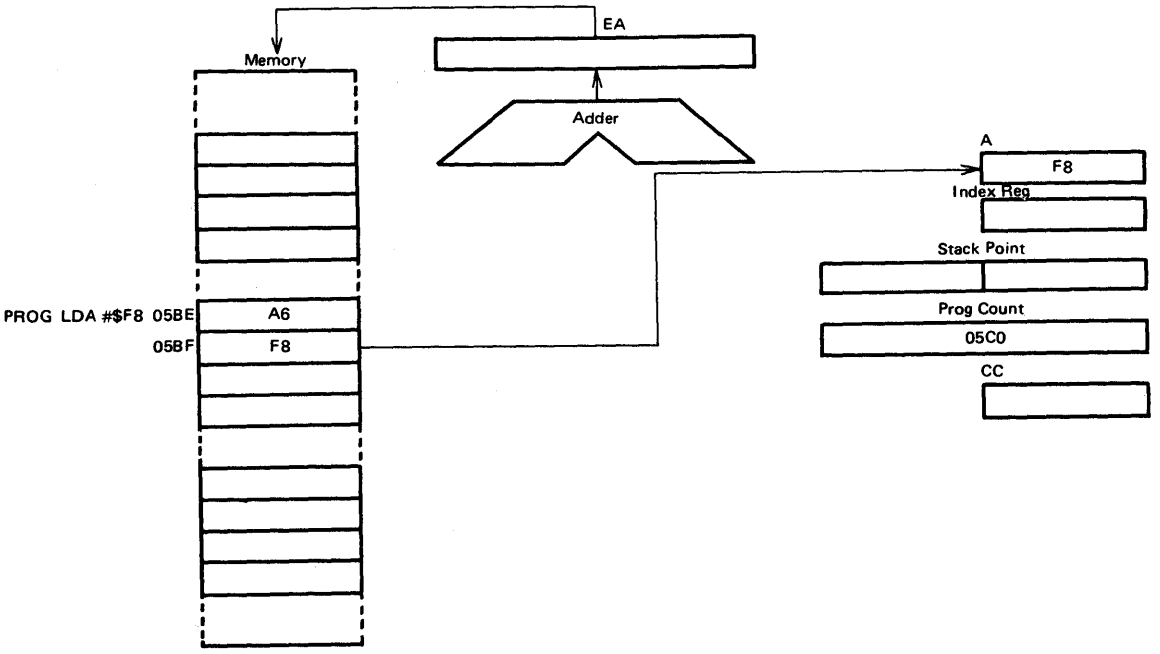


Figure 16 Immediate Addressing Example

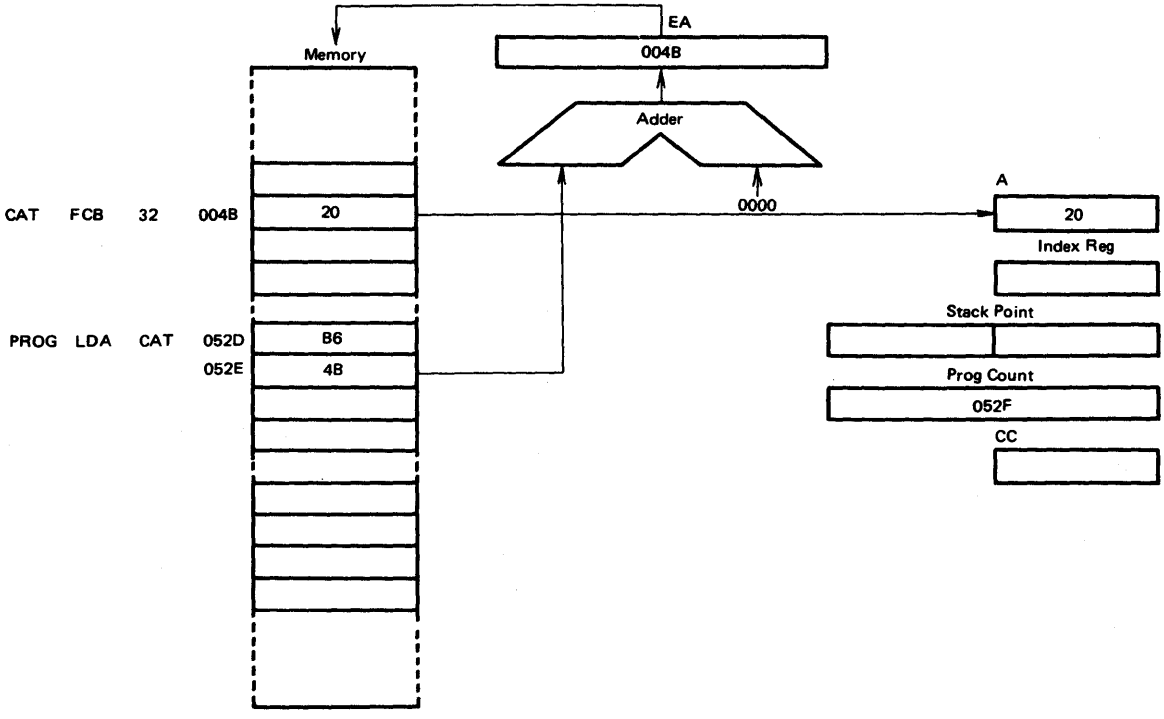


Figure 17 Direct Addressing Example

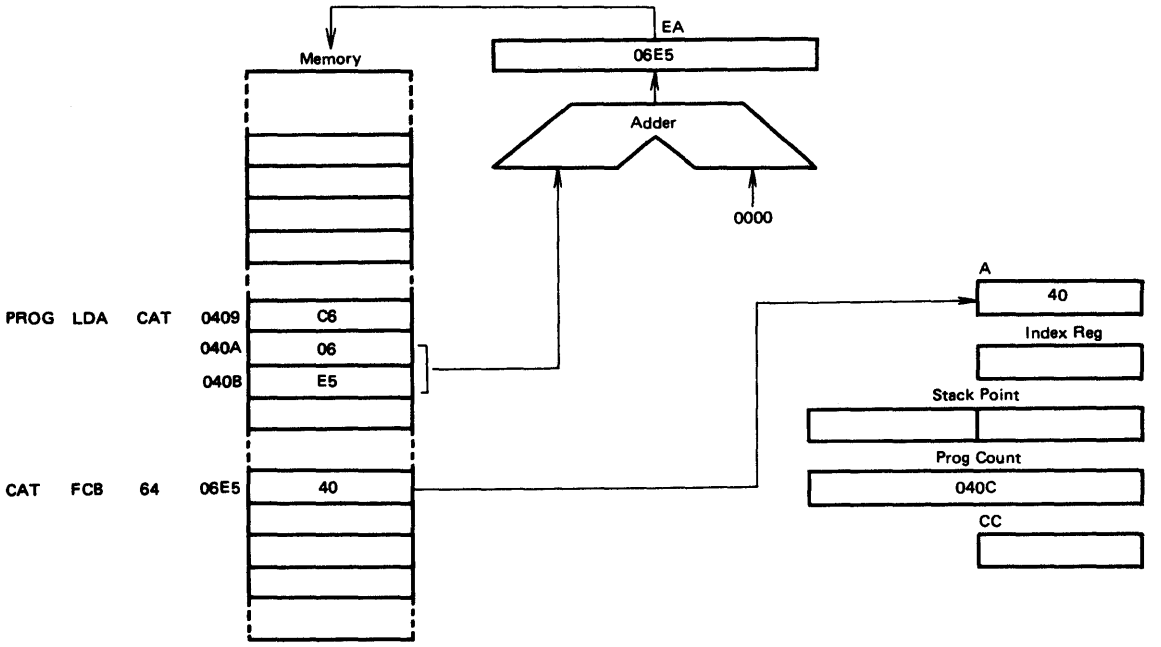


Figure 18 Extended Addressing Example

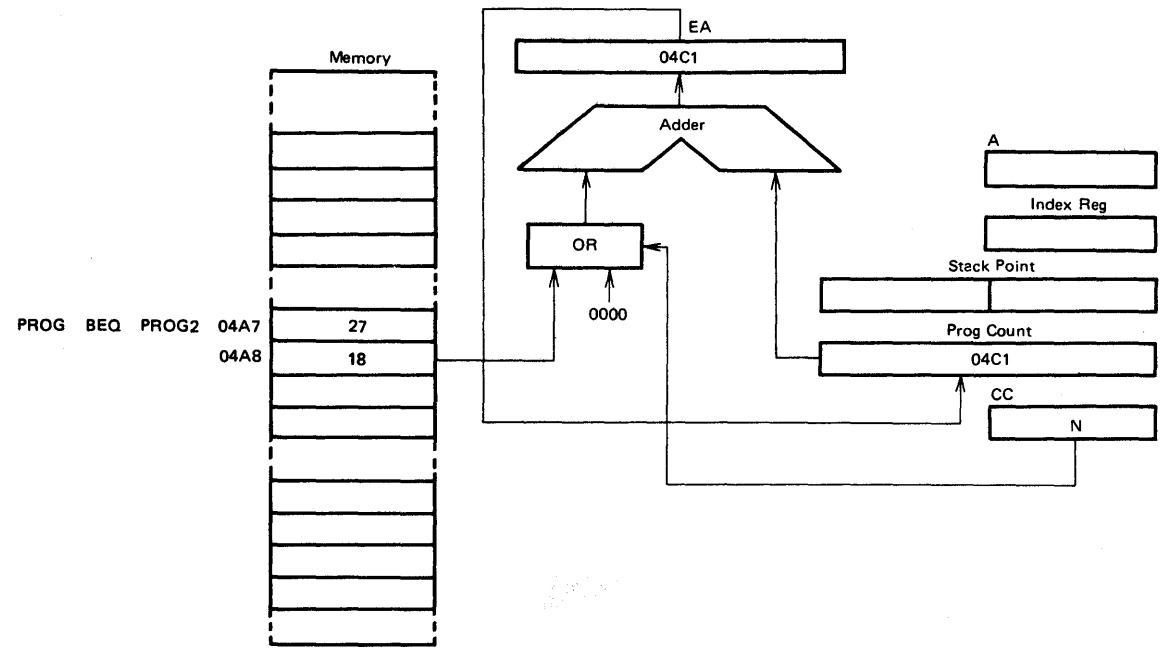


Figure 19 Relative Addressing Example

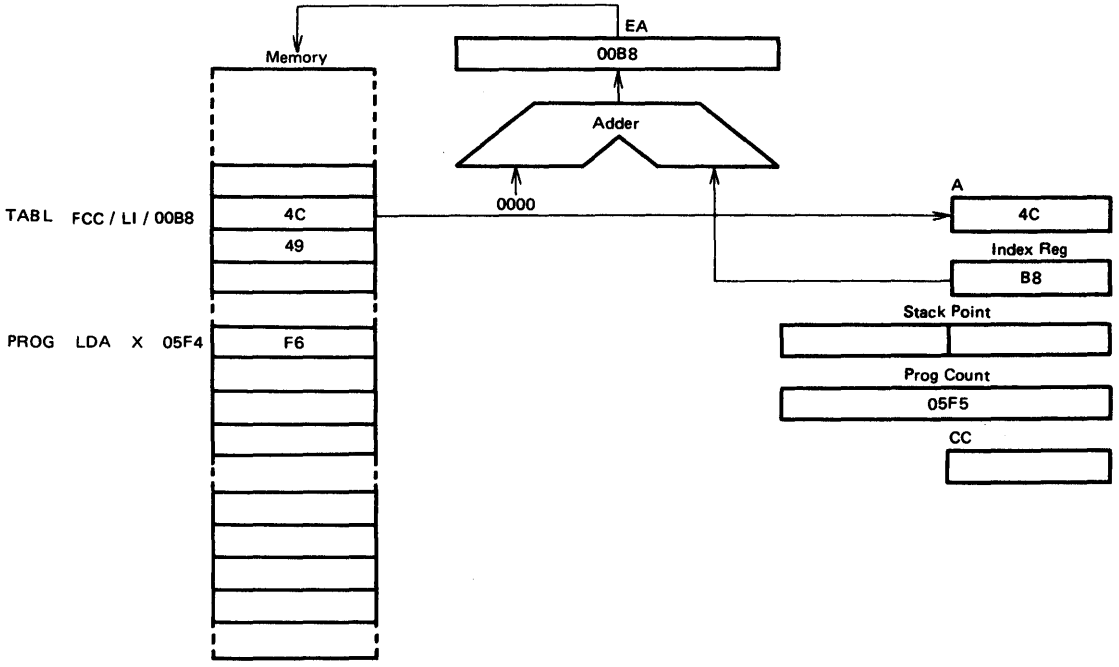


Figure 20 Indexed (No Offset) Addressing Example

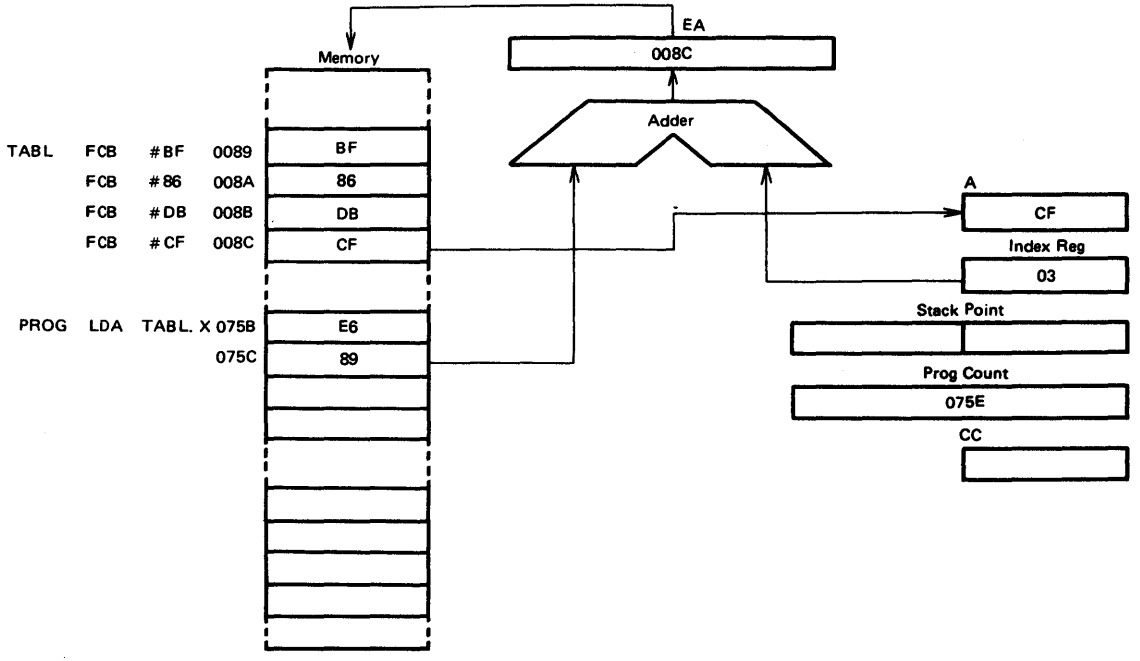


Figure 21 Indexed (8-Bit Offset) Addressing Example

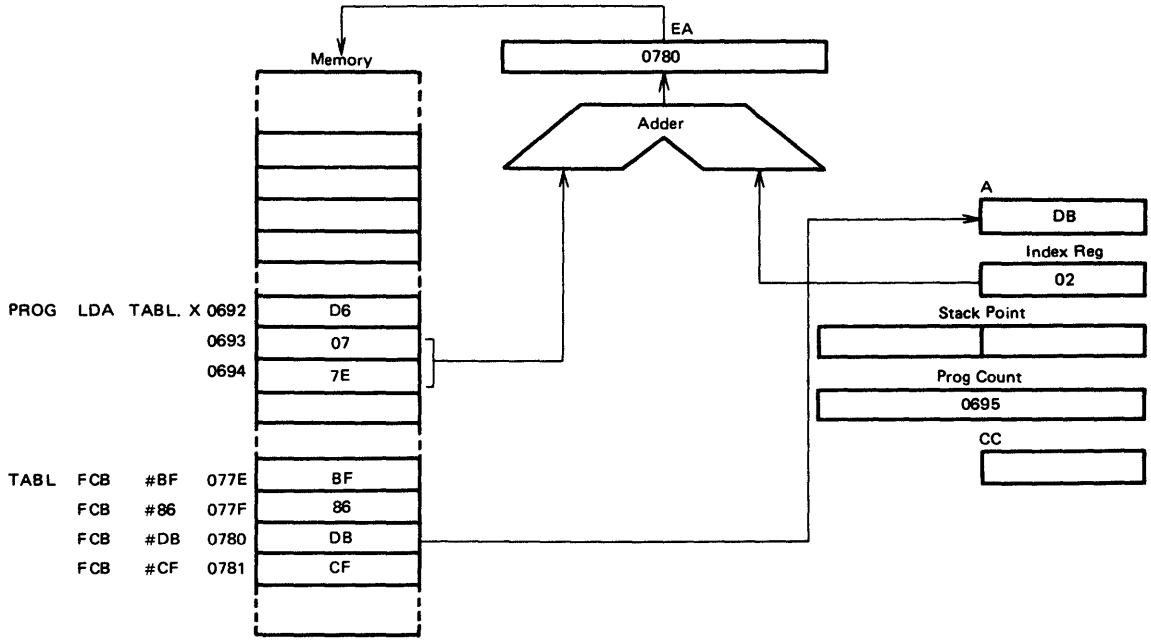


Figure 22 Indexed (16-Bit Offset) Addressing Example

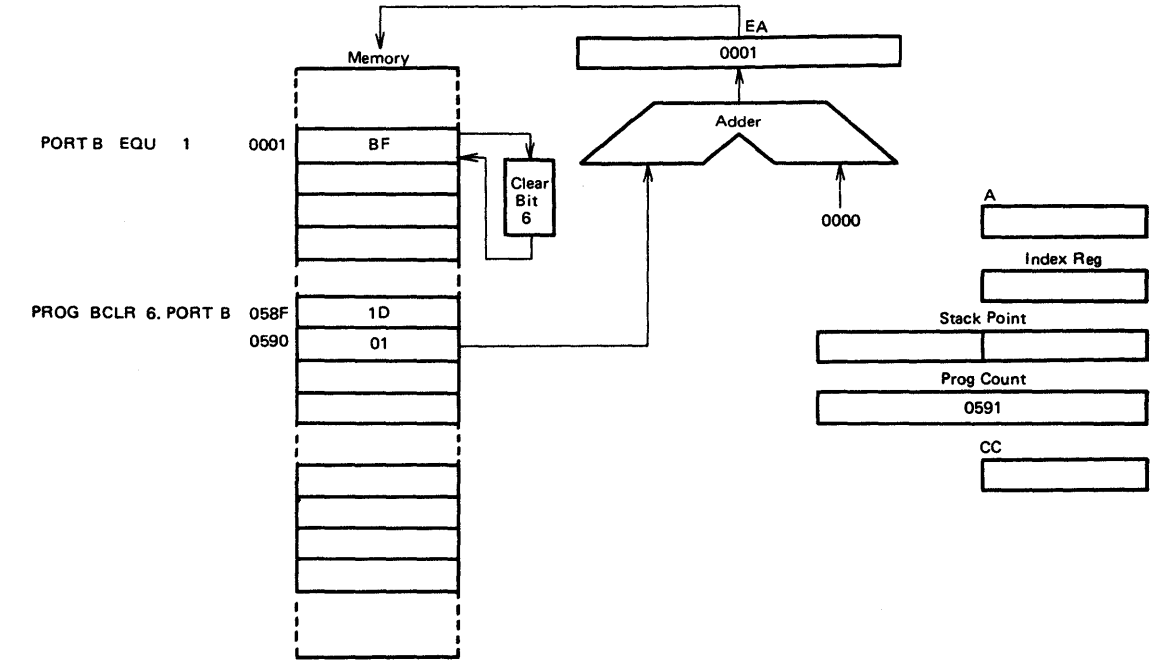


Figure 23 Bit Set/Clear Addressing Example

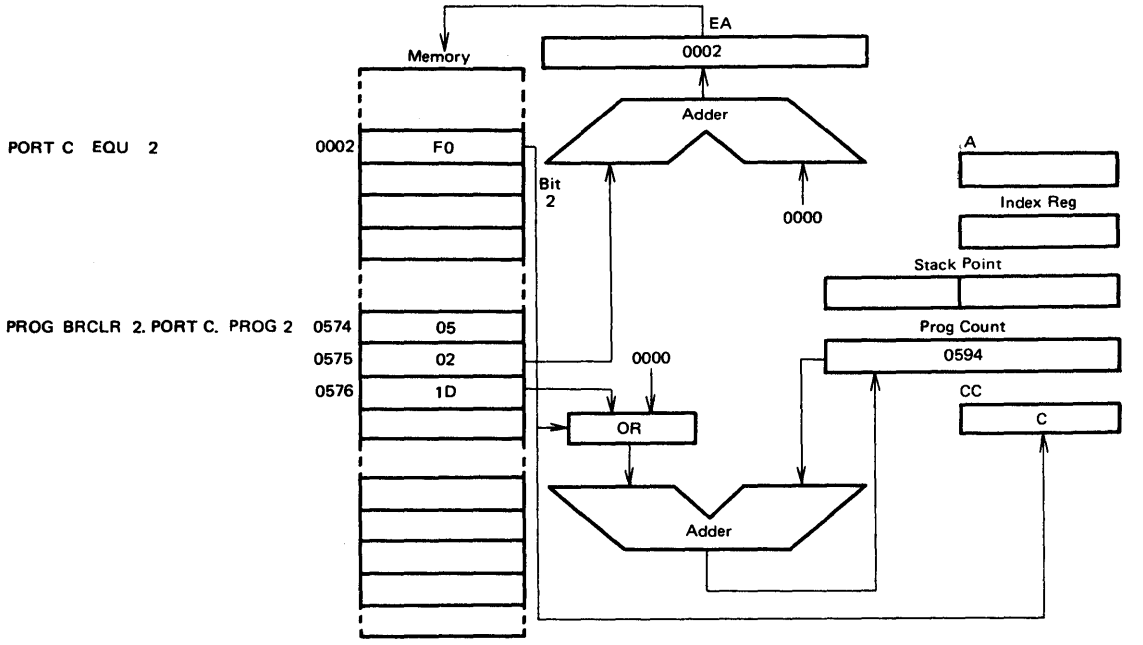


Figure 24 Bit Test and Branch Addressing Example

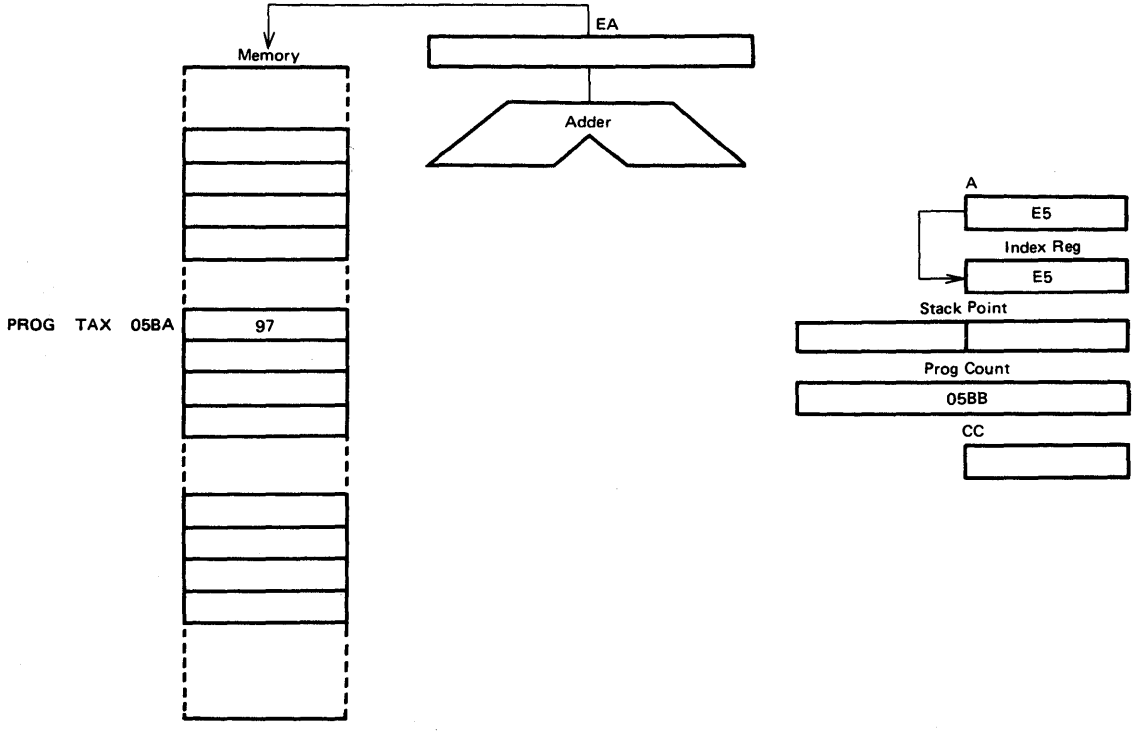


Figure 25 Implied Addressing Example

Table 2 Register/Memory Instructions

Function	Mnemonic	Addressing Modes																	
		Immediate			Direct			Extended			Indexed (No Offset)			Indexed (8-Bit Offset)			Indexed (16-Bit Offset)		
		Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Load A from Memory	LDA	A6	2	2	B6	2	4	C6	3	5	F6	1	4	E6	2	5	D6	3	6
Load X from Memory	LDX	AE	2	2	BE	2	4	CE	3	5	FE	1	4	EE	2	5	DE	3	6
Store A in Memory	STA	–	–	–	B7	2	5	C7	3	6	F7	1	5	E7	2	6	D7	3	7
Store X in Memory	STX	–	–	–	BF	2	5	CF	3	6	FF	1	5	EF	2	6	DF	3	7
Add Memory to A	ADD	AB	2	2	BB	2	4	CB	3	5	FB	1	4	EB	2	5	DB	3	6
Add Memory and Carry to A	ADC	A9	2	2	B9	2	4	C9	3	5	F9	1	4	E9	2	5	D9	3	6
Subtract Memory	SUB	A0	2	2	B0	2	4	C0	3	5	F0	1	4	E0	2	5	D0	3	6
Subtract Memory from A with Borrow	SBC	A2	2	2	B2	2	4	C2	3	5	F2	1	4	E2	2	5	D2	3	6
AND Memory to A	AND	A4	2	2	B4	2	4	C4	3	5	F4	1	4	E4	2	5	D4	3	6
OR Memory with A	ORA	AA	2	2	BA	2	4	CA	3	5	FA	1	4	EA	2	5	DA	3	6
Exclusive OR Memory with A	EOR	A8	2	2	B8	2	4	C8	3	5	F8	1	4	E8	2	5	D8	3	6
Arithmetic Compare A with Memory	CMP	A1	2	2	B1	2	4	C1	3	5	F1	1	4	E1	2	5	D1	3	6
Arithmetic Compare X with Memory	CPX	A3	2	2	B3	2	4	C3	3	5	F3	1	4	E3	2	5	D3	3	6
Bit Test Memory with A (Logical Compare)	BIT	A5	2	2	B5	2	4	C5	3	5	F5	1	4	E5	2	5	D5	3	6
Jump Unconditional	JMP	–	–	–	BC	2	3	CC	3	4	FC	1	3	EC	2	4	DC	3	5
Jump to Subroutine	JSR	–	–	–	BD	2	7	CD	3	8	FD	1	7	ED	2	8	DD	3	9

Table 3 Read/Modify/Write Instructions

Function	Mnemonic	Addressing Modes														
		Implied (A)			Implied (X)			Direct			Indexed (No Offset)			Indexed (8-Bit Offset)		
		Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Increment	INC	4C	1	4	5C	1	4	3C	2	6	7C	1	6	6C	2	7
Decrement	DEC	4A	1	4	5A	1	4	3A	2	6	7A	1	6	6A	2	7
Clear	CLR	4F	1	4	5F	1	4	3F	2	6	7F	1	6	6F	2	7
Complement	COM	43	1	4	53	1	4	33	2	6	73	1	6	63	2	7
Negate (2's Complement)	NEG	40	1	4	50	1	4	30	2	6	70	1	6	60	2	7
Rotate Left Thru Carry	ROL	49	1	4	59	1	4	39	2	6	79	1	6	69	2	7
Rotate Right Thru Carry	ROR	46	1	4	56	1	4	36	2	6	76	1	6	66	2	7
Logical Shift Left	LSL	48	1	4	58	1	4	38	2	6	78	1	6	68	2	7
Logical Shift Right	LSR	44	1	4	54	1	4	34	2	6	74	1	6	64	2	7
Arithmetic Shift Right	ASR	47	1	4	57	1	4	37	2	6	77	1	6	67	2	7
Arithmetic Shift Left	ASL	48	1	4	58	1	4	38	2	6	78	1	6	68	2	7
Test for Negative or Zero	TST	4D	1	4	5D	1	4	3D	2	6	7D	1	6	6D	2	7

Table 4 Branch Instructions

Function	Mnemonic	Relative Addressing Mode		
		Op Code	# Bytes	# Cycles
Branch Always	BRA	20	2	4
Branch Never	BRN	21	2	4
Branch IF Higher	BHI	22	2	4
Branch IF Lower or Same	BLS	23	2	4
Branch IF Carry Clear	BCC	24	2	4
(Branch IF Higher or Same)	(BHS)	24	2	4
Branch IF Carry Set	BCS	25	2	4
(Branch IF Lower)	(BLO)	25	2	4
Branch IF Not Equal	BNE	26	2	4
Branch IF Equal	BEQ	27	2	4
Branch IF Half Carry Clear	BHCC	28	2	4
Branch IF Half Carry Set	BHCS	29	2	4
Branch IF Plus	BPL	2A	2	4
Branch IF Minus	BMI	2B	2	4
Branch IF Interrupt Mask Bit is Clear	BMC	2C	2	4
Branch IF Interrupt Mask Bit is Set	BMS	2D	2	4
Branch IF Interrupt Line is Low	BIL	2E	2	4
Branch IF Interrupt Line is High	BIH	2F	2	4
Branch to Subroutine	BSR	AD	2	8

Table 5 Bit Manipulation Instructions

Function	Mnemonic	Addressing Modes					
		Bit Set/Clear			Bit Test and Branch		
		Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Branch IF Bit n is set	BRSET n (n=0 ..... 7)	—	—	—	2•n	3	10
Branch IF Bit n is clear	BRCLR n (n=0 .....7)	—	—	—	01+2•n	3	10
Set Bit n	BSET n (n=0 ..... 7)	10+2•n	2	7	—	—	—
Clear bit n	BCLR n (n=0 ..... 7)	11+2•n	2	7	—	—	—

Table 6 Control Instructions

Function	Mnemonic	Implied		
		Op Code	# Bytes	# Cycles
Transfer A to X	TAX	97	1	2
Transfer X to A	TXA	9F	1	2
Set Carry Bit	SEC	99	1	2
Clear Carry Bit	CLC	98	1	2
Set Interrupt Mask Bit	SEI	9B	1	2
Clear Interrupt Mask Bit	CLI	9A	1	2
Software Interrupt	SWI	83	1	11
Return from Subroutine	RTS	81	1	6
Return from Interrupt	RTI	80	1	9
Reset Stack Pointer	RSP	9C	1	2
No-Operation	NOP	9D	1	2

Table 7 Instruction Set

Mnemonic	Addressing Modes										Condition Code				
	Implied	Immediate	Direct	Extended	Relative	Indexed (No Offset)	Indexed (8 Bits)	Indexed (16 Bits)	Bit Set/ Clear	Bit Test & Branch	H	I	N	Z	C
ADC		x	x	x		x	x	x			∧	•	∧	∧	∧
ADD		x	x	x		x	x	x			∧	•	∧	∧	∧
AND		x	x	x		x	x	x			•	•	∧	∧	•
ASL	x		x			x	x				•	•	∧	∧	∧
ASR	x		x			x	x				•	•	∧	∧	∧
BCC					x						•	•	•	•	•
BCLR									x		•	•	•	•	•
BCS					x						•	•	•	•	•
BEQ					x						•	•	•	•	•
BHCC					x						•	•	•	•	•
BHCS					x						•	•	•	•	•
BHI					x						•	•	•	•	•
BHS					x						•	•	•	•	•
BIH					x						•	•	•	•	•
BIL					x						•	•	•	•	•
BIT		x	x	x		x	x	x			•	•	∧	∧	•
BLO					x						•	•	•	•	•
BLS					x						•	•	•	•	•
BMC					x						•	•	•	•	•
BMI					x						•	•	•	•	•
BMS					x						•	•	•	•	•
BNE					x						•	•	•	•	•
BPL					x						•	•	•	•	•
BRA					x						•	•	•	•	•
BRN					x						•	•	•	•	•
BRCLR										x	•	•	•	•	∧
BRSET										x	•	•	•	•	∧
BSET									x		•	•	•	•	•
BSR					x						•	•	•	•	•
CLC	x										•	•	•	•	0
CLI	x										•	0	•	•	•
CLR	x		x			x	x				•	•	0	1	•
CMP		x	x	x		x	x	x			•	•	∧	∧	∧
COM	x		x			x	x				•	•	∧	∧	1
CPX		x	x	x		x	x	x			•	•	∧	∧	∧
DEC	x		x			x	x				•	•	∧	∧	•
EOR		x	x	x		x	x	x			•	•	∧	∧	•
INC	x		x			x	x				•	•	∧	∧	•
JMP			x	x		x	x	x			•	•	•	•	•
JSR			x	x		x	x	x			•	•	•	•	•
LDA		x	x	x		x	x	x			•	•	∧	∧	•
LDX		x	x	x		x	x	x			•	•	∧	∧	•

(to be continued)

Condition Code Symbols:
- H Half Carry (From Bit 3)
- I Interrupt Mask
- N Negative (Sign Bit)
- Z Zero
- C Carry Borrow
- ∧ Test and Set if True, Cleared Otherwise
- • Not Affected

Table 7 Instruction Set

Mnemonic	Addressing Modes										Condition Code				
	Implied	Immediate	Direct	Extended	Relative	Indexed (No Offset)	Indexed (8 Bits)	Indexed (16 Bits)	Bit Set/ Clear	Bit Test & Branch	H	I	N	Z	C
LSL	x		x			x	x				•	•	∧	∧	∧
LSR	x		x			x	x				•	•	0	∧	∧
NEG	x		x			x	x				•	•	∧	∧	∧
NOP	x										•	•	•	•	•
ORA		x	x	x		x	x	x			•	•	∧	∧	•
ROL	x		x			x	x				•	•	∧	∧	∧
ROR	x		x			x	x				•	•	∧	∧	∧
RSP	x										•	•	•	•	•
RTI	x										?	?	?	?	?
RTS	x										•	•	•	•	•
SBC		x	x	x		x	x	x			•	•	∧	∧	∧
SEC	x										•	•	•	•	1
SEI	x										•	1	•	•	•
STA			x	x		x	x	x			•	•	∧	∧	•
STX			x	x		x	x	x			•	•	∧	∧	•
SUB		x	x	x		x	x	x			•	•	∧	∧	∧
SWI	x										•	1	•	•	•
TAX	x										•	•	•	•	•
TST	x		x			x	x				•	•	∧	∧	•
TXA	x										•	•	•	•	•

Condition Code Symbols:

- H Half Carry (From Bit 3)
- I Interrupt Mask
- N Negative (Sign Bit)
- Z Zero
- C Carry/Borrow
- ∧ Test and Set if True, Cleared Otherwise
- • Not Affected
- ? Load CC Register From Stack

Table 8 Opcode Map

	Bit Manipulation		Branch	Read/Modify/Write					Control		Register/Memory						
	Test & Branch	Set/ Clear	Rel	DIR	A	X	,X1	,X0	IMP	IMP	IMM	DIR	EXT	,X2	,X1	,X0	
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	← HIGH
0	BRSET0	BSET0	BRA	NEG					RTI*	–	SUB						0
1	BRCLR0	BCLR0	BRN	–					RTS*	–	CMP						1
2	BRSET1	BSET1	BHI	–					–	–	SBC						2
3	BRCLR1	BCLR1	BLS	COM					SWI*	–	CPX						3 L
4	BRSET2	BSET2	BCC	LSR					–	–	AND						4 O
5	BRCLR2	BCLR2	BCS	–					–	–	BIT						5 W
6	BRSET3	BSET3	BNE	ROR					–	–	LDA						6
7	BRCLR3	BCLR3	BEQ	ASR					–	TAX	–	STA(+1)					7
8	BRSET4	BSET4	BHCC	LSL/ASL					–	CLC	EOR						8
9	BRCLR4	BCLR4	BHCS	ROL					–	SEC	ADC						9
A	BRSET5	BSET5	BPL	DEC					–	CLI	ORA						A
B	BRCLR5	BCLR5	BMI	–					–	SEI	ADD						B
C	BRSET6	BSET6	BMC	INC					–	RSP	–	JMP(–1)					C
D	BRCLR6	BCLR6	BMS	TST					–	NOP	BSR*	JSR(–3)					D
E	BRSET7	BSET7	BIL	–					–	–	LDX						E
F	BRCLR7	BCLR7	BIH	CLR					–	TXA	–	STX(+1)					F
	3/10	2/7	2/4	2/6	1/4	1/4	2/7	1/6	1/*	1/2	2/2	2/4	3/5	3/6	2/5	1/4	

(NOTE) 1. Undefined opcodes are marked with "–".
2. The number at the bottom of each column denote the number of bytes and the number of cycles required (Bytes/Cycles). Mnemonics followed by a "*" require a different number of cycles as follows:

RTI	9
RTS	6
SWI	11
BSR	8

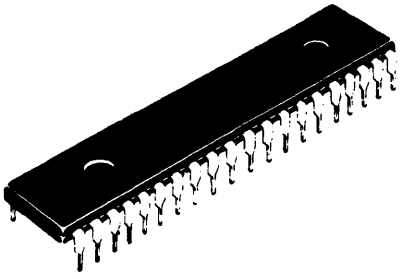
3. ( ) indicate that the number in parenthesis must be added to the cycle count for that instruction.

HD6805U1
MCU (Microcomputer Unit)

The HD6805U1 is the 8-bit Microcomputer Unit (MCU) which contains a CPU, on-chip clock, ROM, RAM, I/O and timer. It is designed for the user who needs an economical microcomputer with the proven capabilities of the HD6800-based instruction set.

The following are some of the hardware and software highlights of the MCU.

HD6805U1P



(DP-40)

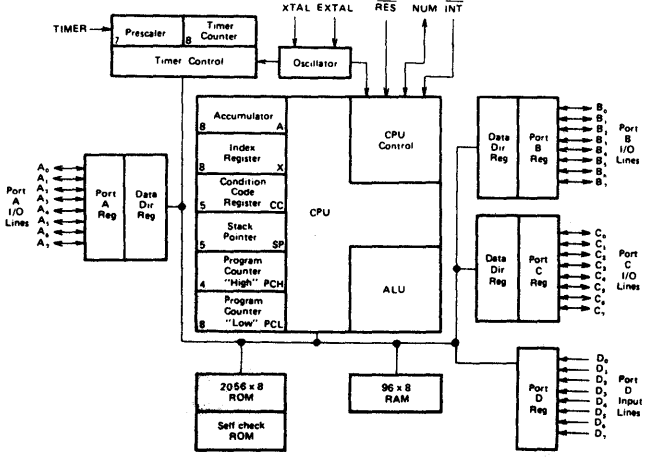
■ HARDWARE FEATURES

- 8-Bit Architecture
- 96 Bytes of RAM
- Memory Mapped I/O
- 2056 Bytes of User ROM
- Internal 8-Bit Timer with 7-Bit Prescaler
- Vectored Interrupts – External and Timer
- 24 I/O Ports + 8 Input Port
  (8 Lines LED Compatible; 7 Voltage Comparator Inputs)
- On-Chip Clock Circuit
- Self-Check Mode
- Master Reset
- Low Voltage Inhibit
- Complete Development System Support by Evaluation Kit
- 5 Vdc Single Supply

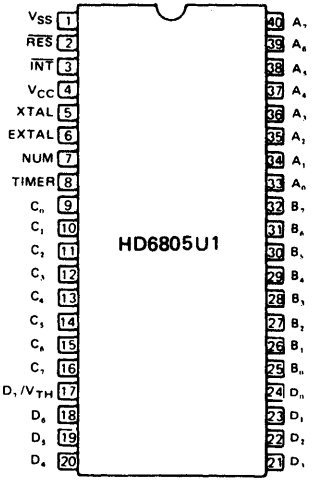
■ SOFTWARE FEATURES

- Similar to HD6800
- Byte Efficient Instruction Set
- Easy to Program
- True Bit Manipulation
- Bit Test and Branch Instructions
- Versatile Interrupt Handing
- Powerful Indexed Addressing for Tables
- Full Set of Conditional Branches
- Memory Usable as Registers/Flags
- Single Instruction Memory Examine/Change
- 10 Powerful Addressing Modes
- All Addressing Modes Apply to ROM, RAM and I/O Compatible Instruction Set with MC6805P2

■ BLOCK DIAGRAM



■ PIN ARRANGEMENT



(Top View)

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC} *	-0.3 ~ +7.0	V
Input Voltage (EXCEPT TIMER)	V_{in} *	-0.3 ~ +7.0	V
Input Voltage (TIMER)		-0.3 ~ +12.0	V
Operating Temperature	T_{opr}	0 ~ +70	°C
Storage Temperature	T_{stg}	- 55 ~ +150	°C

* With respect to V_{SS} (SYSTEM GND)

(NOTE) Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ ELECTRICAL CHARACTERISTICS

● DC CHARACTERISTICS (V_{CC}=5.25V ± 0.5V, V_{SS}=GND, Ta=0~+70°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Input "High" Voltage	$\overline{RES}$	V_{IH}		4.0	–	V_{CC}	V
	$\overline{INT}$			3.0	–	V_{CC}	V
	All Other			2.0	–	V_{CC}	V
Input "High" Voltage Timer	Timer Mode			2.0	–	V_{CC}	V
	Self-Check Mode			9.0	–	11.0	V
Input "Low" Voltage	$\overline{RES}$	V_{IL}		-0.3	–	0.8	V
	$\overline{INT}$			-0.3	–	0.8	V
	XTAL(Crystal Mode)			-0.3	–	0.6	V
	All Other			-0.3	–	0.8	V
Power Dissipation		P_D		–	400	700	mW
Low Voltage Recover		LVR		–	–	4.75	V
Low Voltage Inhibit		LVI		–	4.0	–	V
Input Leak Current	TIMER	I_{IL}	V_{in}=0.4V~V_{CC}	-20	–	20	μA
	$\overline{INT}$			-50	–	50	μA
	XTAL(Crystal Mode)			-1200	–	0	μA

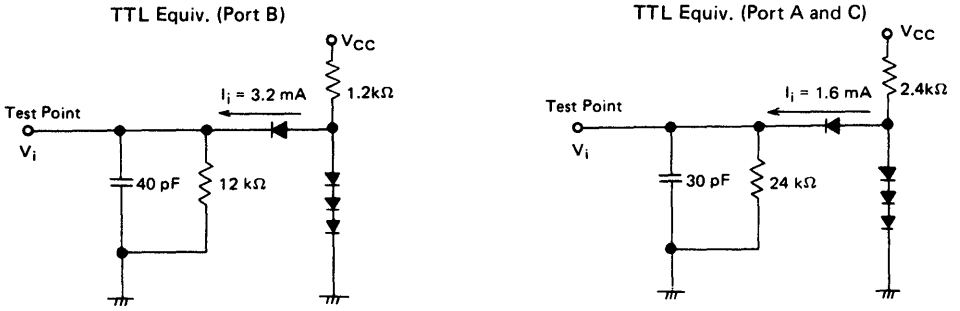
● AC CHARACTERISTICS (V_{CC}=5.25V ± 0.5V, V_{SS}=GND, Ta=0 ~ +70°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Clock Frequency		f_{cl}		0.4	–	4.0	MHz
Cycle Time		t_{cyc}		1.0	–	10	μs
Oscillation Frequency (External Resister Mode)		f_{EXT}	R_{CP}=15.0kΩ±1%	2.7	–	4.0	MHz
$\overline{INT}$ Pulse Width		t_{IWL}		t_{cyc}+250	–	–	ns
$\overline{RES}$ Pulse Width		t_{RWL}		t_{cyc}+250	–	–	ns
TIMER Pulse Width		t_{TWL}		t_{cyc}+250	–	–	ns
Oscillation Start-up Time (Crystal Mode)		t_{OSC}	C_L=22pF±20%, R_S=60Ω max.	–	–	100	ms
Delay Time Reset		t_{RHL}	External Cap. = 2.2 μF	100	–	–	ms
Input Capacitance	EXTAL	C_{in}	V_{in}=0V	–	25	35	pF
	All Other			–	6	10	pF

● PORT ELECTRICAL CHARACTERISTICS (V_{CC} = 5.25V ± 0.5V, V_{SS} = GND, Ta = 0 ~ +70°C unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Output "High" Voltage	Port A	V_{OH}	I_{OH} = −10 μA	3.5	–	–	V
			I_{OH} = −100 μA	2.4	–	–	V
	Port B		I_{OH} = −200 μA	2.4	–	–	V
			I_{OH} = −1 mA	1.5	–	–	V
	Port C		I_{OH} = −100 μA	2.4	–	–	V
Output "Low" Voltage	Port A and C	V_{OL}	I_{OL} = 1.6 mA	–	–	0.4	V
	Port B		I_{OL} = 3.2 mA	–	–	0.4	V
			I_{OL} = 10 mA	–	–	1.0	V
Input "High" Voltage	Port A, B, C, and D*	V_{IH}		2.0	–	V_{CC}	V
Input "Low" Voltage		V_{IL}		−0.3	–	0.8	V
Input Leak Current	Port A	I_{IL}	V_{in} = 0.8V	−500	–	–	μA
			V_{in} = 2V	−300	–	–	μA
	Port B, C, and D		V_{in} = 0.4V ~ V_{CC}	− 20	–	20	μA
Input "High" Voltage	Port D** (D_0 ~ D_6)	V_{IH}		–	V_{TH}+0.2	—	V
Input "Low" Voltage	Port D** (D_0 ~ D_6)	V_{IL}		–	V_{TH}-0.2	—	V
Threshold Voltage	Port D**(D_7)	V_{TH}		0	—	0.8×V_{CC}	V

* Port D as digital input
** Port D as analog input



(NOTE) 1. Load capacitance includes the floating capacitance of the probe and the jig etc.
2. All diodes are 1S2074Ⓗ or equivalent.

Figure 1 Bus Timing Test Loads

■ SIGNAL DESCRIPTION

The input and output signals for the MCU shown in PIN ARRANGEMENT are described in the following paragraphs.

● V_{CC} and V_{SS}

Power is supplied to the MCU using these two pins. V_{CC} is +5.25V ±0.5V. V_{SS} is the ground connection.

● $\overline{INT}$

This pin provides the capability for applying an external interrupt to the MCU Refer to INTERRUPTS for additional information.

● XTAL and EXTAL

These pins provide control input for the on-chip clock circuit. A crystal (AT cut, 4 MHz maximum) or a resistor can be connected to these pins to provide the internal oscillator with varying degrees of stability. Refer to INTERNAL OSCILLATOR OPTIONS for recommendations about these inputs.

● TIMER

This pin allows an external input to be used to decrement the internal timer circuitry. Refer to TIMER for additional information about the timer circuitry.

● $\overline{RES}$

This pin allows resetting of the MCU at times other than the automatic resetting capability already in the MCU. Refer to RESETS for additional information.

● NUM

This pin is not for user application and should be connected to ground.

- **Input/Output Lines ($A_0 \sim A_7$, $B_0 \sim B_7$, $C_0 \sim C_7$)**

These 24 lines are arranged into three 8-bit ports (A, B and C). All lines are programmable as either inputs or outputs under software control of the data direction registers. Refer to INPUTS/OUTPUTS for additional information.

- **Input Lines ($D_0 \sim D_7$)**

These are 8-bit input lines, which has two functions. Firstly, these become TTL compatible inputs, by reading $003 address. The other function of them is 7 Voltage comparators, by reading $007 address. Please refer to INPUT PORT for more detail.

■ MEMORY

The MCU memory is configured as shown in Figure 2. During the processing of an interrupt, the contents of the MCU registers are pushed onto the stack in the order shown in Figure 3. Since the stack pointer decrements during pushes, the low order byte (PCL) of the program counter is stacked first; then the high order four bits (PCH) are stacked. This ensures that the program counter is loaded correctly as the stack pointer increments when it pulls data from the stack. A subroutine call will cause only the program counter (PCH, PCL) contents to be pushed onto the stack.

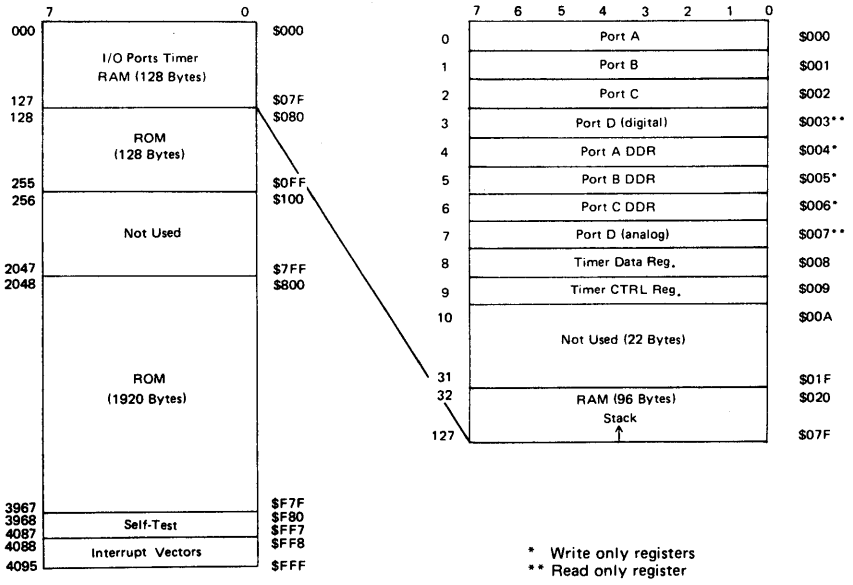


Figure 2 MCU Memory Configuration

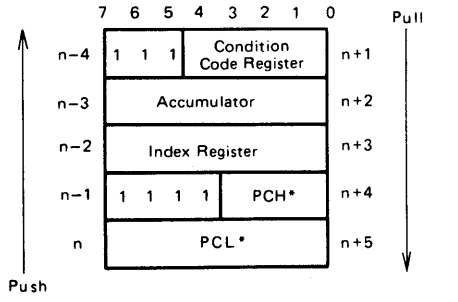


Figure 3 Interrupt Stacking Order

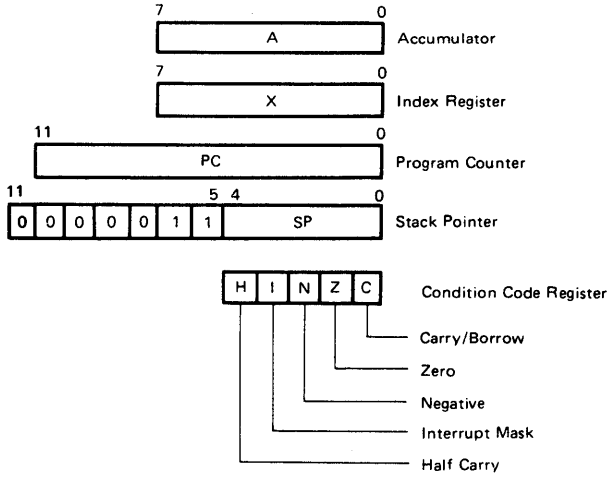


Figure 4 Programming Model

■ REGISTERS

The MCU has five registers available to the programmer. They are shown in Figure 4 and are explained in the following paragraphs.

- **Accumulator (A)**

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.

- **Index Register (X)**

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit address that may be added to an offset value to create an effective address. The index register can also be used for limited calculations and data manipulations when using read/modify/write instructions. When not required by a code sequence being executed, the index register can be used as a temporary storage area.

- **Program Counter (PC)**

The program counter is a 12-bit register that contains the address of the next instruction to be executed.

- **Stack Pointer (SP)**

The stack pointer is a 12-bit register that contains the address of the next free location on the stack. Initially, the stack pointer is set to location $07F and is decremented as data is being pushed onto the stack and incremented as data is being pulled from the stack. The six most significant bits of the stack pointer are permanently set to 0000011. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set to location $07F. Subroutines and interrupts may be nested down to location $061 which allows the programmer to use up to 15 levels of subroutine calls.

- **Condition Code Register (CC)**

The condition code register is a 5-bit register in which each bit is used to indicate or flag the results of the instruction just executed. These bits can be individually tested by a program and specific action taken as a result of their state. Each individual condition code register bit is explained in the following paragraphs.

- **Half Carry (H)**

Used during arithmetic operations (ADD and ADC) to indicate that a carry occurred between bits 3 and 4.

- **Interrupt (I)**

This bit is set to mask the timer and external interrupt ($\overline{INT}$). If an interrupt occurs while this bit is set it is latched and will be processed as soon as the interrupt bit is reset.

- **Negative (N)**

Used to indicate that the result of the last arithmetic, logical or data manipulation was negative (bit 7 in result equal to a logical one).

- **Zero (Z)**

Used to indicate that the result of the last arithmetic, logical or data manipulation was zero.

- **Carry/Borrow (C)**

Used to indicate that a carry or borrow out of the arithmetic logic unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions, shifts, and rotates.

■ TIMER

The MCU timer circuitry is shown in Figure 5. The 8-bit counter is loaded under program control and counts down toward zero as soon as the clock input is applied. When the timer reaches zero the timer interrupt request bit (bit 7) in the timer control register is set. The MCU responds to this interrupt by saving the present MCU state in the stack, fetching the timer interrupt vector from locations $FF8 and $FF9 and executing the interrupt routine. The timer interrupt can be masked by setting the timer interrupt mask bit (bit 6) in the time control register. The interrupt bit (I bit) in the condition code register will also prevent a timer interrupt from being processed.

The clock input to the timer can be from an external source applied to the TIMER input pin or it can be the internal ϕ_2 signal. Note that when the ϕ_2 signal is used as the source it can be gated by an input applied to the TIMER input pin allowing the user to easily perform pulse-width measurements. The source of the clock input is one of the options that has to be specified before manufacture of the MCU. A prescaler option can be applied to the clock input that extends the timing interval up to a maximum of 128 counts before being applied to

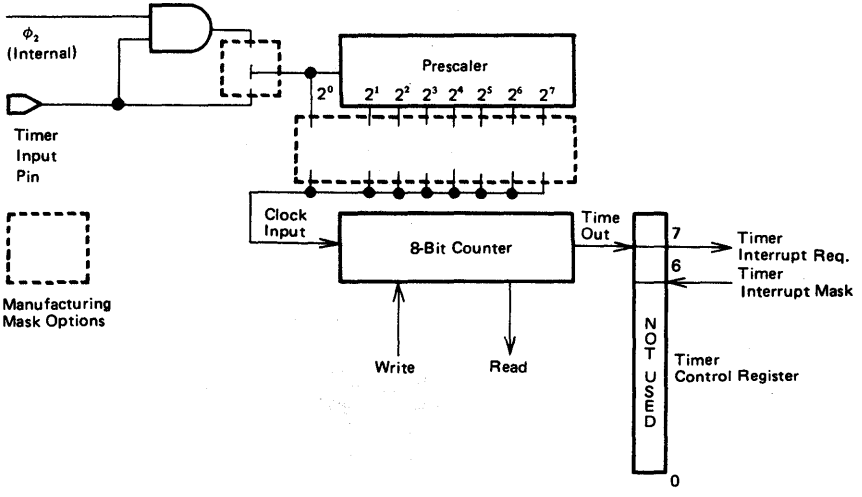


Figure 5 Timer Block Diagram

the counter. This prescaling option must also be specified before manufacturing begins. The timer continues to count past zero and its present count can be monitored at any time by monitoring the timer data register. This allows a program to determine the length of time since a timer interrupt has occured and not disturb the counting process.

The Timer Data Register is 8-bit Read/Write Register with address $008 on Memory-Map. This Timer Data Register and the prescaler are initialize with all logical ones at Reset time.

The Timer Interrupt Request bit (bit 7 of Timer Control Register) is set to one by hardware when timer count reaches zero, and is cleared by program or by hardware reset. The bit 6 of Timer Control Register is writable by program. Both of those bits can be read by MPU.

■ SELF CHECK

The self check capability of the MCU provides an internal check to determine if the part is functional. Connect the MCU as shown in Figure 6 and monitor the output of port C bit 3 for an oscillation of approximately three hertz.

■ RESETS

The MCU can be reset three ways: by initial powerup, by the external reset input ($\overline{RES}$) and by an internal low voltage detect circuit, (mask option) see Figure 7. All the I/O port are initialized to Input mode (DDR's are cleared) during RESET.

Upon power up, a minimum of 100 milliseconds is needed before allowing the reset input to go "High".

This time allows the internal crystal oscillator to stabilize. Connecting a capacitor to the $\overline{RES}$ input as shown in Figure 8 will provide sufficient delay.

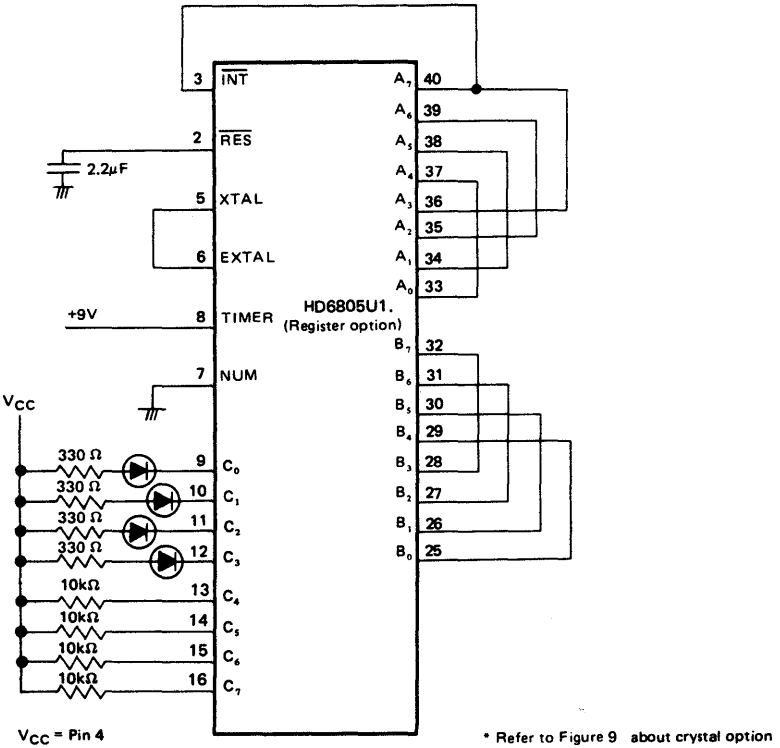


Figure 6 Self Check Connections

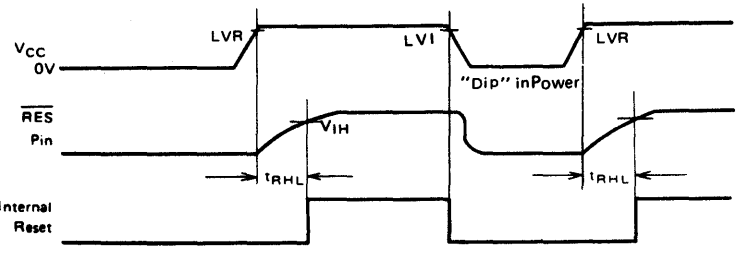


Figure 7 Power Up and $\overline{RES}$ Timing

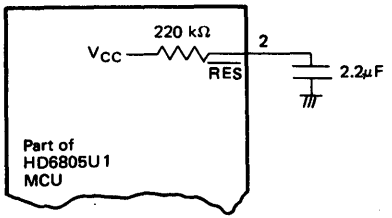


Figure 8 Power Up Reset Delay Circuit

■ INTERNAL OSCILLATOR OPTIONS

The internal oscillator circuit has been designed to require a minimum of external components. The use of a crystal (AT cut, 4 MHz max) or a resistor is sufficient to drive the internal oscillator with varying degrees of stability. A manufacturing mask option is available to provide better matching between the external components and the internal oscillator. The different connection methods are shown in Figure 9. Crystal specifications are given in Figure 10. A resistor selection graph is given in Figure 11.

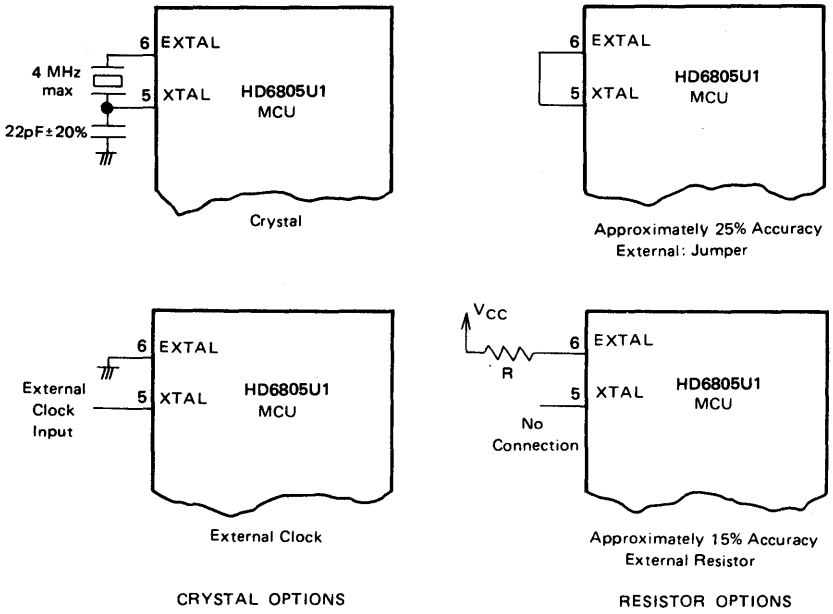
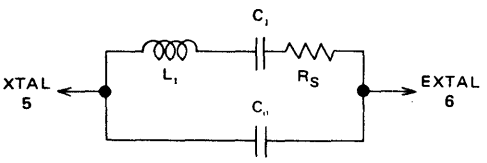


Figure 9 Internal Oscillator Options



AT – Cut Parallel Resonance Crystal
C_0 = 7 pF max.
f = 4 MHz
R_S = 60 Ω max.

Figure 10 Crystal Parameters

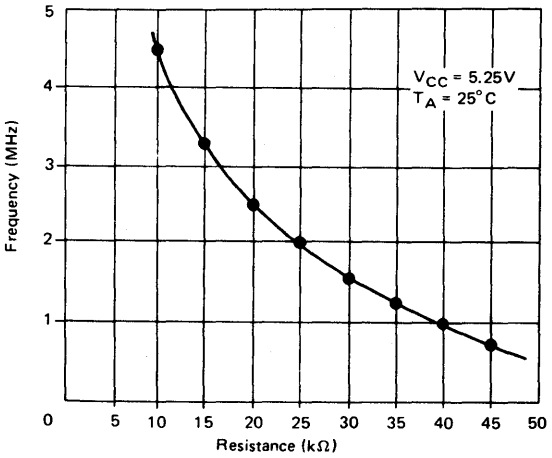


Figure 11 Typical Resistor Selection Graph

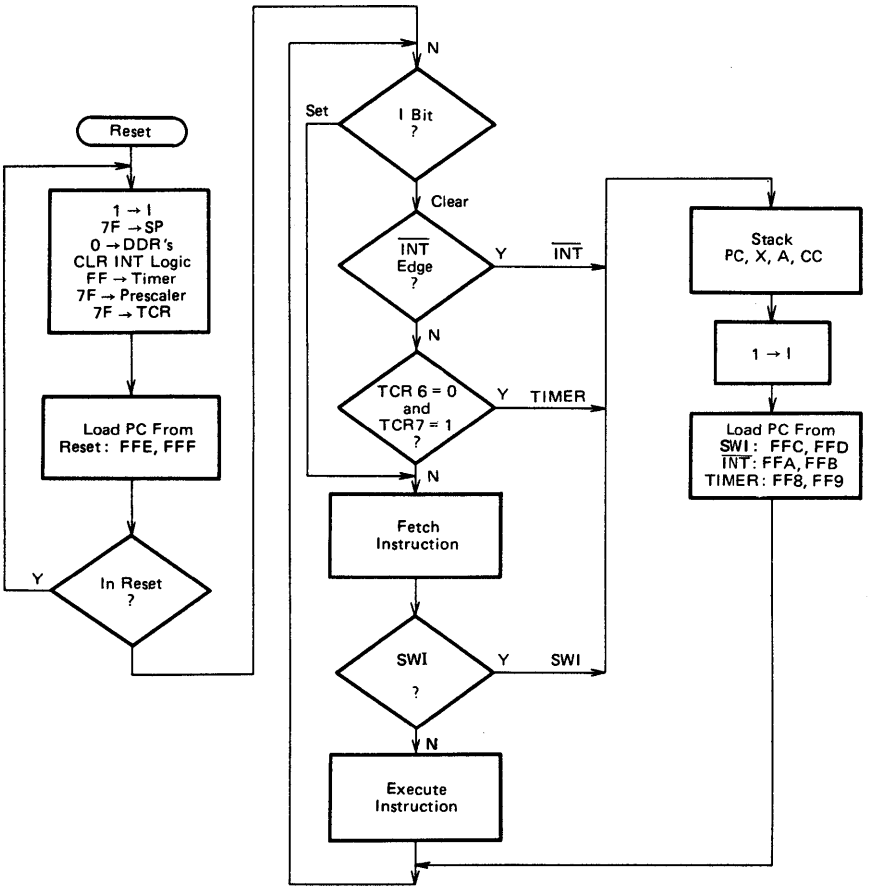
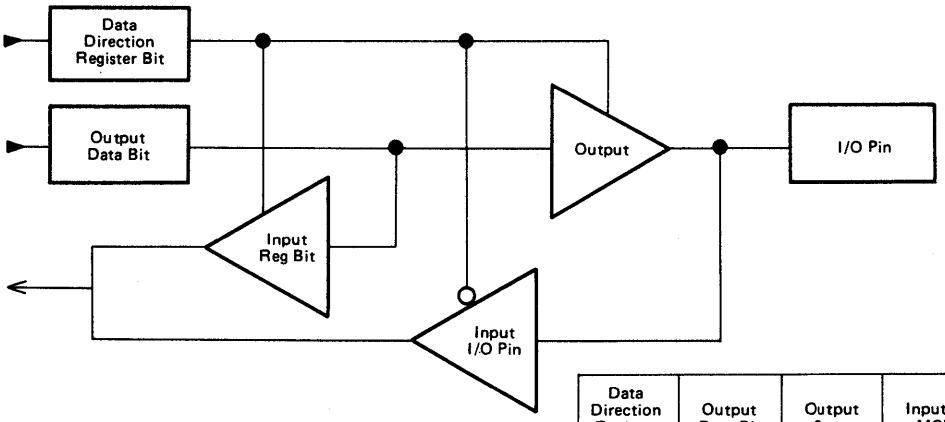


Figure 12 Interrupt Processing Flowchart



Data Direction Register Bit	Output Data Bit	Output State	Input to MCU
1	0	0	0
1	1	1	1
0	x	3-State	Pin

Figure 13 Typical Port I/O Circuitry

■ INTERRUPTS

The MCU can be interrupted three different ways: through the external interrupt ($\overline{INT}$) input pin, the internal timer interrupt request, and a software interrupt instruction (SWI). When any interrupt occurs, processing is suspended, the present MCU state is pushed onto the stack, the interrupt bit (I) in the condition code register is set, the address of the interrupt routine is obtained from the appropriate interrupt vector address, and the interrupt routine is executed. The interrupt service routines normally end with a return from interrupt (RTI) instruction which allows the MCU to resume processing of the program prior to the interrupt. Table 1 provides a listing of the interrupts, their priority, and the vector address that contain the starting address of the appropriate interrupt routine.

A flowchart of the interrupt processing sequence is given in Fig. 12.

Table 1 Interrupt Priorities

Interrupt	Priority	Vector Address
$\overline{RES}$	1	$FFE and $FFF
SWI	2	$FFC and $FFD
$\overline{INT}$	3	$FFA and $FFB
TIMER	4	$FF8 and $FF9

■ INPUT/OUTPUT

There are 24 input/output pins. All pins are programmable as either inputs or outputs under software control of the data direction registers. When programmed as outputs, all I/O pins read latched output data regardless of the logic level at the output pin due to output loading (see Fig. 13). When port B is programmed for outputs, it is capable of sinking 10 millamperes on each pin (V_{OL} = 1V max). All input/output lines are TTL compatible as both inputs and outputs. Port A lines are CMOS compatible as outputs while port B and C lines are CMOS compatible as inputs. Figure 14 provides some examples of port connections.

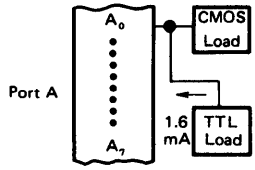
■ INPUT

Port D is 8-bit input port, which has two functions. One of them is usual digital signal input port and the other is voltage compare type input port. In the former case, the input data can be read by MPU at $003 address. In the latter case, D_7 (pin 17) is the input pin of V_{TH} (reference level), and the other seven input pins ($D_0 \sim D_6$) are analog level inputs, which are compared with V_{TH} (see Figure 15(a), (b)).

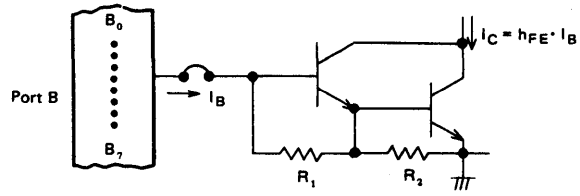
"1" or "0" signals appear at internal data bus, if the input levels are higher or lower respectively when $007 address is read. This function is effective in such case that unusual logic level inputs are used. A capacitive touch panel interface and a diode isolated keyboard interface are the examples. Figure 15(c) shows the application of Port D to A/D converter, and Figure 15(d) shows 3 levels inputs.

■ BIT MANIPULATION

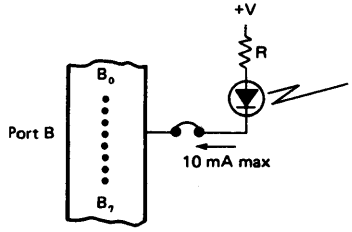
The MCU has the ability to set or clear any single random access memory or input/output bit (except the data direction registers) with a single instruction (BSET, BCLR). Any bit in the page zero read only memory can be tested, using the BRSET and BRCLR instructions, and the program branches as a result of its state. This capability to work with any bit in RAM, ROM or I/O allows the user to have individual flags in RAM or to handle single I/O bits as control lines. The example in Figure 16 illustrates the usefulness of the bit manipulation and test



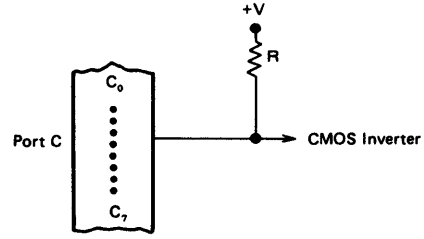
Port A Programmed as output(s) driving CMOS and TTL Load directly.
(a)



Port B Programmed as output(s) driving Darlington base directly.
(b)



Port B Programmed as output(s) driving LED(s) directly.
(c)

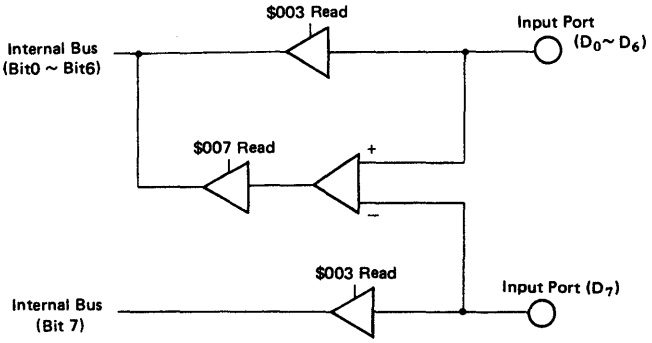


Port C Programmed as output(s) driving CMOS using external pull-up resistors.
(d)

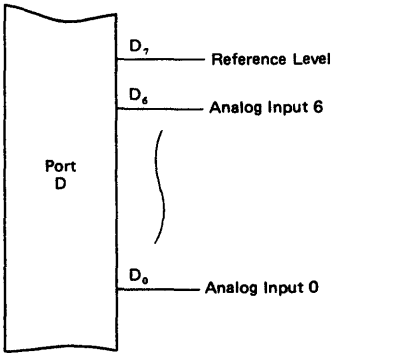
Figure 14 Typical Port Connections

instructions. Assume that bit 0 of port A is connected to a zero crossing detector circuit and that bit 1 of port A is connected to the trigger of a TRIAC which power the controlled hardware.

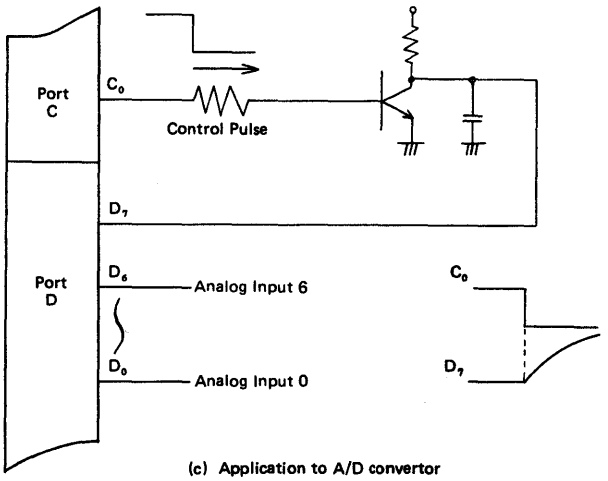
This program, which uses only seven ROM locations, provides turn-on of the TRIAC within 14 microseconds of the zero crossing. The timer could also be incorporated to provide turn-on at some later time which would permit pulse-width modulation of the controlled power.



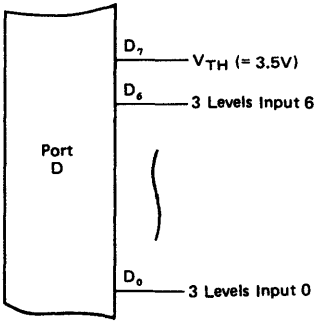
(a) The logic configuration of Port D



(b) Seven analog inputs and a reference level input of Port D



(c) Application to A/D convertor



(d) Application to 3 levels input

Input Voltage	($003)	($007)
0V ~ 0.8V	0	0
2.0V ~ 3.3V	1	0
3.7V ~ V_{CC}	1	1

Figure 15 Configuration and Application of Port D

```
        .
        .
        .
SELF 1  BRCLR 0, PORT A, SELF 1
        BSET 1, PORT A
        BCLR 1, PORT A
        .
        .
        .
        .
```

Figure 16 Bit Manipulation Example

■ ADDRESSING MODES

The MCU has ten addressing modes available for use by the programmer. They are explained and illustrated briefly in the following paragraphs.

- **Immediate**

Refer to Figure 17. The immediate addressing mode accesses constants which do not change during program execution. Such instructions are two bytes long. The effective address (EA) is the PC and the operand is fetched from the byte following the opcode.

- **Direct**

Refer to Figure 18. In direct addressing, the address of the operand is contained in the second byte of the instruction. Direct addressing allows the user to directly address the lowest 256 bytes in memory. All RAM space, I/O registers and 128 bytes of ROM are located in page zero to take advantage of this efficient memory addressing mode.

- **Extended**

Refer to Figure 19. Extended addressing is used to reference any location in memory space. The EA is the contents of the two bytes following the opcode. Extended addressing instructions are three bytes long.

- **Relative**

Refer to Figure 20. The relative addressing mode applies only to the branch instructions. In this mode the contents of the byte following the opcode is added to the program counter when the branch is taken. EA=(PC)+2+Rel. Rel is the contents of the location following the instruction opcode with bit 7 being the sign bit. If the branch is not taken Rel=0, when a branch takes place, the program goes to somewhere within the range of +129 bytes to -127 of the present instruction. These instructions are two bytes long.

- **Indexed (No Offset)**

Refer to Figure 21. This mode of addressing accesses the lowest 256 bytes of memory. These instructions are one byte long and their EA is the contents of the index register.

- **Indexed (8-bit Offset)**

Refer to Figure 22. The EA is calculated by adding the contents of the byte following the opcode to the contents of the index register. In this mode, 511 low memory locations are accessable. These instructions occupy two bytes.

- **Indexed (16-bit Offset)**

Refer to Figure 23. This addressing mode calculates the EA by adding the contents of the two bytes following the opcode to the index register. Thus, the entire memory space may be accessed. Instructions which use this addressing mode are three bytes long.

- **Bit Set/Clear**

Refer to Figure 24. This mode of addressing applies to instructions which can set or clear any bit on page zero. The lower three bits in the opcode specify the bit to be set or cleared while the byte following the opcode specifies the address in page zero.

- **Bit Test and Branch**

Refer to Figure 25. This mode of addressing applies to instructions which can test any bit in the first 256 locations ($00-$FF) and branch to any location relative to the PC. The byte to be tested is addressed by the byte following the opcode. The individual bit within that byte to be tested is addressed by the lower three bits of the opcode. The third byte is the relative address to be added to the program counter if the branch condition is met. These instructions are three bytes long. The value of the bit tested is written to the carry bit in the condition code register.

- **Implied**

Refer to Figure 26. The implied mode of addressing has no EA. All the information necessary to execute an instruction is contained in the opcode. Direct operations on the accumulator and the index register are included in this mode of addressing. In addition, control instructions such as SWI, RTI belong to this group. All implied addressing instructions are one byte long.

■ INSTRUCTION SET

The MCU has a set of 59 basic instructions. They can be divided into five different types: register/memory, read/modify/write, branch, bit manipulation, and control. The following paragraphs briefly explain each type. All the instructions within a given type are presented in individual tables.

- **Register/Memory Instructions**

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to Table 2.

- **Read/Modity/Write Instructions**

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read/modify/write instructions since it does not perform the write. Refer to Table 3.

- **Branch Instructions**

The branch instructions cause a branch from the program when a certain condition is met. Refer to Table 4.

- **Bit Manipulation Instructions**

These instructions are used on any bit in the first 256 bytes of the memory. One group either sets or clears. The other group performs the bit test and branch operations. Refer to Table 5.

- **Control Instructions**

The control instructions control the MCU operations during program execution. Refer to Table 6.

- **Alphabetical Listing**

The complete instruction set is given in alphabetical order in Table 7.

- **Opcode Map**

Table 8 is an opcode map for the instructions used on the MCU.

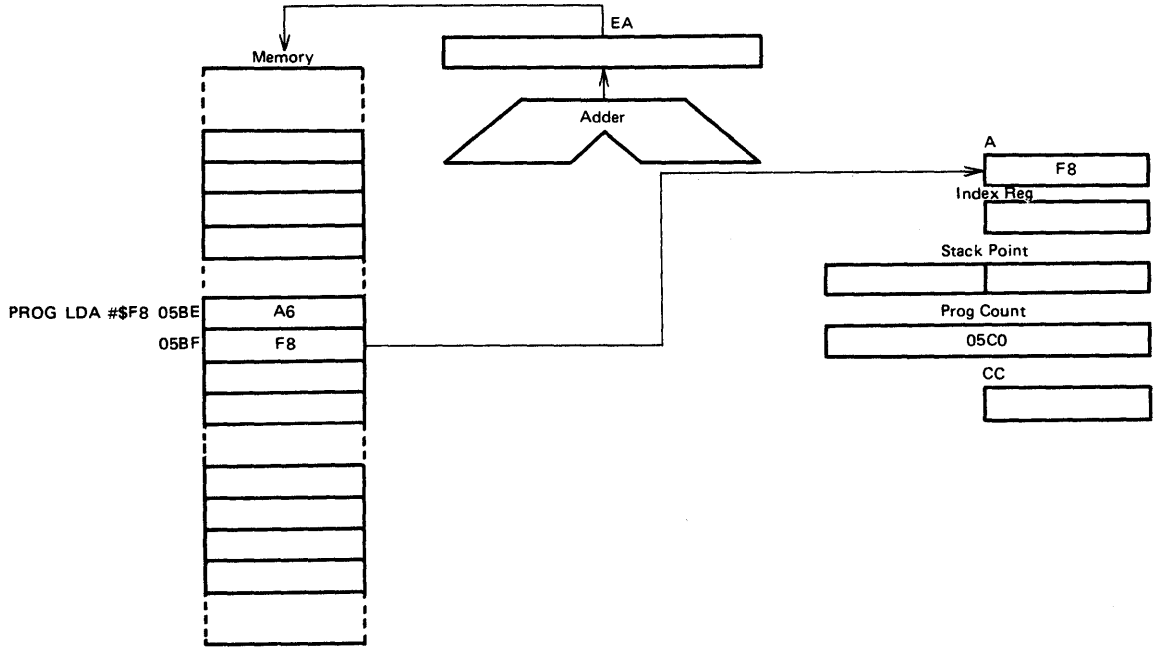


Figure 17 Immediate Addressing Example

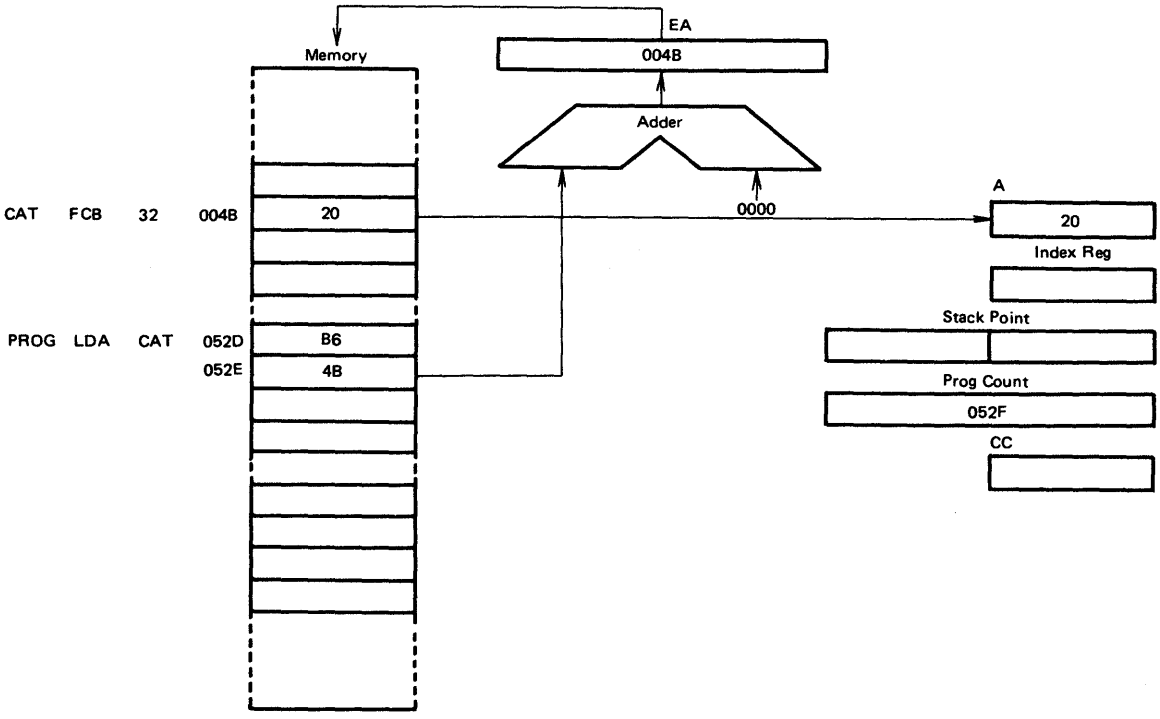


Figure 18 Direct Addressing Example

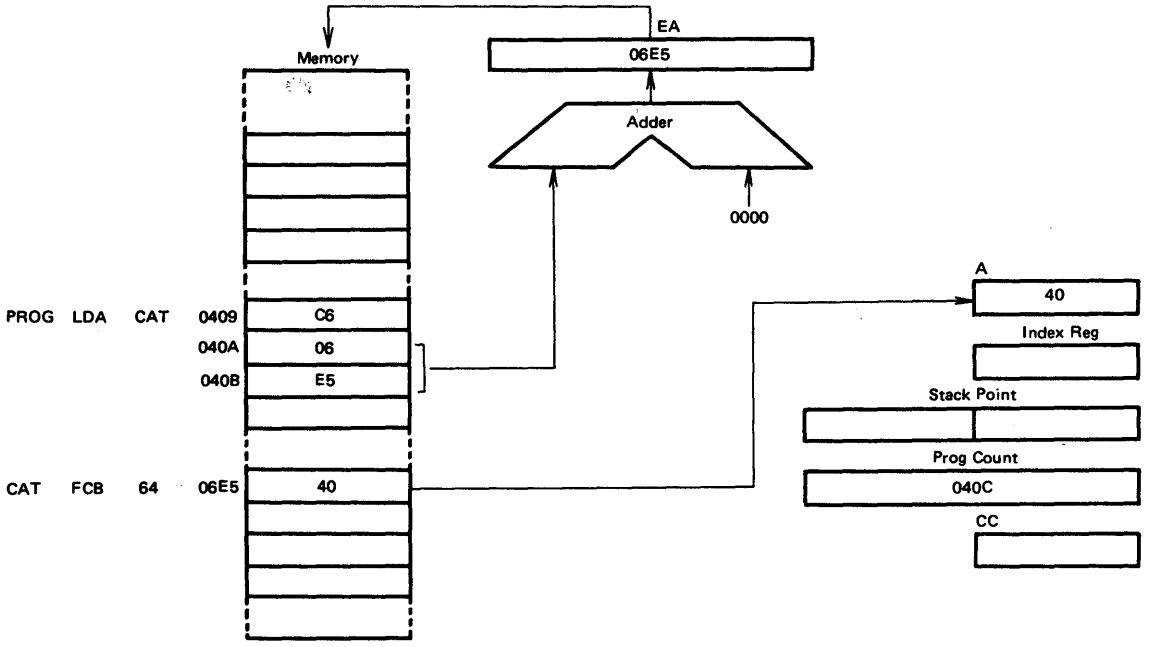


Figure 19 Extended Addressing Example

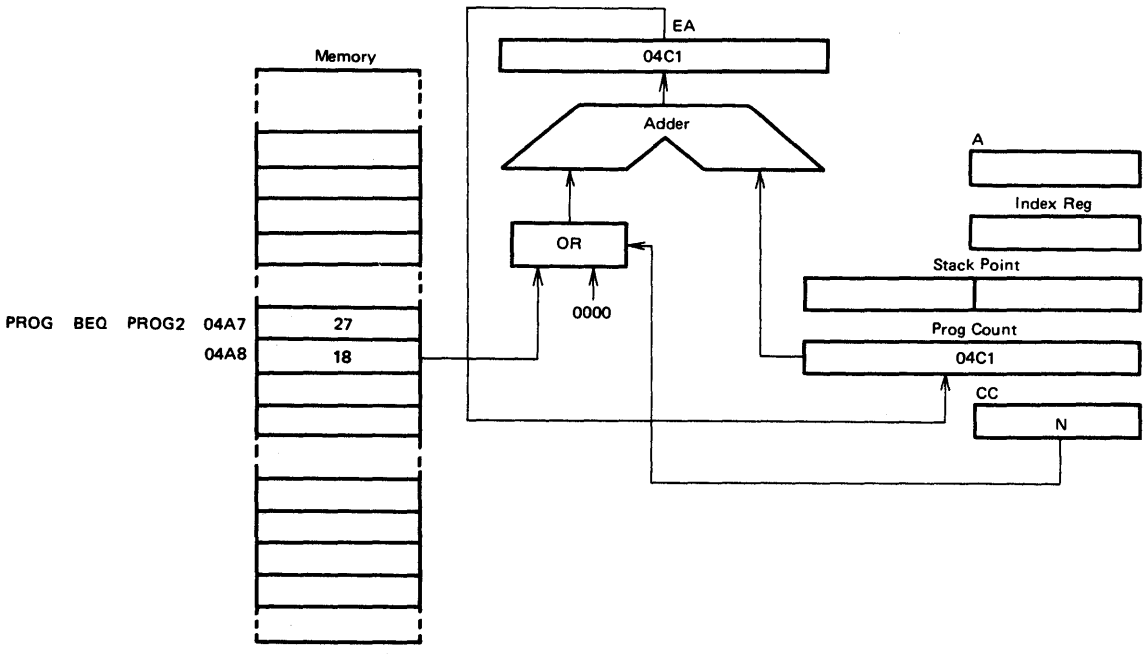


Figure 20 Relative Addressing Example

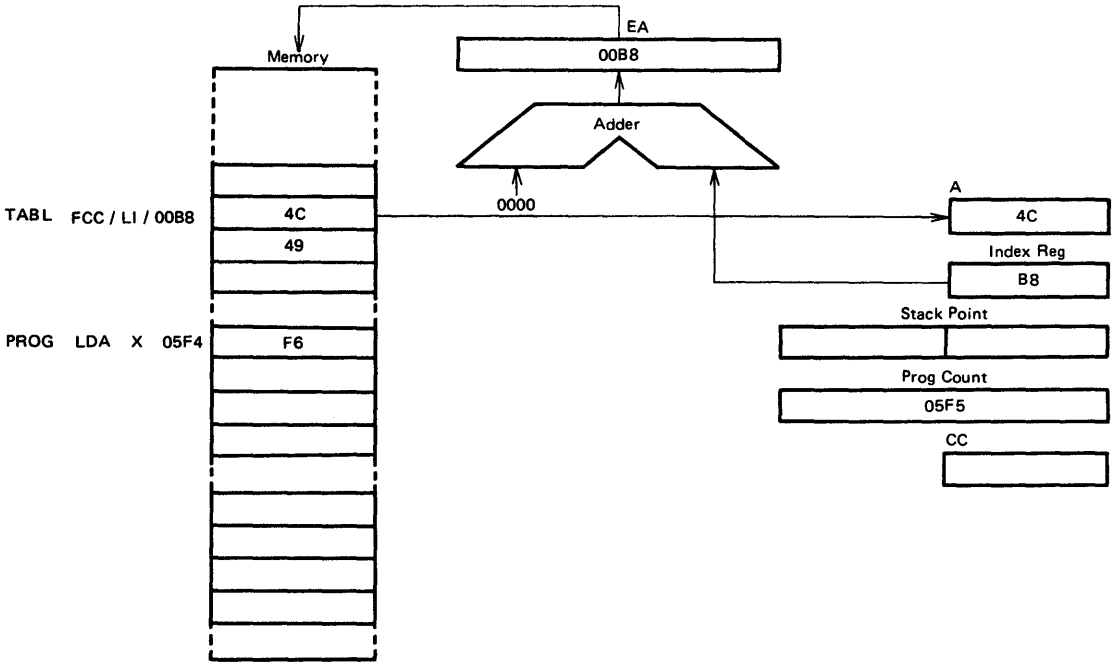


Figure 21 Indexed (No Offset) Addressing Example

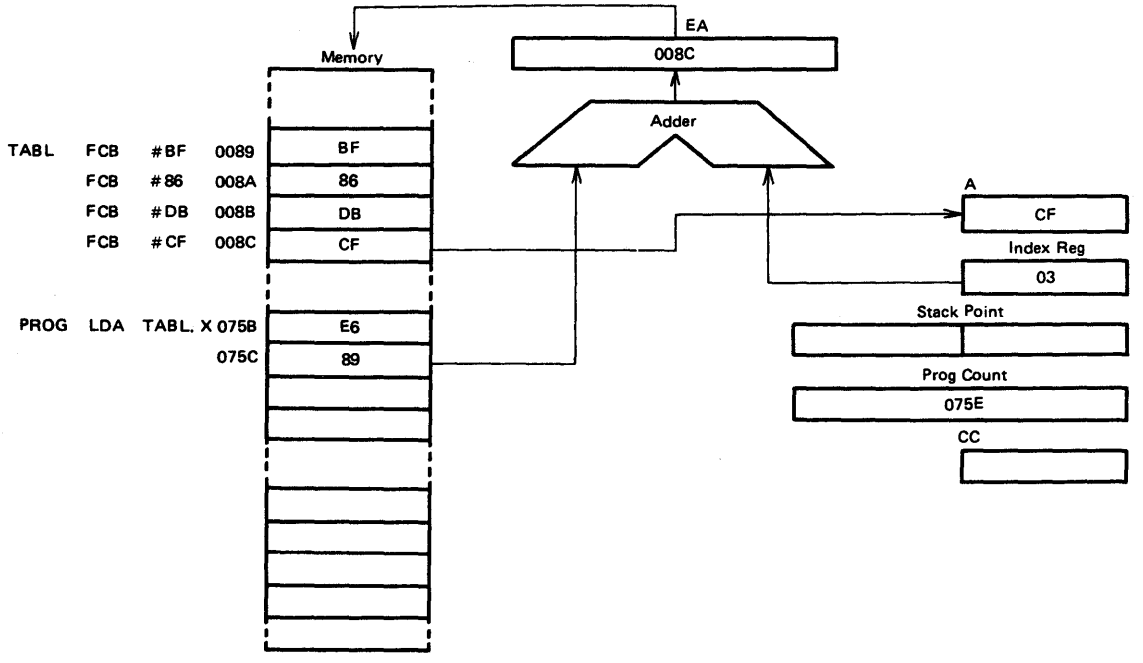


Figure 22 Indexed (8-Bit Offset) Addressing Example

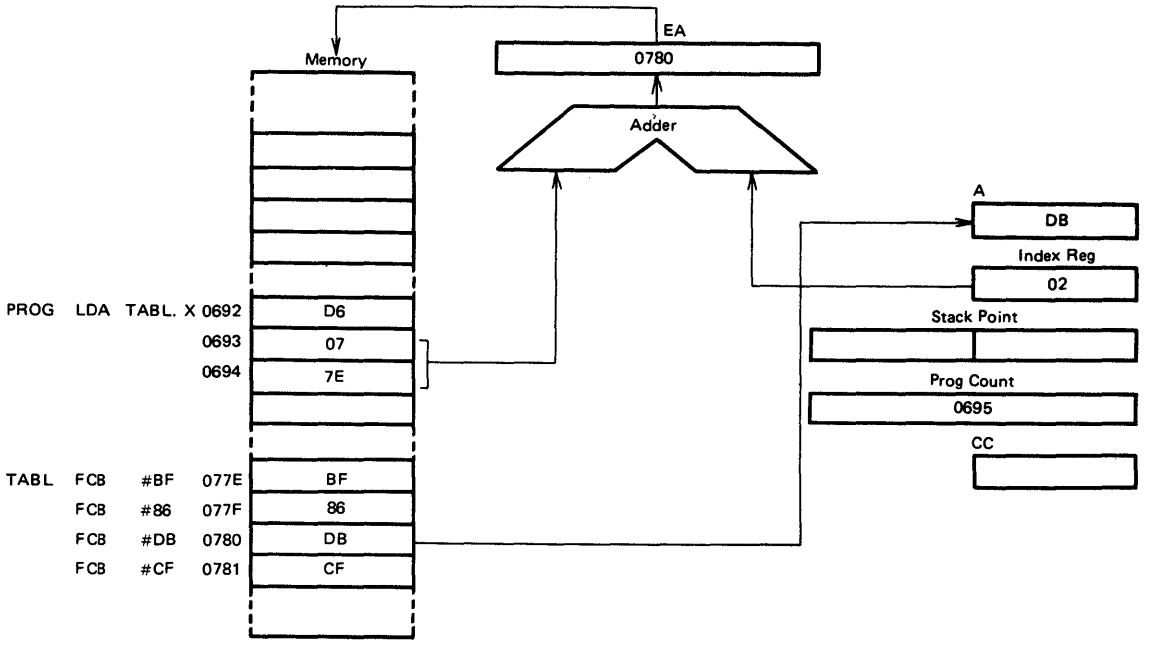


Figure 23 Indexed (16-Bit Offset) Addressing Example

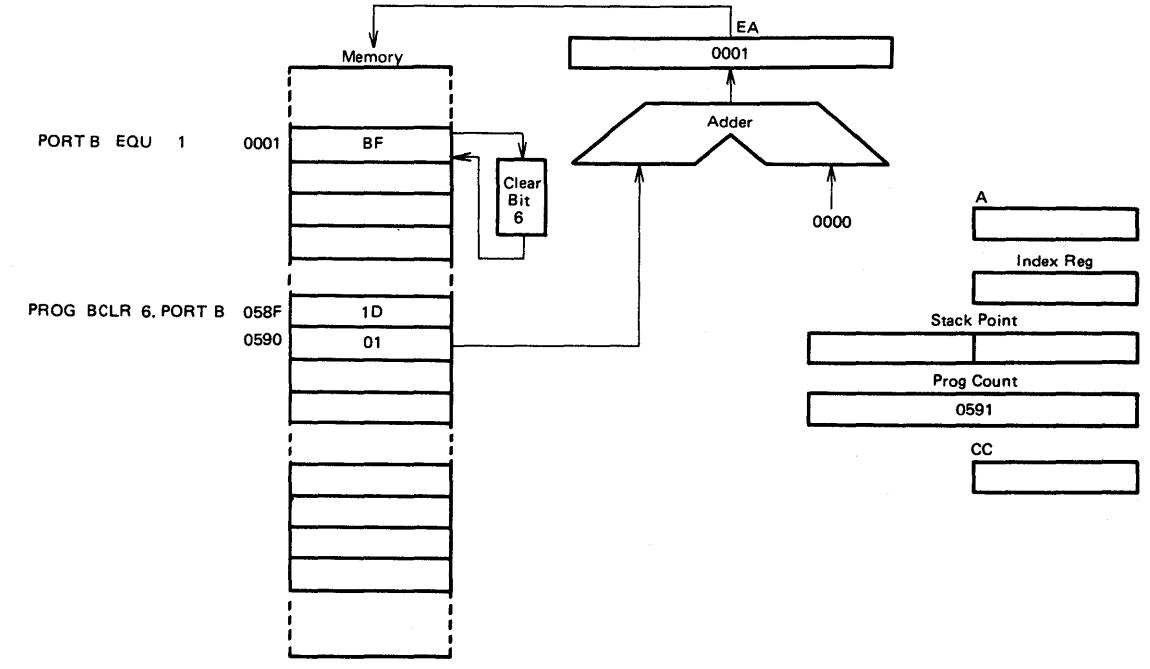


Figure 24 Bit Set/Clear Addressing Example

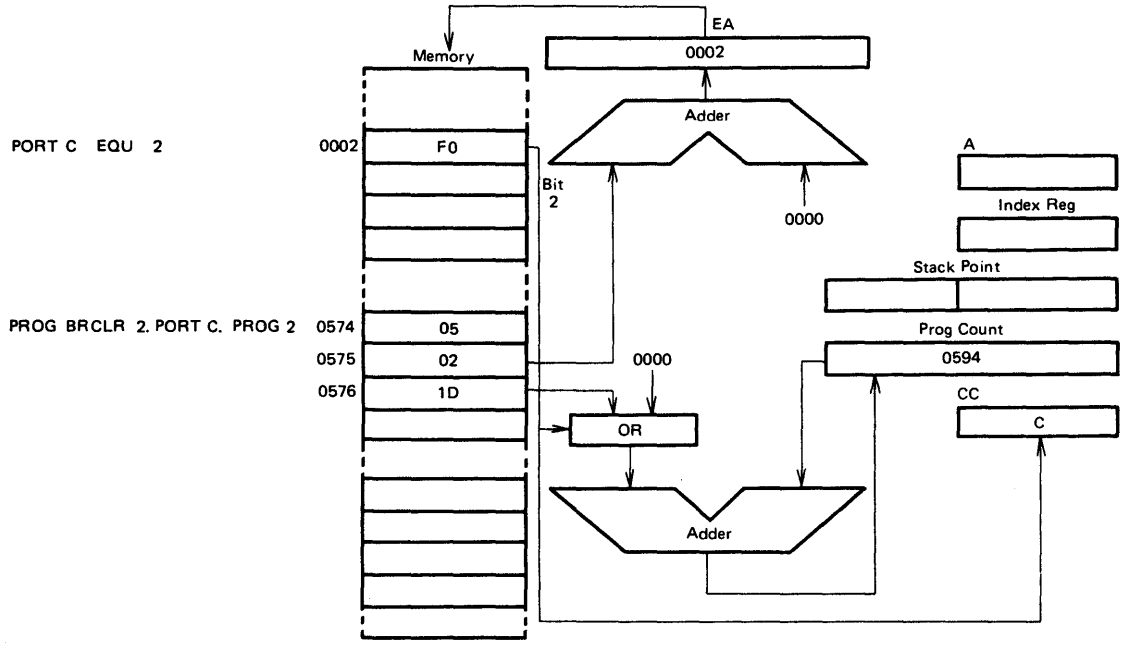


Figure 25 Bit Test and Branch Addressing Example

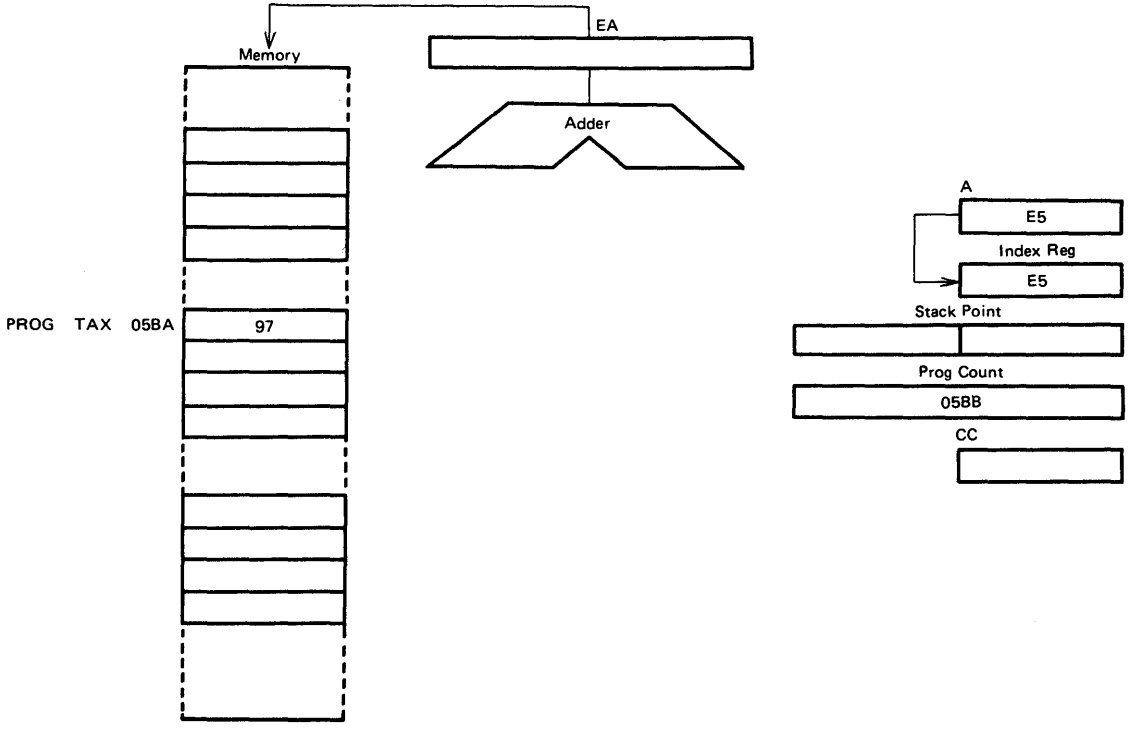


Figure 26 Implied Addressing Example

Table 2 Register/Memory Instructions

Function	Mnemonic	Addressing Modes																	
		Immediate			Direct			Extended			Indexed (No Offset)			Indexed (8-Bit Offset)			Indexed (16-Bit Offset)		
		Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Load A from Memory	LDA	A6	2	2	B6	2	4	C6	3	5	F6	1	4	E6	2	5	D6	3	6
Load X from Memory	LDX	AE	2	2	BE	2	4	CE	3	5	FE	1	4	EE	2	5	DE	3	6
Store A in Memory	STA	–	–	–	B7	2	5	C7	3	6	F7	1	5	E7	2	6	D7	3	7
Store X in Memory	STX	–	–	–	BF	2	5	CF	3	6	FF	1	5	EF	2	6	DF	3	7
Add Memory to A	ADD	AB	2	2	BB	2	4	CB	3	5	FB	1	4	EB	2	5	DB	3	6
Add Memory and Carry to A	ADC	A9	2	2	B9	2	4	C9	3	5	F9	1	4	E9	2	5	D9	3	6
Subtract Memory	SUB	A0	2	2	B0	2	4	C0	3	5	F0	1	4	E0	2	5	D0	3	6
Subtract Memory from A with Borrow	SBC	A2	2	2	B2	2	4	C2	3	5	F2	1	4	E2	2	5	D2	3	6
AND Memory to A	AND	A4	2	2	B4	2	4	C4	3	5	F4	1	4	E4	2	5	D4	3	6
OR Memory with A	ORA	AA	2	2	BA	2	4	CA	3	5	FA	1	4	EA	2	5	DA	3	6
Exclusive OR Memory with A	EOR	A8	2	2	B8	2	4	C8	3	5	F8	1	4	E8	2	5	D8	3	6
Arithmetic Compare A with Memory	CMP	A1	2	2	B1	2	4	C1	3	5	F1	1	4	E1	2	5	D1	3	6
Arithmetic Compare X with Memory	CPX	A3	2	2	B3	2	4	C3	3	5	F3	1	4	E3	2	5	D3	3	6
Bit Test Memory with A (Logical Compare)	BIT	A5	2	2	B5	2	4	C5	3	5	F5	1	4	E5	2	5	D5	3	6
Jump Unconditional	JMP	–	–	–	BC	2	3	CC	3	4	FC	1	3	EC	2	4	DC	3	5
Jump to Subroutine	JSR	–	–	–	BD	2	7	CD	3	8	FD	1	7	ED	2	8	DD	3	9

Table 3 Read/Modify/Write Instructions

Function	Mnemonic	Addressing Modes														
		Implied (A)			Implied (X)			Direct			Indexed (No Offset)			Indexed (8-Bit Offset)		
		Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Increment	INC	4C	1	4	5C	1	4	3C	2	6	7C	1	6	6C	2	7
Decrement	DEC	4A	1	4	5A	1	4	3A	2	6	7A	1	6	6A	2	7
Clear	CLR	4F	1	4	5F	1	4	3F	2	6	7F	1	6	6F	2	7
Complement	COM	43	1	4	53	1	4	33	2	6	73	1	6	63	2	7
Negate (2's Complement)	NEG	40	1	4	50	1	4	30	2	6	70	1	6	60	2	7
Rotate Left Thru Carry	ROL	49	1	4	59	1	4	39	2	6	79	1	6	69	2	7
Rotate Right Thru Carry	ROR	46	1	4	56	1	4	36	2	6	76	1	6	66	2	7
Logical Shift Left	LSL	48	1	4	58	1	4	38	2	6	78	1	6	68	2	7
Logical Shift Right	LSR	44	1	4	54	1	4	34	2	6	74	1	6	64	2	7
Arithmetic Shift Right	ASR	47	1	4	57	1	4	37	2	6	77	1	6	67	2	7
Arithmetic Shift Left	ASL	48	1	4	58	1	4	38	2	6	78	1	6	68	2	7
Test for Negative or Zero	TST	4D	1	4	5D	1	4	3D	2	6	7D	1	6	6D	2	7

Table 4 Branch Instructions

Function	Mnemonic	Relative Addressing Mode		
		Op Code	# Bytes	# Cycles
Branch Always	BRA	20	2	4
Branch Never	BRN	21	2	4
Branch IF Higher	BHI	22	2	4
Branch IF Lower or Same	BLS	23	2	4
Branch IF Carry Clear	BCC	24	2	4
(Branch IF Higher or Same)	(BHS)	24	2	4
Branch IF Carry Set	BCS	25	2	4
(Branch IF Lower)	(BLO)	25	2	4
Branch IF Not Equal	BNE	26	2	4
Branch IF Equal	BEQ	27	2	4
Branch IF Half Carry Clear	BHCC	28	2	4
Branch IF Half Carry Set	BHCS	29	2	4
Branch IF Plus	BPL	2A	2	4
Branch IF Minus	BMI	2B	2	4
Branch IF Interrupt Mask Bit is Clear	BMC	2C	2	4
Branch IF Interrupt Mask Bit is Set	BMS	2D	2	4
Branch IF Interrupt Line is Low	BIL	2E	2	4
Branch IF Interrupt Line is High	BIH	2F	2	4
Branch to Subroutine	BSR	AD	2	8

Table 5 Bit Manipulation Instructions

Function	Mnemonic	Addressing Modes					
		Bit Set/Clear			Bit Test and Branch		
		Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Branch IF Bit n is set	BRSET n (n=0 ..... 7)	—	—	—	2•n	3	10
Branch IF Bit n is clear	BRCLR n (n=0 .....7)	—	—	—	01+2•n	3	10
Set Bit n	BSET n (n=0 ..... 7)	10+2•n	2	7	—	—	—
Clear bit n	BCLR n (n=0 ..... 7)	11+2•n	2	7	—	—	—

Table 6 Control Instructions

Function	Mnemonic	Implied		
		Op Code	# Bytes	# Cycles
Transfer A to X	TAX	97	1	2
Transfer X to A	TXA	9F	1	2
Set Carry Bit	SEC	99	1	2
Clear Carry Bit	CLC	98	1	2
Set Interrupt Mask Bit	SEI	9B	1	2
Clear Interrupt Mask Bit	CLI	9A	1	2
Software Interrupt	SWI	83	1	11
Return from Subroutine	RTS	81	1	6
Return from Interrupt	RTI	80	1	9
Reset Stack Pointer	RSP	9C	1	2
No-Operation	NOP	9D	1	2

Table 7 Instruction Set

Mnemonic	Addressing Modes										Condition Code				
	Implied	Immediate	Direct	Extended	Relative	Indexed (No Offset)	Indexed (8 Bits)	Indexed (16 Bits)	Bit Set/ Clear	Bit Test & Branch	H	I	N	Z	C
ADC		x	x	x		x	x	x			∧	•	∧	∧	∧
ADD		x	x	x		x	x	x			∧	•	∧	∧	∧
AND		x	x	x		x	x	x			•	•	∧	∧	•
ASL	x		x			x	x				•	•	∧	∧	∧
ASR	x		x			x	x				•	•	∧	∧	∧
BCC					x						•	•	•	•	•
BCLR									x		•	•	•	•	•
BCS					x						•	•	•	•	•
BEQ					x						•	•	•	•	•
BHCC					x						•	•	•	•	•
BHCS					x						•	•	•	•	•
BHI					x						•	•	•	•	•
BHS					x						•	•	•	•	•
BIH					x						•	•	•	•	•
BIL					x						•	•	•	•	•
BIT		x	x	x		x	x	x			•	•	∧	∧	•
BLO					x						•	•	•	•	•
BLS					x						•	•	•	•	•
BMC					x						•	•	•	•	•
BMI					x						•	•	•	•	•
BMS					x						•	•	•	•	•
BNE					x						•	•	•	•	•
BPL					x						•	•	•	•	•
BRA					x						•	•	•	•	•
BRN					x						•	•	•	•	•
BRCLR										x	•	•	•	•	∧
BRSET										x	•	•	•	•	∧
BSET									x		•	•	•	•	•
BSR					x						•	•	•	•	•
CLC	x										•	•	•	•	0
CLI	x										•	0	•	•	•
CLR	x		x			x	x				•	•	0	1	•
CMP		x	x	x		x	x	x			•	•	∧	∧	∧
COM	x		x			x	x				•	•	∧	∧	1
CPX		x	x	x		x	x	x			•	•	∧	∧	∧
DEC	x		x			x	x				•	•	∧	∧	•
EOR		x	x	x		x	x	x			•	•	∧	∧	•
INC	x		x			x	x				•	•	∧	∧	•
JMP			x	x		x	x	x			•	•	•	•	•
JSR			x	x		x	x	x			•	•	•	•	•
LDA		x	x	x		x	x	x			•	•	∧	∧	•
LDX		x	x	x		x	x	x			•	•	∧	∧	•

(to be continued)

Condition Code Symbols:
- H Half Carry (From Bit 3)
- I Interrupt Mask
- N Negative (Sign Bit)
- Z Zero
- C Carry Borrow
- ∧ Test and Set if True, Cleared Otherwise
- • Not Affected

Table 7 Instruction Set

Mnemonic	Addressing Modes										Condition Code				
	Implied	Immediate	Direct	Extended	Relative	Indexed (No Offset)	Indexed (8 Bits)	Indexed (16 Bits)	Bit Set/ Clear	Bit Test & Branch	H	I	N	Z	C
LSL	x		x			x	x				•	•	∧	∧	∧
LSR	x		x			x	x				•	•	0	∧	∧
NEG	x		x			x	x				•	•	∧	∧	∧
NOP	x										•	•	•	•	•
ORA		x	x	x		x	x	x			•	•	∧	∧	•
ROL	x		x			x	x				•	•	∧	∧	∧
ROR	x		x			x	x				•	•	∧	∧	∧
RSP	x										•	•	•	•	•
RTI	x										?	?	?	?	?
RTS	x										•	•	•	•	•
SBC		x	x	x		x	x	x			•	•	∧	∧	∧
SEC	x										•	•	•	•	1
SEI	x										•	1	•	•	•
STA			x	x		x	x	x			•	•	∧	∧	•
STX			x	x		x	x	x			•	•	∧	∧	•
SUB		x	x	x		x	x	x			•	•	∧	∧	∧
SWI	x										•	1	•	•	•
TAX	x										•	•	•	•	•
TST	x		x			x	x				•	•	∧	∧	•
TXA	x										•	•	•	•	•

Condition Code Symbols:
- H Half Carry (From Bit 3)
- I Interrupt Mask
- N Negative (Sign Bit)
- Z Zero
- C Carry/Borrow
- ∧ Test and Set if True, Cleared Otherwise
- • Not Affected
- ? Load CC Register From Stack

Table 8 Opcode Map

	Bit Manipulation		Branch	Read/Modify/Write					Control		Register/Memory						
	Test & Branch	Set/ Clear	Rel	DIR	A	X	,X1	,X0	IMP	,IMP	IMM	DIR	EXT	,X2	,X1	,X0	
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	← HIGH
0	BRSET0	BSET0	BRA	NEG					RTI*	–	SUB						0
1	BRCLR0	BCLR0	BRN	–					RTS*	–	CMP						1
2	BRSET1	BSET1	BHI	–					–	–	SBC						2
3	BRCLR1	BCLR1	BLS	COM					SWI*	–	CPX						3 L
4	BRSET2	BSET2	BCC	LSR					–	–	AND						4 O
5	BRCLR2	BCLR2	BCS	–					–	–	BIT						5 W
6	BRSET3	BSET3	BNE	ROR					–	–	LDA						6
7	BRCLR3	BCLR3	BEQ	ASR					–	TAX	–	STA(+1)					7
8	BRSET4	BSET4	BHCC	LSL/ASL					–	CLC	EOR						8
9	BRCLR4	BCLR4	BHCS	ROL					–	SEC	ADC						9
A	BRSET5	BSET5	BPL	DEC					–	CLI	ORA						A
B	BRCLR5	BCLR5	BMI	–					–	SEI	ADD						B
C	BRSET6	BSET6	BMC	INC					–	RSP	–	JMP(-1)					C
D	BRCLR6	BCLR6	BMS	TST					–	NOP	BSR*	JSR(-3)					D
E	BRSET7	BSET7	BIL	–					–	–	LDX						E
F	BRCLR7	BCLR7	BIH	CLR					–	TXA	–	STX(+1)					F
	3/10	2/7	2/4	2/6	1/4	1/4	2/7	1/6	1/*	1/2	2/2	2/4	3/5	3/6	2/5	1/4	

(NOTE) 1. Undefined opcodes are marked with "–".
2. The number at the bottom of each column denote the number of bytes and the number of cycles required (Bytes/Cycles).
Mnemonics followed by a "*" require a different number of cycles as follows:

RTI	9
RTS	6
SWI	11
BSR	8

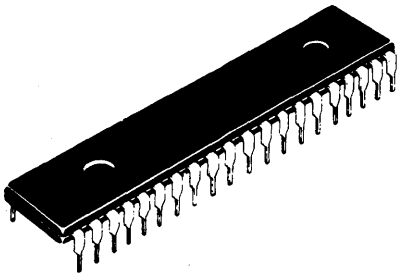
3. ( ) indicate that the number in parenthesis must be added to the cycle count for that instruction.

HD6805V1
MCU (Microcomputer Unit)

The HD6805V1 is the 8-bit Microcomputer Unit (MCU) which contains a CPU, on-chip clock, ROM, RAM, I/O and timer. It is designed for the user who needs an economical microcomputer with the proven capabilities of the HD6800-based instruction set.

The following are some of the hardware and software highlights of the MCU.

HD6805V1P



(DP-40)

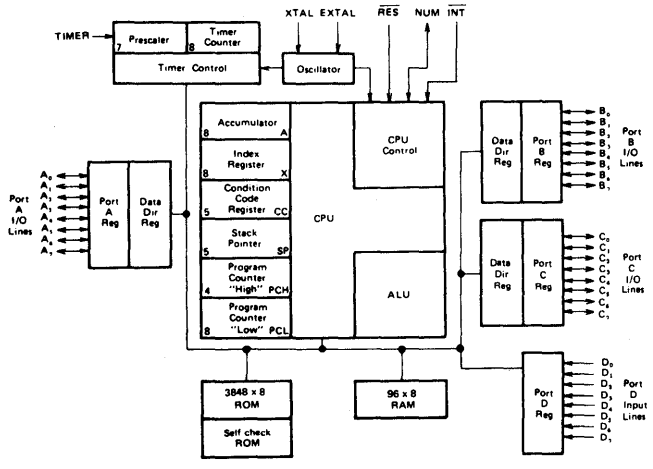
■ HARDWARE FEATURES

- 8-Bit Architecture
- 96 Bytes of RAM
- Memory Mapped I/O
- 3848 Bytes of User ROM
- Internal 8-Bit Timer with 7-Bit Prescaler
- Vectored Interrupts – External and Timer
- 24 I/O Ports + 8 Input Port
  (8 Lines LED Compatible; 7 Voltage Comparator Inputs)
- On-Chip Clock Circuit
- Self-Check Mode
- Master Reset
- Low Voltage Inhibit
- Complete Development System Support by Evaluation Kit
- 5 Vdc Single Supply

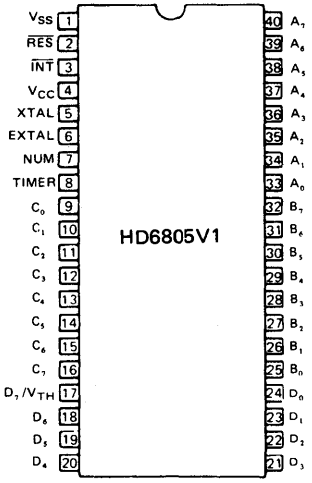
■ SOFTWARE FEATURES

- Similar to HD6800
- Byte Efficient Instruction Set
- Easy to Program
- True Bit Manipulation
- Bit Test and Branch Instructions
- Versatile Interrupt Handing
- Powerful Indexed Addressing for Tables
- Full Set of Conditional Branches
- Memory Usable as Registers/Flags
- Single Instruction Memory Examine/Change
- 10 Powerful Addressing Modes
- All Addressing Modes Apply to ROM, RAM and I/O
- Compatible Instruction Set with MC6805P2

■ BLOCK DIAGRAM



■ PIN ARRANGEMENT



(Top View)

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC} *	-0.3 ~ +7.0	V
Input Voltage (EXCEPT TIMER)	V_{in} *	-0.3 ~ +7.0	V
Input Voltage (TIMER)		-0.3 ~ +12.0	V
Operating Temperature	T_{opr}	0 ~ +70	°C
Storage Temperature	T_{stg}	- 55 ~ +150	°C

* With respect to V_{SS} (SYSTEM GND)

(NOTE) Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ ELECTRICAL CHARACTERISTICS

• DC CHARACTERISTICS (V_{CC}=5.25V ± 0.5V, V_{SS}=GND, Ta=0~+70°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Input "High" Voltage	$\overline{RES}$	V_{IH}		4.0	–	V_{CC}	V
	$\overline{INT}$			3.0	–	V_{CC}	V
	All Other			2.0	–	V_{CC}	V
Input "High" Voltage Timer	Timer Mode			2.0	–	V_{CC}	V
	Self-Check Mode			9.0	–	11.0	V
Input "Low" Voltage	$\overline{RES}$	V_{IL}		-0.3	–	0.8	V
	$\overline{INT}$			-0.3	–	0.8	V
	XTAL(Crystal Mode)			-0.3	–	0.6	V
	All Other			-0.3	–	0.8	V
Power Dissipation		P_D		–	400	700	mW
Low Voltage Recover		LVR		–	–	4.75	V
Low Voltage Inhibit		LVI		–	4.0	–	V
Input Leak Current	TIMER	I_{IL}	V_{in}=0.4V~V_{CC}	-20	–	20	μA
	$\overline{INT}$			-50	–	50	μA
	XTAL(Crystal Mode)			-1200	–	0	μA

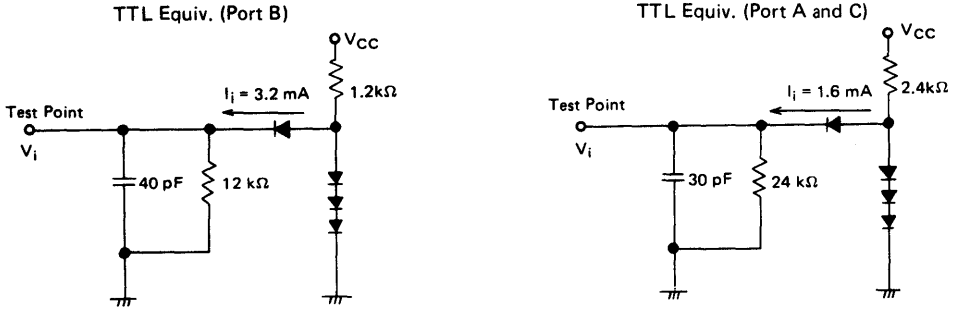
• AC CHARACTERISTICS (V_{CC}=5.25V ± 0.5V, V_{SS}=GND, Ta=0 ~ +70°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Clock Frequency		f_{cl}		0.4	–	4.0	MHz
Cycle Time		t_{cyc}		1.0	–	10	μs
Oscillation Frequency (External Resister Mode)		f_{EXT}	R_{CP}=15.0kΩ±1%	2.7	–	4.0	MHz
$\overline{INT}$ Pulse Width		t_{IWL}		t_{cyc}+250	–	–	ns
$\overline{RES}$ Pulse Width		t_{RWL}		t_{cyc}+250	–	–	ns
TIMER Pulse Width		t_{TWL}		t_{cyc}+250	–	–	ns
Oscillation Start-up Time (Crystal Mode)		t_{OSC}	C_L=22pF±20%, R_S=60Ω max.	–	–	100	ms
Delay Time Reset		t_{RHL}	External Cap. = 2.2 μF	100	–	–	ms
Input Capacitance	EXTAL	C_{in}	V_{in}=0V	–	25	35	pF
	All Other			–	6	10	pF

● PORT ELECTRICAL CHARACTERISTICS (V_{CC} = 5.25V ± 0.5V, V_{SS} = GND, Ta = 0 ~ +70°C unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Output "High" Voltage	Port A	V_{OH}	I_{OH} = −10 μA	3.5	–	–	V
			I_{OH} = −100 μA	2.4	–	–	V
	Port B		I_{OH} = −200 μA	2.4	–	–	V
			I_{OH} = −1 mA	1.5	–	–	V
	Port C		I_{OH} = −100 μA	2.4	–	–	V
Output "Low" Voltage	Port A and C	V_{OL}	I_{OL} = 1.6 mA	–	–	0.4	V
	Port B		I_{OL} = 3.2 mA	–	–	0.4	V
			I_{OL} = 10 mA	–	–	1.0	V
Input "High" Voltage	Port A, B, C, and D*	V_{IH}		2.0	–	V_{CC}	V
Input "Low" Voltage		V_{IL}		−0.3	–	0.8	V
Input Leak Current	Port A	I_{IL}	V_{in} = 0.8V	−500	–	–	μA
			V_{in} = 2V	−300	–	–	μA
	Port B, C, and D		V_{in} = 0.4V ~ V_{CC}	− 20	–	20	μA
Input "High" Voltage	Port D** (D_0 ~ D_6)	V_{IH}		–	V_{TH}+0.2	—	V
Input "Low" Voltage	Port D** (D_0 ~ D_6)	V_{IL}		–	V_{TH}-0.2	—	V
Threshold Voltage	Port D**(D_7)	V_{TH}		0	—	0.8×V_{CC}	V

* Port D as digital input
** Port D as analog input



(NOTE) 1. Load capacitance includes the floating capacitance of the probe and the jig etc.
2. All diodes are 1S2074Ⓗ or equivalent.

Figure 1 Bus Timing Test Loads

■ SIGNAL DESCRIPTION

The input and output signals for the MCU shown in PIN ARRANGEMENT are described in the following paragraphs.

● V_{CC} and V_{SS}

Power is supplied to the MCU using these two pins. V_{CC} is +5.25V ±0.5V. V_{SS} is the ground connection.

● $\overline{INT}$

This pin provides the capability for applying an external interrupt to the MCU Refer to INTERRUPTS for additional information.

● XTAL and EXTAL

These pins provide control input for the on-chip clock circuit. A crystal (AT cut, 4 MHz maximum) or a resistor can be connected to these pins to provide the internal oscillator with varying degrees of stability. Refer to INTERNAL OSCILLATOR OPTIONS for recommendations about these inputs.

● TIMER

This pin allows an external input to be used to decrement the internal timer circuitry. Refer to TIMER for additional information about the timer circuitry.

● $\overline{RES}$

This pin allows resetting of the MCU at times other than the automatic resetting capability already in the MCU. Refer to RESETS for additional information.

● NUM

This pin is not for user application and should be connected to ground.

- **Input/Output Lines ($A_0 \sim A_7$, $B_0 \sim B_7$, $C_0 \sim C_7$)**

These 24 lines are arranged into three 8-bit ports (A, B and C). All lines are programmable as either inputs or outputs under software control of the data direction registers. Refer to INPUTS/OUTPUTS for additional information.

- **Input Lines ($D_0 \sim D_7$)**

These are 8-bit input lines, which has two functions. Firstly, these become TTL compatible inputs, by reading $003 address. The other function of them is 7 Voltage comparators, by reading $007 address. Please refer to INPUT PORT for more detail.

- **MEMORY**

The MCU memory is configured as shown in Figure 2. During the processing of an interrupt, the contents of the MCU registers are pushed onto the stack in the order shown in Figure 3. Since the stack pointer decrements during pushes, the low order byte (PCL) of the program counter is stacked first; then the high order four bits (PCH) are stacked. This ensures that the program counter is loaded correctly as the stack pointer increments when it pulls data from the stack. A subroutine call will cause only the program counter (PCH, PCL) contents to be pushed onto the stack.

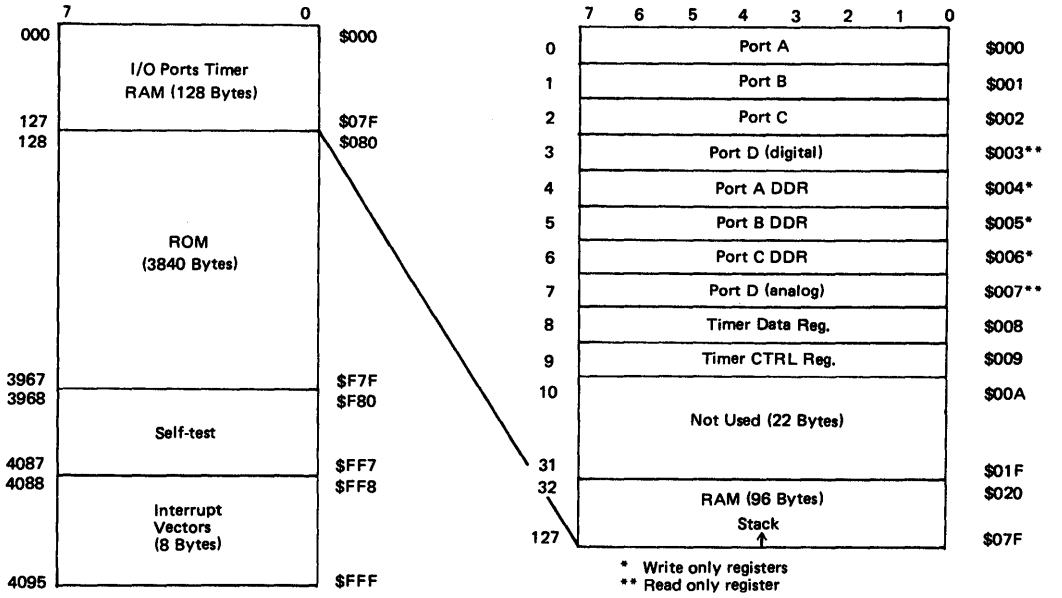


Figure 2 MCU Memory Configuration

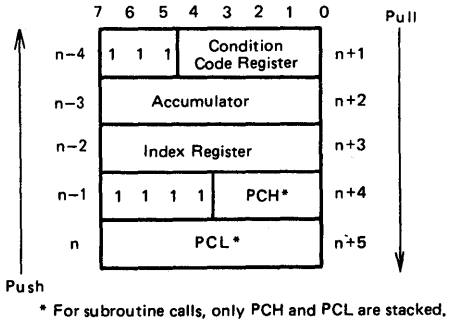


Figure 3 Interrupt Stacking Order

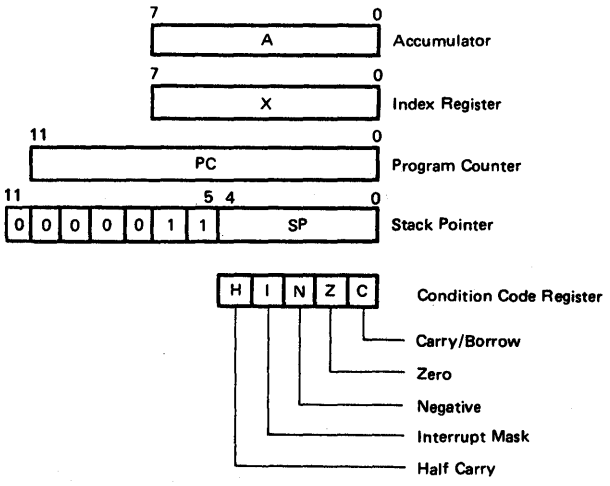


Figure 4 Programming Model

■ REGISTERS

The MCU has five registers available to the programmer. They are shown in Figure 4 and are explained in the following paragraphs.

- **Accumulator (A)**

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.

- **Index Register (X)**

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit address that may be added to an offset value to create an effective address. The index register can also be used for limited calculations and data manipulations when using read/modify/write instructions. When not required by a code sequence being executed, the index register can be used as a temporary storage area.

- **Program Counter (PC)**

The program counter is a 12-bit register that contains the address of the next instruction to be executed.

- **Stack Pointer (SP)**

The stack pointer is a 12-bit register that contains the address of the next free location on the stack. Initially, the stack pointer is set to location $07F and is decremented as data is being pushed onto the stack and incremented as data is being pulled from the stack. The six most significant bits of the stack pointer are permanently set to 0000011. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set to location $07F. Subroutines and interrupts may be nested down to location $061 which allows the programmer to use up to 15 levels of subroutine calls.

- **Condition Code Register (CC)**

The condition code register is a 5-bit register in which each bit is used to indicate or flag the results of the instruction just executed. These bits can be individually tested by a program and specific action taken as a result of their state. Each individual condition code register bit is explained in the following paragraphs.

- **Half Carry (H)**

Used during arithmetic operations (ADD and ADC) to indicate that a carry occurred between bits 3 and 4.

- **Interrupt (I)**

This bit is set to mask the timer and external interrupt ($\overline{INT}$). If an interrupt occurs while this bit is set it is latched and will be processed as soon as the interrupt bit is reset.

- **Negative (N)**

Used to indicate that the result of the last arithmetic, logical or data manipulation was negative (bit 7 in result equal to a logical one).

- **Zero (Z)**

Used to indicate that the result of the last arithmetic, logical or data manipulation was zero.

- **Carry/Borrow (C)**

Used to indicate that a carry or borrow out of the arithmetic logic unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions, shifts, and rotates.

■ TIMER

The MCU timer circuitry is shown in Figure 5. The 8-bit counter is loaded under program control and counts down toward zero as soon as the clock input is applied. When the timer reaches zero the timer interrupt request bit (bit 7) in the timer control register is set. The MCU responds to this interrupt by saving the present MCU state in the stack, fetching the timer interrupt vector from locations $FF8 and $FF9 and executing the interrupt routine. The timer interrupt can be masked by setting the timer interrupt mask bit (bit 6) in the time control register. The interrupt bit (I bit) in the condition code register will also prevent a timer interrupt from being processed.

The clock input to the timer can be from an external source applied to the TIMER input pin or it can be the internal ϕ_2 signal. Note that when the ϕ_2 signal is used as the source it can be gated by an input applied to the TIMER input pin allowing the user to easily perform pulse-width measurements. The source of the clock input is one of the options that has to be specified before manufacture of the MCU. A prescaler option can be applied to the clock input that extends the timing interval up to a maximum of 128 counts before being applied to

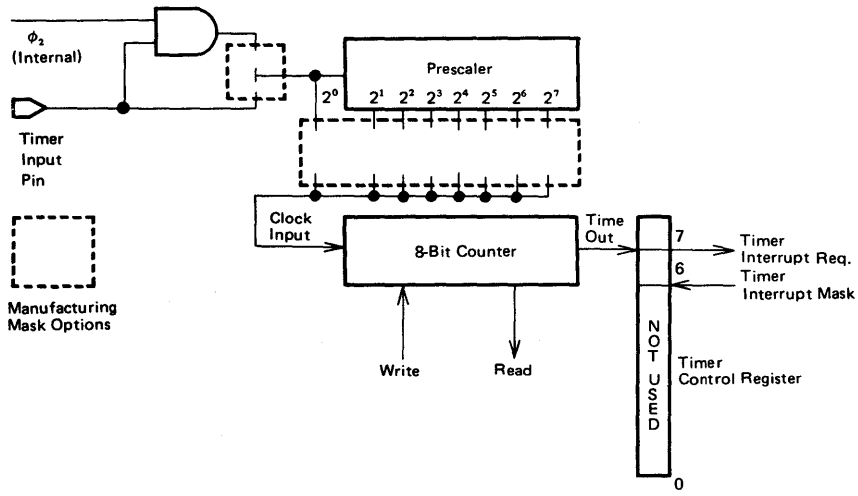


Figure 5 Timer Block Diagram

the counter. This prescaling option must also be specified before manufacturing begins. The timer continues to count past zero and its present count can be monitored at any time by monitoring the timer data register. This allows a program to determine the length of time since a timer interrupt has occured and not disturb the counting process.

The Timer Data Register is 8-bit Read/Write Register with address $008 on Memory-Map. This Timer Data Register and the prescaler are initialize with all logical ones at Reset time.

The Timer Interrupt Request bit (bit 7 of Timer Control Register) is set to one by hardware when timer count reaches zero, and is cleared by program or by hardware reset. The bit 6 of Timer Control Register is writable by program. Both of those bits can be read by MPU.

■ SELF CHECK

The self check capability of the MCU provides an internal check to determine if the part is functional. Connect the MCU as shown in Figure 6 and monitor the output of port C bit 3 for an oscillation of approximately three hertz.

■ RESETS

The MCU can be reset three ways: by initial powerup, by the external reset input ($\overline{RES}$) and by an internal low voltage detect circuit, (mask option) see Figure 7. All the I/O port are initialized to Input mode (DDR's are cleared) during RESET.

Upon power up, a minimum of 100 milliseconds is needed before allowing the reset input to go "High".

This time allows the internal crystal oscillator to stabilize. Connecting a capacitor to the $\overline{RES}$ input as shown in Figure 8 will provide sufficient delay.

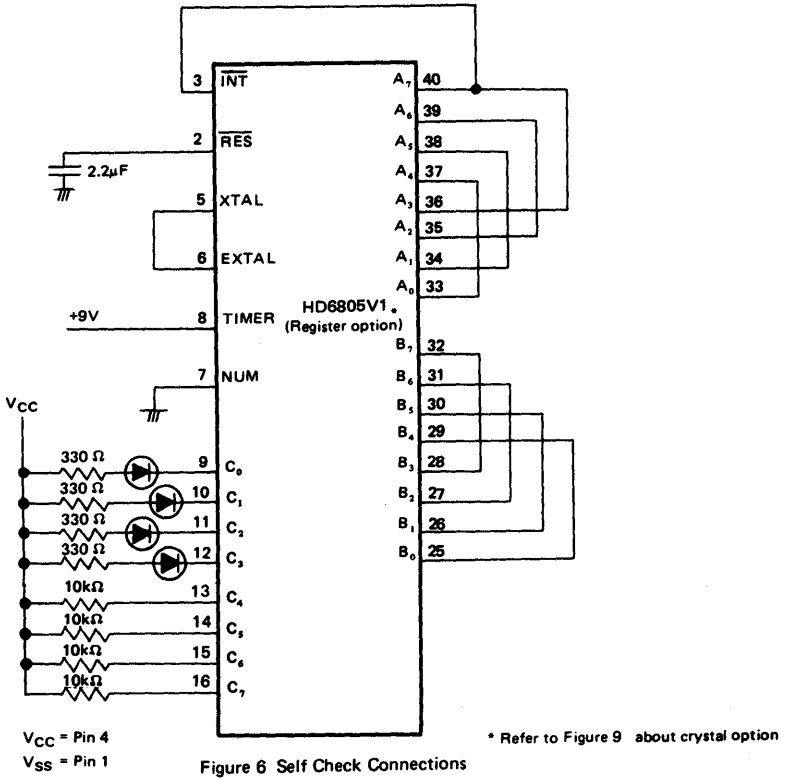


Figure 6 Self Check Connections

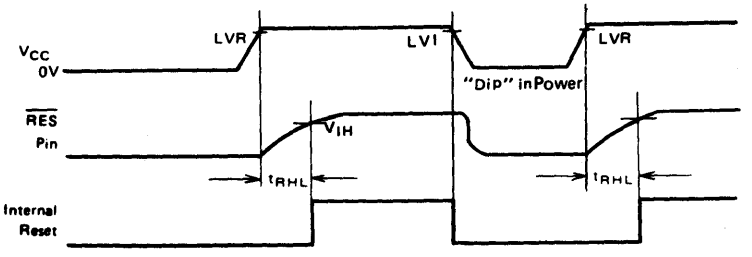


Figure 7 Power Up and $\overline{RES}$ Timing

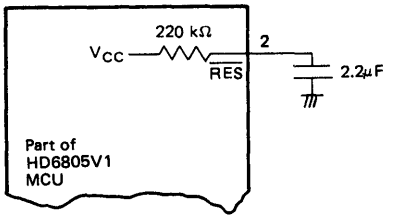


Figure 8 Power Up Reset Delay Circuit

■ INTERNAL OSCILLATOR OPTIONS

The internal oscillator circuit has been designed to require a minimum of external components. The use of a crystal (AT cut, 4 MHz max) or a resistor is sufficient to drive the internal oscillator with varying degrees of stability. A manufacturing mask option is available to provide better matching between the external components and the internal oscillator. The different connection methods are shown in Figure 9. Crystal specifications are given in Figure 10. A resistor selection graph is given in Figure 11.

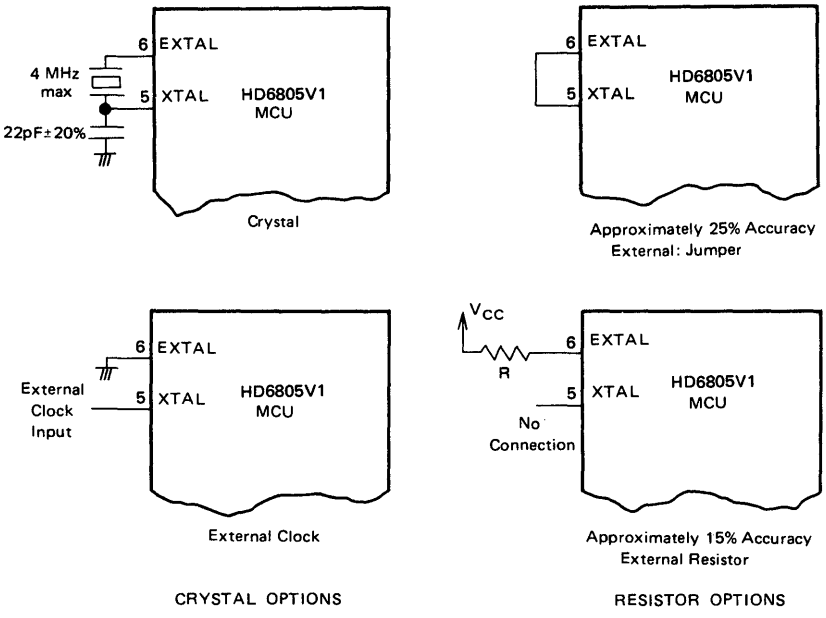
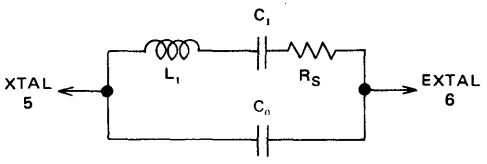


Figure 9 Internal Oscillator Options



AT – Cut Parallel Resonance Crystal
C_0 = 7 pF max.
f = 4 MHz
R_S = 60 Ω max.

Figure 10 Crystal Parameters

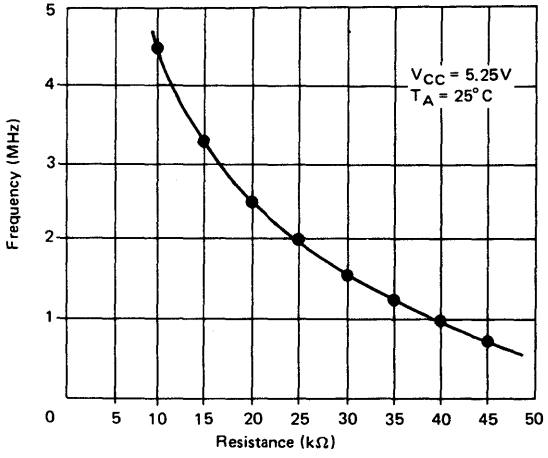


Figure 11 Typical Resistor Selection Graph

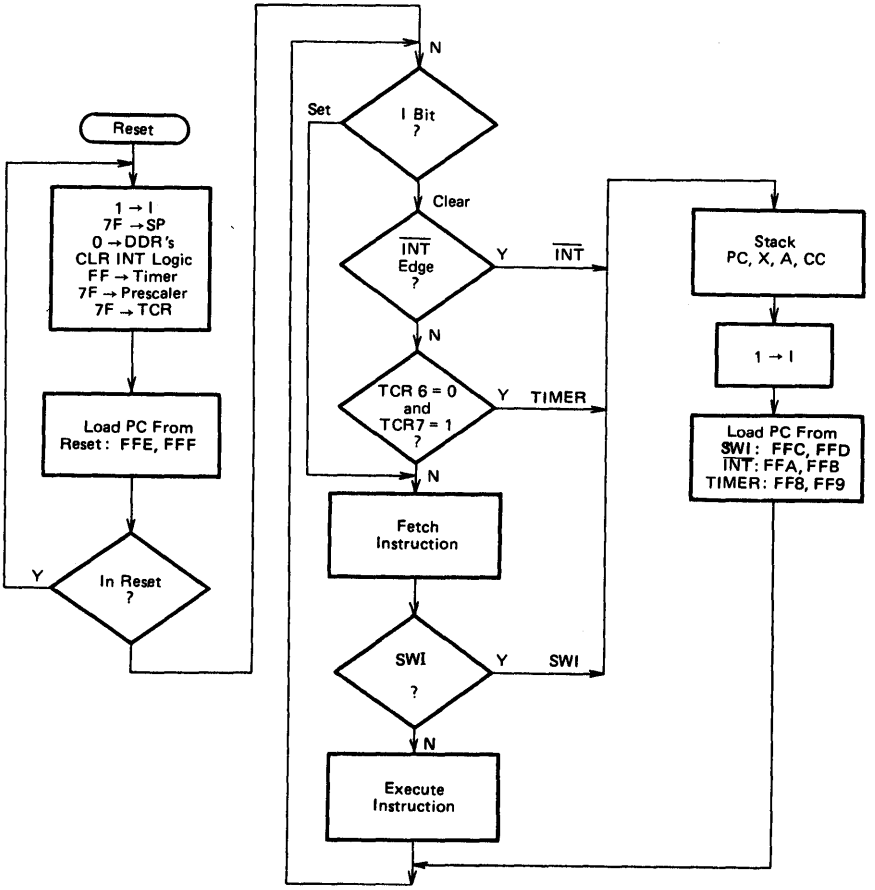
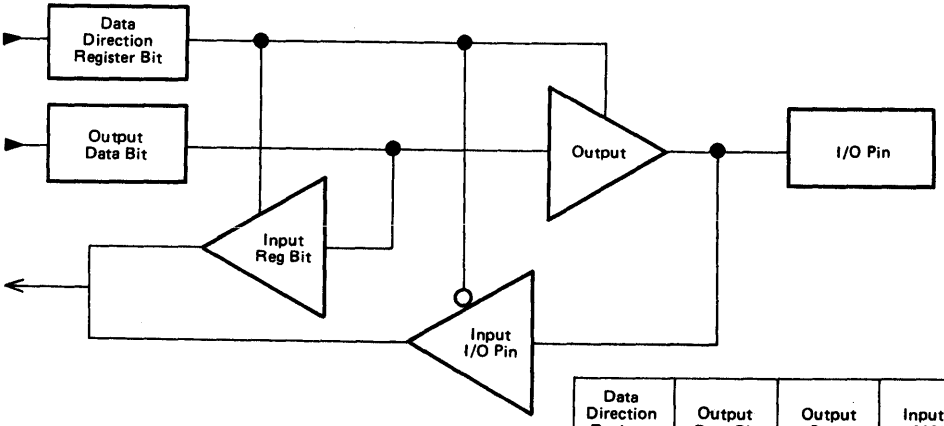


Figure 12 Interrupt Processing Flowchart



Data Direction Register Bit	Output Data Bit	Output State	Input to MCU
1	0	0	0
1	1	1	1
0	x	3-State	Pin

Figure 13 Typical Port I/O Circuitry

■ INTERRUPTS

The MCU can be interrupted three different ways: through the external interrupt ($\overline{INT}$) input pin, the internal timer interrupt request, and a software interrupt instruction (SWI). When any interrupt occurs, processing is suspended, the present MCU state is pushed onto the stack, the interrupt bit (I) in the condition code register is set, the address of the interrupt routine is obtained from the appropriate interrupt vector address, and the interrupt routine is executed. The interrupt service routines normally end with a return from interrupt (RTI) instruction which allows the MCU to resume processing of the program prior to the interrupt. Table 1 provides a listing of the interrupts, their priority, and the vector address that contain the starting address of the appropriate interrupt routine.

A flowchart of the interrupt processing sequence is given in Fig. 12.

Table 1 Interrupt Priorities

Interrupt	Priority	Vector Address
$\overline{RES}$	1	$FFE and $FFF
SWI	2	$FFC and $FFD
$\overline{INT}$	3	$FFA and $FFB
TIMER	4	$FF8 and $FF9

■ INPUT/OUTPUT

There are 24 input/output pins. All pins are programmable as either inputs or outputs under software control of the data direction registers. When programmed as outputs, all I/O pins read latched output data regardless of the logic level at the output pin due to output loading (see Fig. 13). When port B is programmed for outputs, it is capable of sinking 10 millamperes on each pin (V_{OL} = 1V max). All input/output lines are TTL compatible as both inputs and outputs. Port A lines are CMOS compatible as outputs while port B and C lines are CMOS compatible as inputs. Figure 14 provides some examples of port connections.

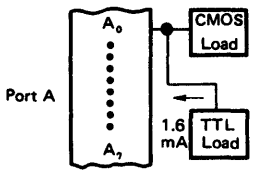
■ INPUT

Port D is 8-bit input port, which has two functions. One of them is usual digital signal input port and the other is voltage compare type input port. In the former case, the input data can be read by MPU at $003 address. In the latter case, D_7 (pin 17) is the input pin of V_{TH} (reference level), and the other seven input pins ($D_0 \sim D_6$) are analog level inputs, which are compared with V_{TH} (see Figure 15(a), (b)).

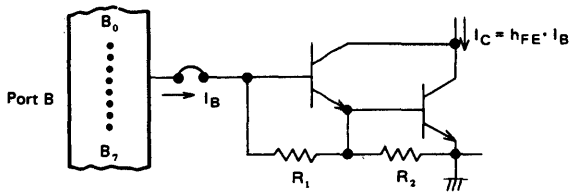
"1" or "0" signals appear at internal data bus, if the input levels are higher or lower respectively when $007 address is read. This function is effective in such case that unusual logic level inputs are used. A capacitive touch panel interface and a diode isolated keyboard interface are the examples. Figure 15(c) shows the application of Port D to A/D converter, and Figure 15(d) shows 3 levels inputs.

■ BIT MANIPULATION

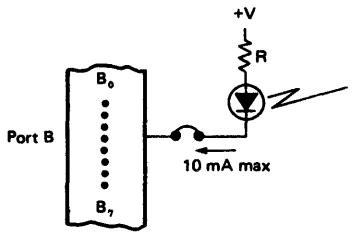
The MCU has the ability to set or clear any single random access memory or input/output bit (except the data direction registers) with a single instruction (BSET, BCLR). Any bit in the page zero read only memory can be tested, using the BRSET and BRCLR instructions, and the program branches as a result of its state. This capability to work with any bit in RAM, ROM or I/O allows the user to have individual flags in RAM or to handle single I/O bits as control lines. The example in Figure 16 illustrates the usefulness of the bit manipulation and test



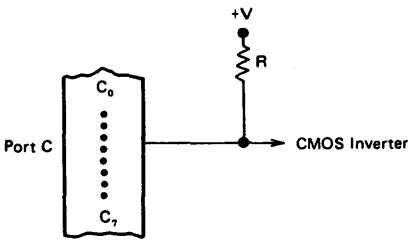
Port A Programmed as output(s) driving CMOS and TTL Load directly.
(a)



Port B Programmed as output(s) driving Darlington base directly.
(b)



Port B Programmed as output(s) driving LED(s) directly.
(c)

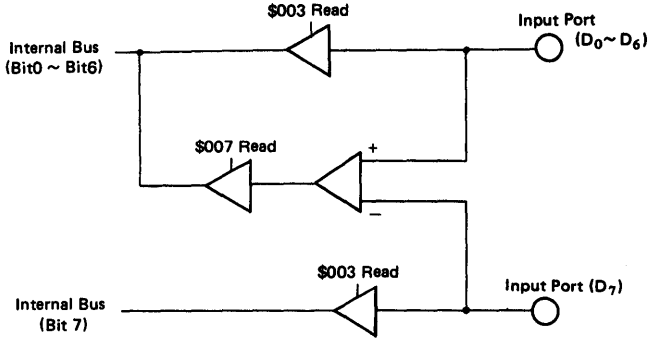


Port C Programmed as output(s) driving CMOS using external pull-up resistors.
(d)

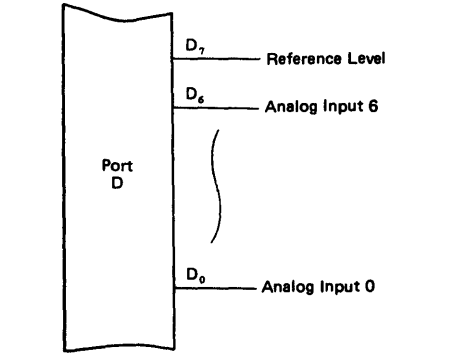
Figure 14 Typical Port Connections

instructions. Assume that bit 0 of port A is connected to a zero crossing detector circuit and that bit 1 of port A is connected to the trigger of a TRIAC which power the controlled hardware.

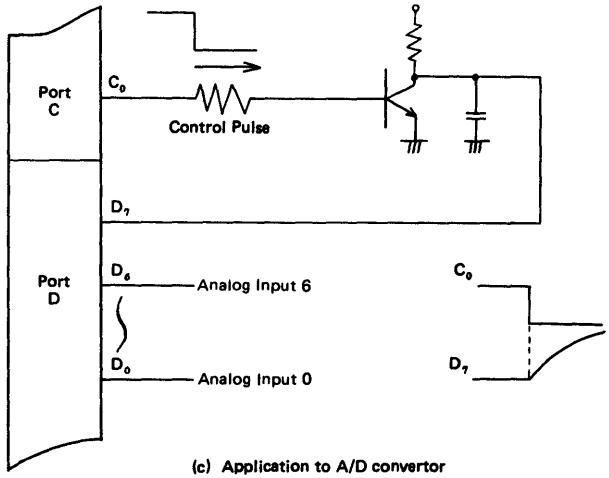
This program, which uses only seven ROM locations, provides turn-on of the TRIAC within 14 microseconds of the zero crossing. The timer could also be incorporated to provide turn-on at some later time which would permit pulse-width modulation of the controlled power.



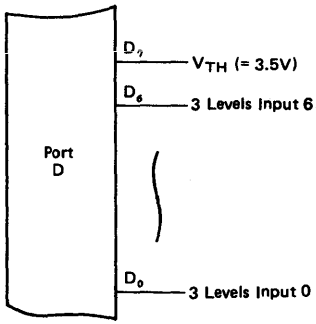
(a) The logic configuration of Port D



(b) Seven analog inputs and a reference level input of Port D



(c) Application to A/D convertor



(d) Application to 3 levels input

Input Voltage	($003)	($007)
0V ~ 0.8V	0	0
2.0V ~ 3.3V	1	0
3.7V ~ V_{CC}	1	1

Figure 15 Configuration and Application of Port D

```
        .
        .
        .
        .
        .
SELF 1  BRCLR 0, PORT A, SELF 1
        BSET 1, PORT A
        BCLR 1, PORT A
        .
        .
        .
        .
        .
```

Figure 16 Bit Manipulation Example

■ ADDRESSING MODES

The MCU has ten addressing modes available for use by the programmer. They are explained and illustrated briefly in the following paragraphs.

- **Immediate**

Refer to Figure 17. The immediate addressing mode accesses constants which do not change during program execution. Such instructions are two bytes long. The effective address (EA) is the PC and the operand is fetched from the byte following the opcode.

- **Direct**

Refer to Figure 18. In direct addressing, the address of the operand is contained in the second byte of the instruction. Direct addressing allows the user to directly address the lowest 256 bytes in memory. All RAM space, I/O registers and 128 bytes of ROM are located in page zero to take advantage of this efficient memory addressing mode.

- **Extended**

Refer to Figure 19. Extended addressing is used to reference any location in memory space. The EA is the contents of the two bytes following the opcode. Extended addressing instructions are three bytes long.

- **Relative**

Refer to Figure 20. The relative addressing mode applies only to the branch instructions. In this mode the contents of the byte following the opcode is added to the program counter when the branch is taken. EA=(PC)+2+Rel. Rel is the contents of the location following the instruction opcode with bit 7 being the sign bit. If the branch is not taken Rel=0, when a branch takes place, the program goes to somewhere within the range of +129 bytes to -127 of the present instruction. These instructions are two bytes long.

- **Indexed (No Offset)**

Refer to Figure 21. This mode of addressing accesses the lowest 256 bytes of memory. These instructions are one byte long and their EA is the contents of the index register.

- **Indexed (8-bit Offset)**

Refer to Figure 22. The EA is calculated by adding the contents of the byte following the opcode to the contents of the index register. In this mode, 511 low memory locations are accessable. These instructions occupy two bytes.

- **Indexed (16-bit Offset)**

Refer to Figure 23. This addressing mode calculates the EA by adding the contents of the two bytes following the opcode to the index register. Thus, the entire memory space may be accessed. Instructions which use this addressing mode are three bytes long.

- **Bit Set/Clear**

Refer to Figure 24. This mode of addressing applies to instructions which can set or clear any bit on page zero. The lower three bits in the opcode specify the bit to be set or cleared while the byte following the opcode specifies the address in page zero.

- **Bit Test and Branch**

Refer to Figure 25. This mode of addressing applies to instructions which can test any bit in the first 256 locations ($00-$FF) and branch to any location relative to the PC. The byte to be tested is addressed by the byte following the opcode. The individual bit within that byte to be tested is addressed by the lower three bits of the opcode. The third byte is the relative address to be added to the program counter if the branch condition is met. These instructions are three bytes long. The value of the bit tested is written to the carry bit in the condition code register.

- **Implied**

Refer to Figure 26. The implied mode of addressing has no EA. All the information necessary to execute an instruction is contained in the opcode. Direct operations on the accumulator and the index register are included in this mode of addressing. In addition, control instructions such as SWI, RTI belong to this group. All implied addressing instructions are one byte long.

■ INSTRUCTION SET

The MCU has a set of 59 basic instructions. They can be divided into five different types: register/memory, read/modify/write, branch, bit manipulation, and control. The following paragraphs briefly explain each type. All the instructions within a given type are presented in individual tables.

- **Register/Memory Instructions**

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to Table 2.

- **Read/Modity/Write Instructions**

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read/modify/write instructions since it does not perform the write. Refer to Table 3.

- **Branch Instructions**

The branch instructions cause a branch from the program when a certain condition is met. Refer to Table 4.

- **Bit Manipulation Instructions**

These instructions are used on any bit in the first 256 bytes of the memory. One group either sets or clears. The other group performs the bit test and branch operations. Refer to Table 5.

- **Control Instructions**

The control instructions control the MCU operations during program execution. Refer to Table 6.

- **Alphabetical Listing**

The complete instruction set is given in alphabetical order in Table 7.

- **Opcode Map**

Table 8 is an opcode map for the instructions used on the MCU.

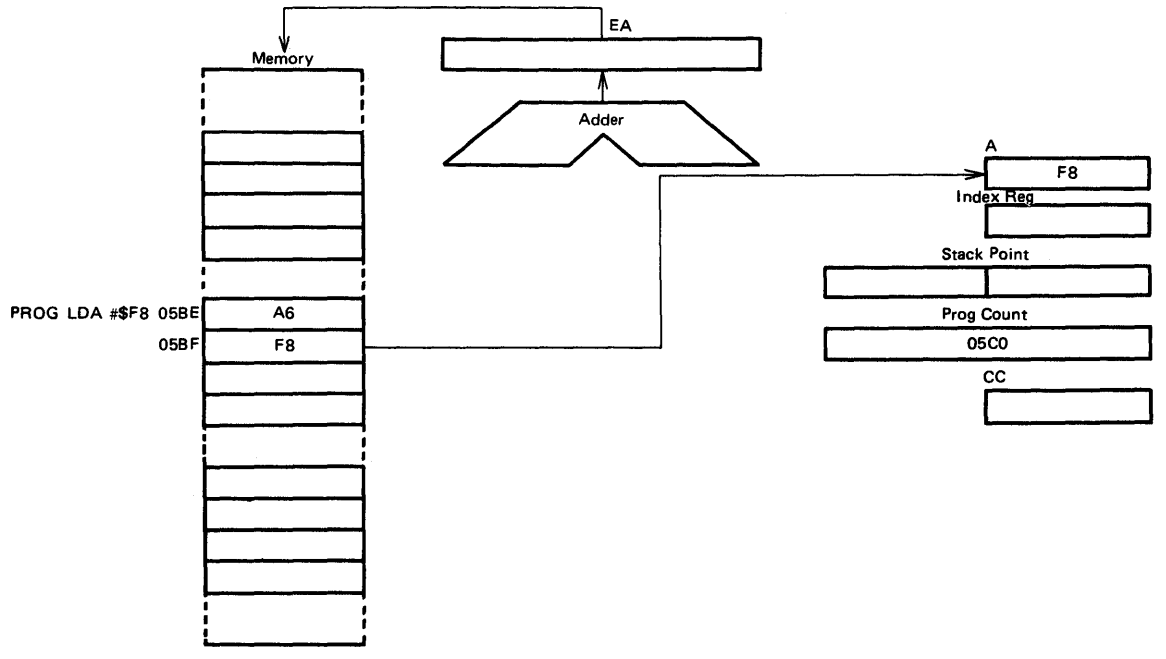


Figure 17 Immediate Addressing Example

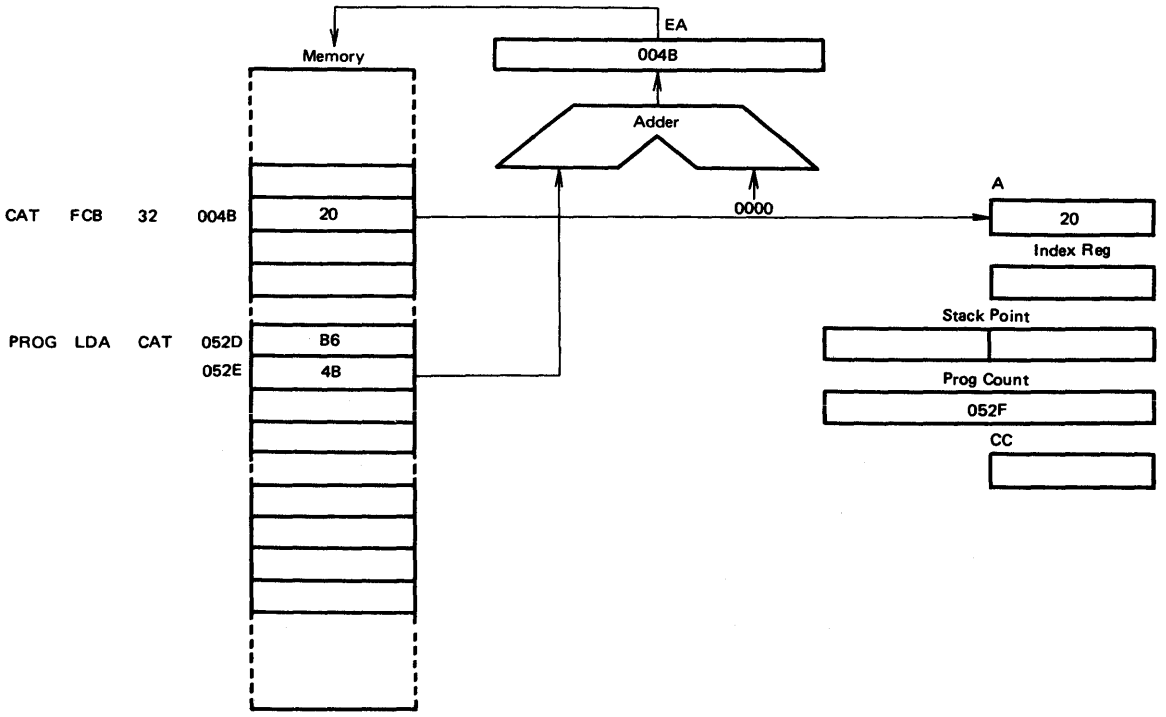


Figure 18 Direct Addressing Example

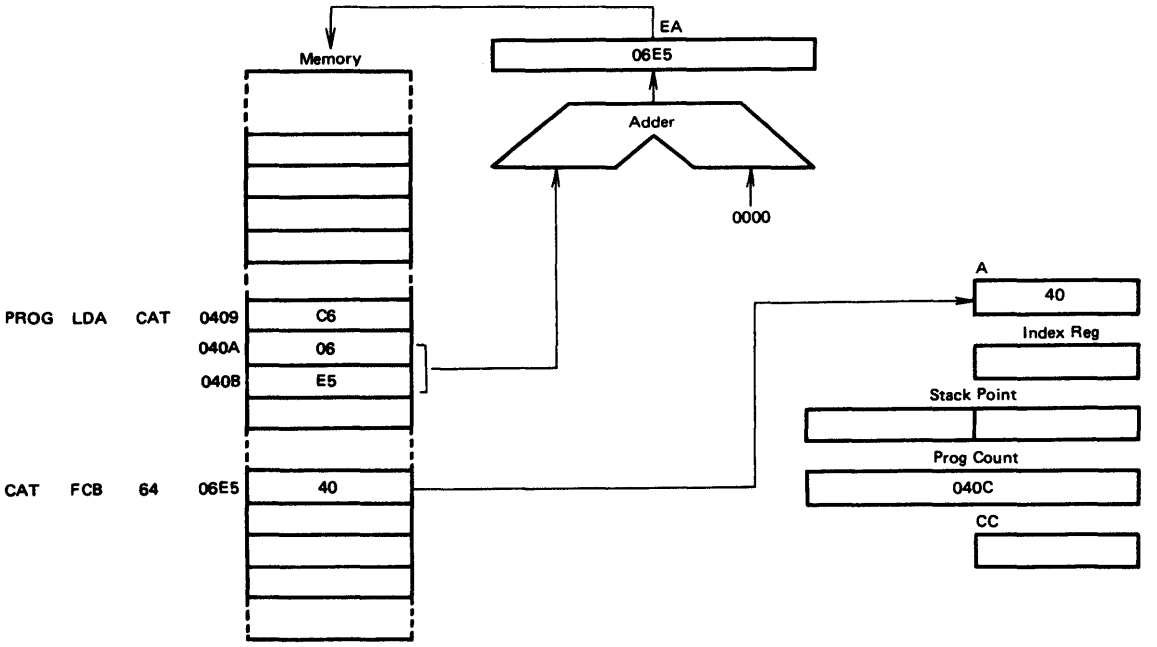


Figure 19 Extended Addressing Example

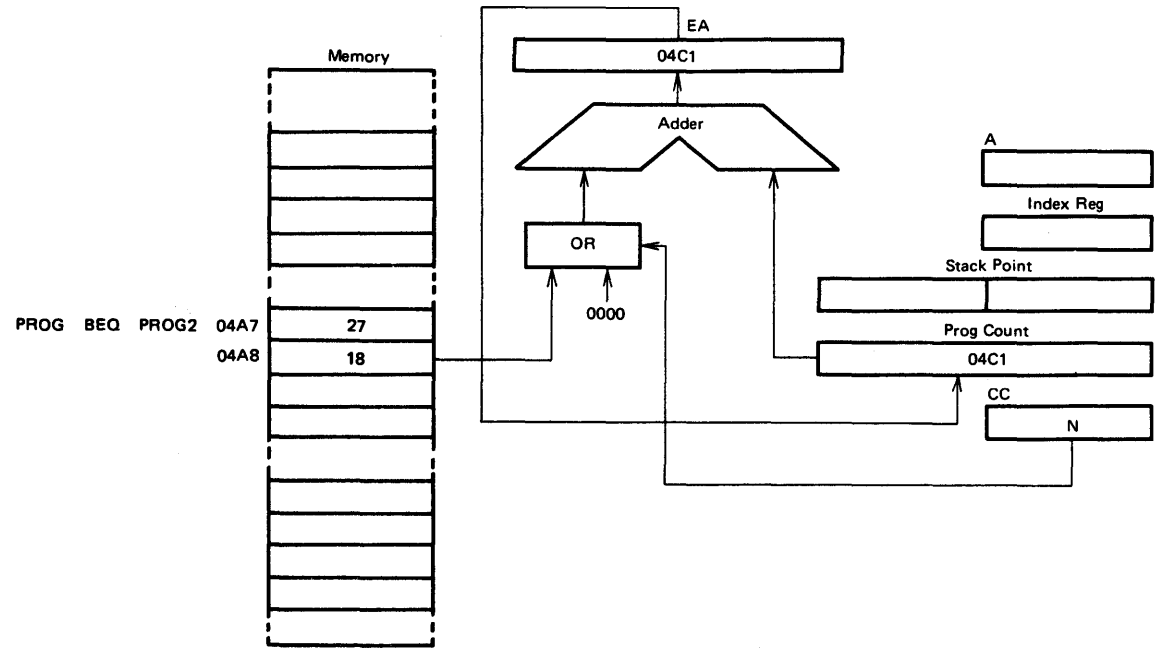


Figure 20 Relative Addressing Example

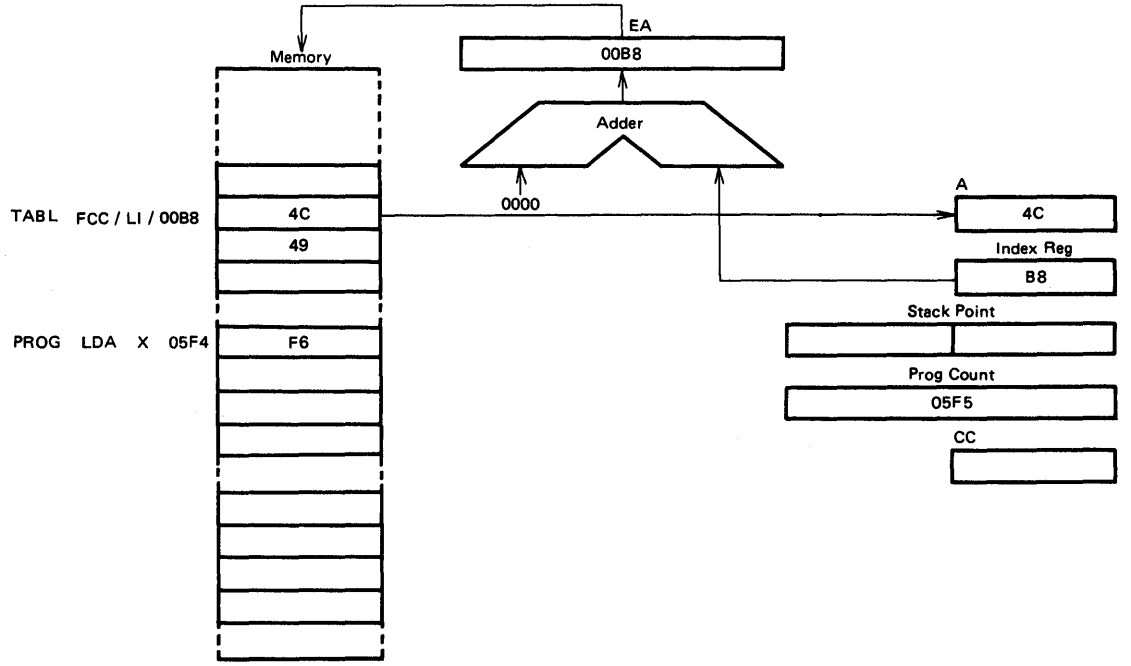


Figure 21 Indexed (No Offset) Addressing Example

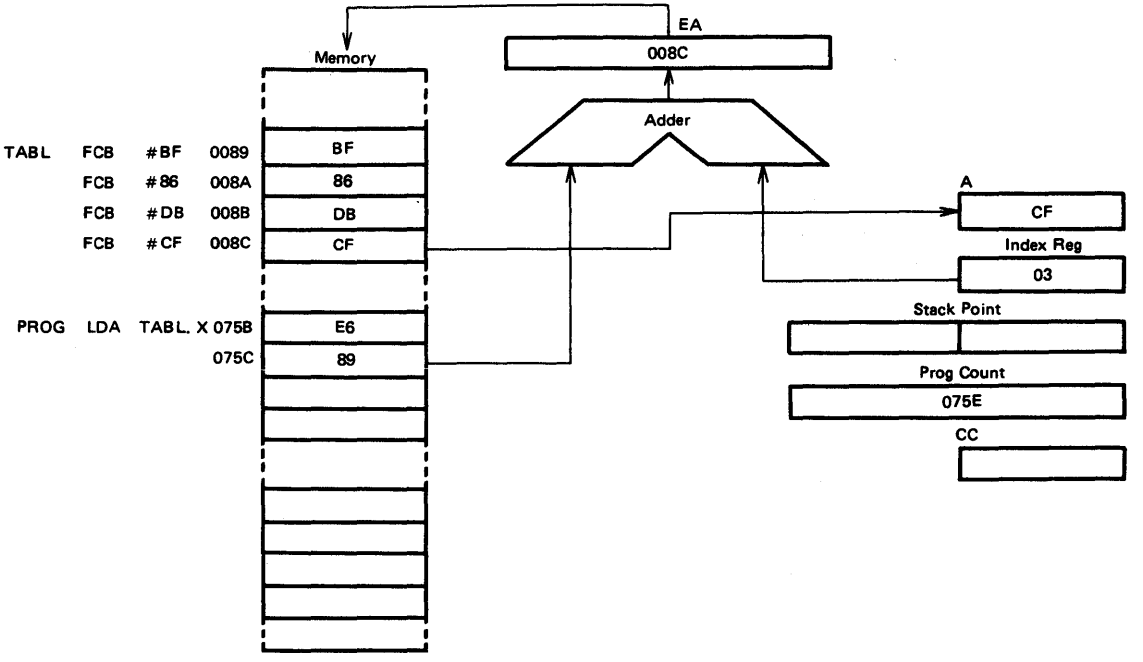


Figure 22 Indexed (8-Bit Offset) Addressing Example

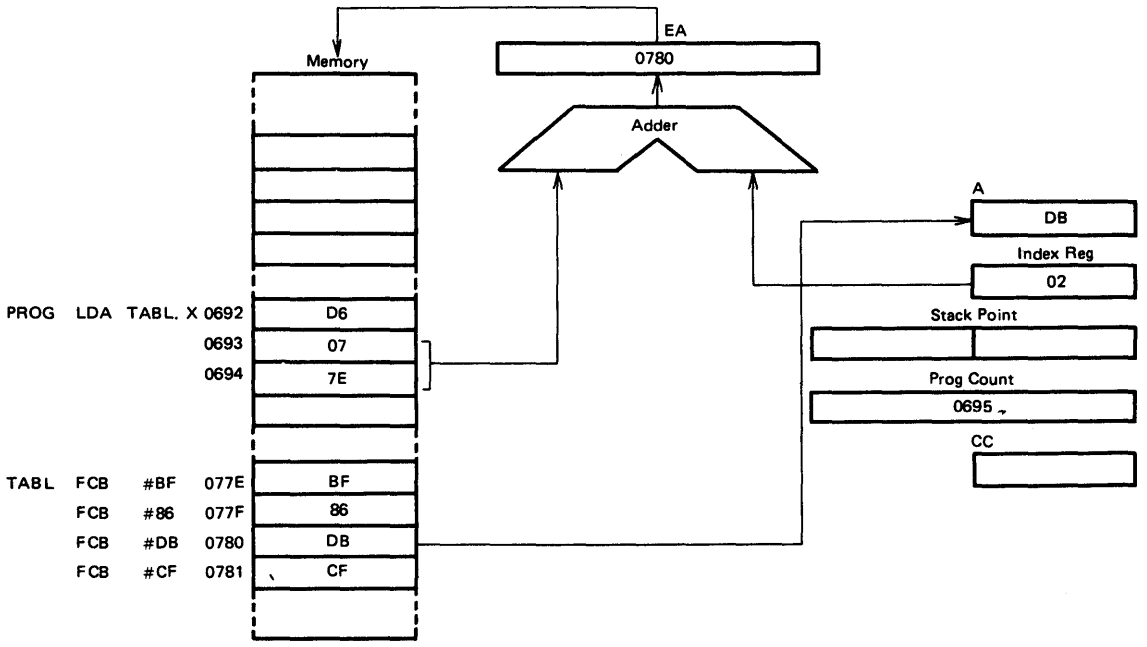


Figure 23 Indexed (16-Bit Offset) Addressing Example

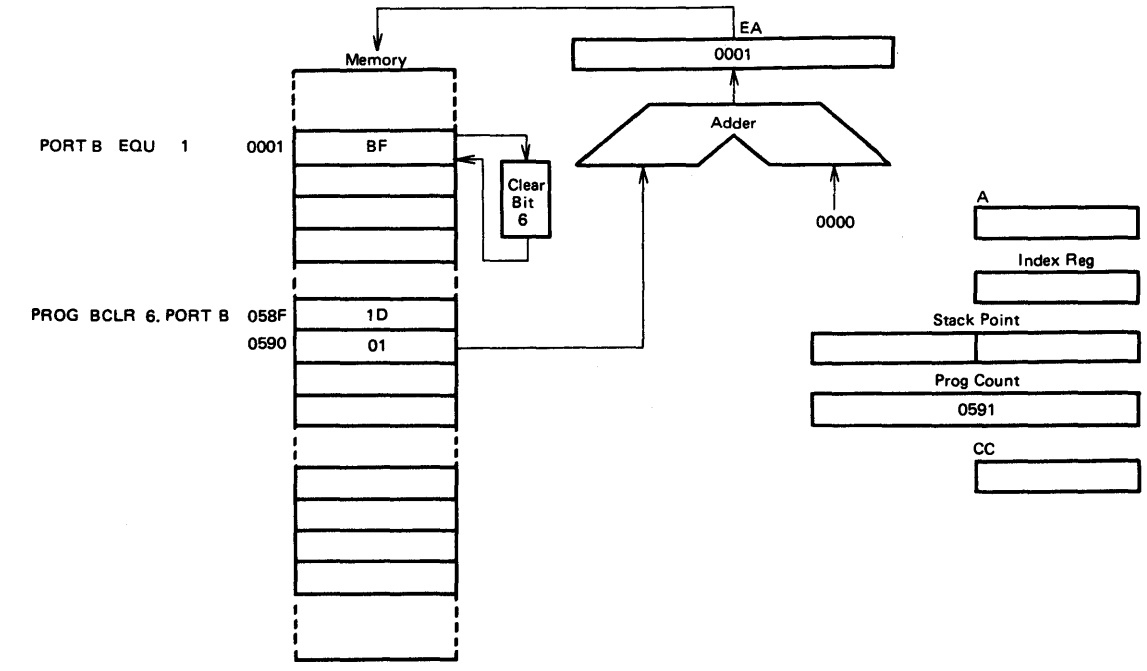


Figure 24 Bit Set/Clear Addressing Example

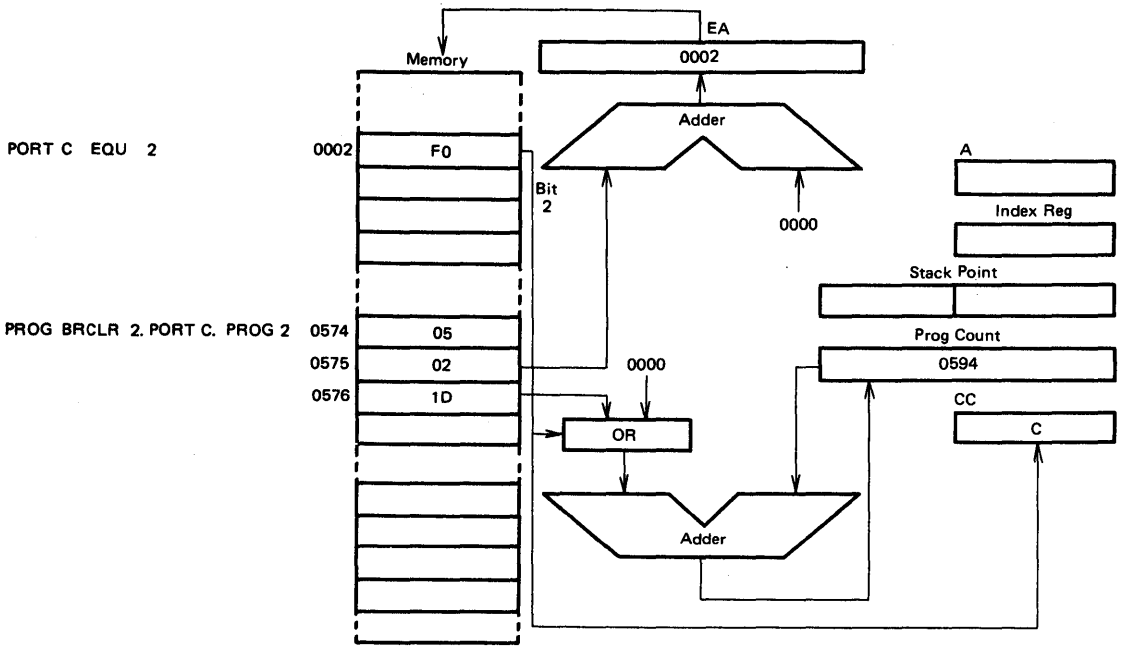


Figure 25 Bit Test and Branch Addressing Example

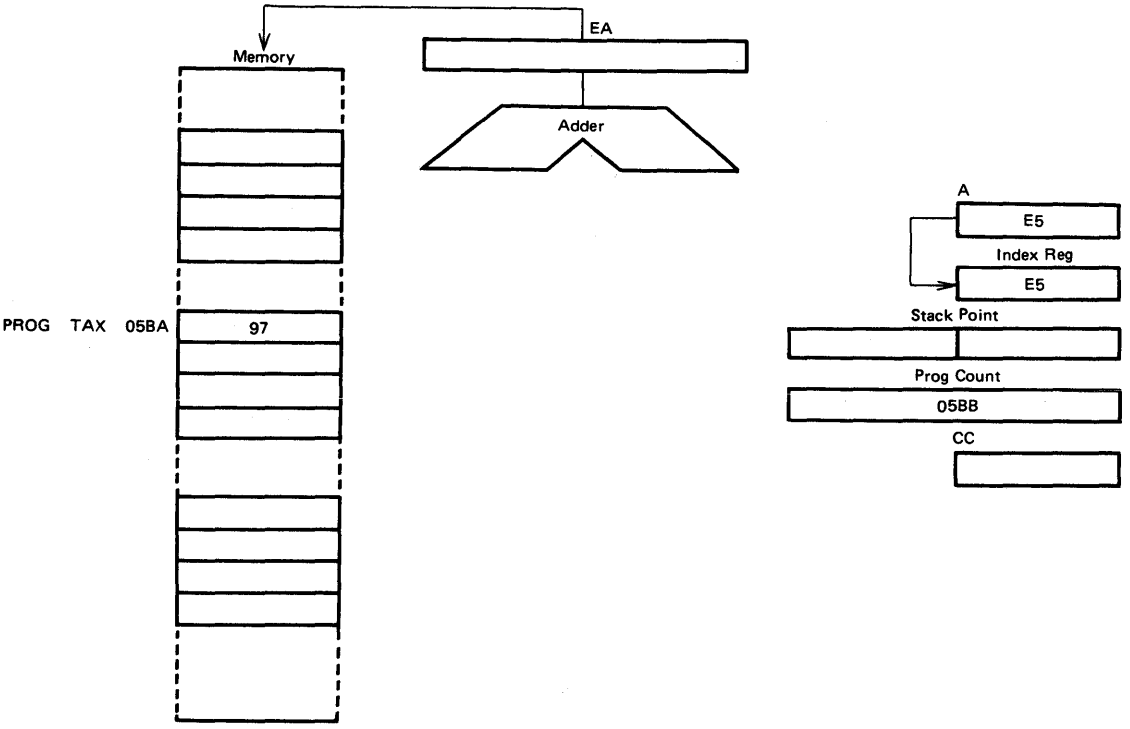


Figure 26 Implied Addressing Example

Table 2 Register/Memory Instructions

Function	Mnemonic	Addressing Modes																	
		Immediate			Direct			Extended			Indexed (No Offset)			Indexed (8-Bit Offset)			Indexed (16-Bit Offset)		
		Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Load A from Memory	LDA	A6	2	2	B6	2	4	C6	3	5	F6	1	4	E6	2	5	D6	3	6
Load X from Memory	LDX	AE	2	2	BE	2	4	CE	3	5	FE	1	4	EE	2	5	DE	3	6
Store A in Memory	STA	–	–	–	B7	2	5	C7	3	6	F7	1	5	E7	2	6	D7	3	7
Store X in Memory	STX	–	–	–	BF	2	5	CF	3	6	FF	1	5	EF	2	6	DF	3	7
Add Memory to A	ADD	AB	2	2	BB	2	4	CB	3	5	FB	1	4	EB	2	5	DB	3	6
Add Memory and Carry to A	ADC	A9	2	2	B9	2	4	C9	3	5	F9	1	4	E9	2	5	D9	3	6
Subtract Memory	SUB	A0	2	2	B0	2	4	C0	3	5	F0	1	4	E0	2	5	D0	3	6
Subtract Memory from A with Borrow	SBC	A2	2	2	B2	2	4	C2	3	5	F2	1	4	E2	2	5	D2	3	6
AND Memory to A	AND	A4	2	2	B4	2	4	C4	3	5	F4	1	4	E4	2	5	D4	3	6
OR Memory with A	ORA	AA	2	2	BA	2	4	CA	3	5	FA	1	4	EA	2	5	DA	3	6
Exclusive OR Memory with A	EOR	A8	2	2	B8	2	4	C8	3	5	F8	1	4	E8	2	5	D8	3	6
Arithmetic Compare A with Memory	CMP	A1	2	2	B1	2	4	C1	3	5	F1	1	4	E1	2	5	D1	3	6
Arithmetic Compare X with Memory	CPX	A3	2	2	B3	2	4	C3	3	5	F3	1	4	E3	2	5	D3	3	6
Bit Test Memory with A (Logical Compare)	BIT	A5	2	2	B5	2	4	C5	3	5	F5	1	4	E5	2	5	D5	3	6
Jump Unconditional	JMP	–	–	–	BC	2	3	CC	3	4	FC	1	3	EC	2	4	DC	3	5
Jump to Subroutine	JSR	–	–	–	BD	2	7	CD	3	8	FD	1	7	ED	2	8	DD	3	9

Table 3 Read/Modify/Write Instructions

Function	Mnemonic	Addressing Modes														
		Implied (A)			Implied (X)			Direct			Indexed (No Offset)			Indexed (8-Bit Offset)		
		Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Increment	INC	4C	1	4	5C	1	4	3C	2	6	7C	1	6	6C	2	7
Decrement	DEC	4A	1	4	5A	1	4	3A	2	6	7A	1	6	6A	2	7
Clear	CLR	4F	1	4	5F	1	4	3F	2	6	7F	1	6	6F	2	7
Complement	COM	43	1	4	53	1	4	33	2	6	73	1	6	63	2	7
Negate (2's Complement)	NEG	40	1	4	50	1	4	30	2	6	70	1	6	60	2	7
Rotate Left Thru Carry	ROL	49	1	4	59	1	4	39	2	6	79	1	6	69	2	7
Rotate Right Thru Carry	ROR	46	1	4	56	1	4	36	2	6	76	1	6	66	2	7
Logical Shift Left	LSL	48	1	4	58	1	4	38	2	6	78	1	6	68	2	7
Logical Shift Right	LSR	44	1	4	54	1	4	34	2	6	74	1	6	64	2	7
Arithmetic Shift Right	ASR	47	1	4	57	1	4	37	2	6	77	1	6	67	2	7
Arithmetic Shift Left	ASL	48	1	4	58	1	4	38	2	6	78	1	6	68	2	7
Test for Negative or Zero	TST	4D	1	4	5D	1	4	3D	2	6	7D	1	6	6D	2	7

Table 4 Branch Instructions

Function	Mnemonic	Relative Addressing Mode		
		Op Code	# Bytes	# Cycles
Branch Always	BRA	20	2	4
Branch Never	BRN	21	2	4
Branch IF Higher	BHI	22	2	4
Branch IF Lower or Same	BLS	23	2	4
Branch IF Carry Clear	BCC	24	2	4
(Branch IF Higher or Same)	(BHS)	24	2	4
Branch IF Carry Set	BCS	25	2	4
(Branch IF Lower)	(BLO)	25	2	4
Branch IF Not Equal	BNE	26	2	4
Branch IF Equal	BEQ	27	2	4
Branch IF Half Carry Clear	BHCC	28	2	4
Branch IF Half Carry Set	BHCS	29	2	4
Branch IF Plus	BPL	2A	2	4
Branch IF Minus	BMI	2B	2	4
Branch IF Interrupt Mask Bit is Clear	BMC	2C	2	4
Branch IF Interrupt Mask Bit is Set	BMS	2D	2	4
Branch IF Interrupt Line is Low	BIL	2E	2	4
Branch IF Interrupt Line is High	BIH	2F	2	4
Branch to Subroutine	BSR	AD	2	8

Table 5 Bit Manipulation Instructions

Function	Mnemonic	Addressing Modes					
		Bit Set/Clear			Bit Test and Branch		
		Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Branch IF Bit n is set	BRSET n (n=0 ..... 7)	—	—	—	2•n	3	10
Branch IF Bit n is clear	BRCLR n (n=0 .....7)	—	—	—	01+2•n	3	10
Set Bit n	BSET n (n=0 ..... 7)	10+2•n	2	7	—	—	—
Clear bit n	BCLR n (n=0 .....7)	11+2•n	2	7	—	—	—

Table 6 Control Instructions

Function	Mnemonic	Implied		
		Op Code	# Bytes	# Cycles
Transfer A to X	TAX	97	1	2
Transfer X to A	TXA	9F	1	2
Set Carry Bit	SEC	99	1	2
Clear Carry Bit	CLC	98	1	2
Set Interrupt Mask Bit	SEI	9B	1	2
Clear Interrupt Mask Bit	CLI	9A	1	2
Software Interrupt	SWI	83	1	11
Return from Subroutine	RTS	81	1	6
Return from Interrupt	RTI	80	1	9
Reset Stack Pointer	RSP	9C	1	2
No-Operation	NOP	9D	1	2

Table 7 Instruction Set

Mnemonic	Addressing Modes										Condition Code				
	Implied	Immediate	Direct	Extended	Relative	Indexed (No Offset)	Indexed (8 Bits)	Indexed (16 Bits)	Bit Set/ Clear	Bit Test & Branch	H	I	N	Z	C
ADC		x	x	x		x	x	x			∧	•	∧	∧	∧
ADD		x	x	x		x	x	x			∧	•	∧	∧	∧
AND		x	x	x		x	x	x			•	•	∧	∧	•
ASL	x		x			x	x				•	•	∧	∧	∧
ASR	x		x			x	x				•	•	∧	∧	∧
BCC					x						•	•	•	•	•
BCLR									x		•	•	•	•	•
BCS					x						•	•	•	•	•
BEQ					x						•	•	•	•	•
BHCC					x						•	•	•	•	•
BHCS					x						•	•	•	•	•
BHI					x						•	•	•	•	•
BHS					x						•	•	•	•	•
BIH					x						•	•	•	•	•
BIL					x						•	•	•	•	•
BIT		x	x	x		x	x	x			•	•	∧	∧	•
BLO					x						•	•	•	•	•
BLS					x						•	•	•	•	•
BMC					x						•	•	•	•	•
BMI					x						•	•	•	•	•
BMS					x						•	•	•	•	•
BNE					x						•	•	•	•	•
BPL					x						•	•	•	•	•
BRA					x						•	•	•	•	•
BRN					x						•	•	•	•	•
BRCLR										x	•	•	•	•	∧
BRSET										x	•	•	•	•	∧
BSET									x		•	•	•	•	•
BSR					x						•	•	•	•	•
CLC	x										•	•	•	•	0
CLI	x										•	0	•	•	•
CLR	x		x			x	x				•	•	0	1	•
CMP		x	x	x		x	x	x			•	•	∧	∧	∧
COM	x		x			x	x				•	•	∧	∧	1
CPX		x	x	x		x	x	x			•	•	∧	∧	∧
DEC	x		x			x	x				•	•	∧	∧	•
EOR		x	x	x		x	x	x			•	•	∧	∧	•
INC	x		x			x	x				•	•	∧	∧	•
JMP			x	x		x	x	x			•	•	•	•	•
JSR			x	x		x	x	x			•	•	•	•	•
LDA		x	x	x		x	x	x			•	•	∧	∧	•
LDX		x	x	x		x	x	x			•	•	∧	∧	•

(to be continued)

Condition Code Symbols:
- H Half Carry (From Bit 3)
- I Interrupt Mask
- N Negative (Sign Bit)
- Z Zero
- C Carry Borrow
- ∧ Test and Set if True, Cleared Otherwise
- • Not Affected

Table 7 Instruction Set

Mnemonic	Addressing Modes										Condition Code				
	Implied	Immediate	Direct	Extended	Relative	Indexed (No Offset)	Indexed (8 Bits)	Indexed (16 Bits)	Bit Set/ Clear	Bit Test & Branch	H	I	N	Z	C
LSL	x		x			x	x				•	•	∧	∧	∧
LSR	x		x			x	x				•	•	0	∧	∧
NEG	x		x			x	x				•	•	∧	∧	∧
NOP	x										•	•	•	•	•
ORA		x	x	x		x	x	x			•	•	∧	∧	•
ROL	x		x			x	x				•	•	∧	∧	∧
ROR	x		x			x	x				•	•	∧	∧	∧
RSP	x										•	•	•	•	•
RTI	x										?	?	?	?	?
RTS	x										•	•	•	•	•
SBC		x	x	x		x	x	x			•	•	∧	∧	∧
SEC	x										•	•	•	•	1
SEI	x										•	1	•	•	•
STA			x	x		x	x	x			•	•	∧	∧	•
STX			x	x		x	x	x			•	•	∧	∧	•
SUB		x	x	x		x	x	x			•	•	∧	∧	∧
SWI	x										•	1	•	•	•
TAX	x										•	•	•	•	•
TST	x		x			x	x				•	•	∧	∧	•
TXA	x										•	•	•	•	•

Condition Code Symbols:

- H Half Carry (From Bit 3)
- I Interrupt Mask
- N Negative (Sign Bit)
- Z Zero
- C Carry/Borrow
- ∧ Test and Set if True, Cleared Otherwise
- • Not Affected
- ? Load CC Register From Stack

Table 8 Opcode Map

	Bit Manipulation		Branch	Read/Modify/Write					Control		Register/Memory						
	Test & Branch	Set/ Clear	Rel	DIR	A	X	,X1	,X0	IMP	IMP	IMM	DIR	EXT	,X2	,X1	,X0	
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	← HIGH
0	BRSET0	BSET0	BRA	NEG					RTI*	–	SUB						0
1	BRCLR0	BCLR0	BRN	–					RTS*	–	CMP						1
2	BRSET1	BSET1	BHI	–					–	–	SBC						2
3	BRCLR1	BCLR1	BLS	COM					SWI*	–	CPX						3 L
4	BRSET2	BSET2	BCC	LSR					–	–	AND						4 O
5	BRCLR2	BCLR2	BCS	–					–	–	BIT						5 W
6	BRSET3	BSET3	BNE	ROR					–	–	LDA						6
7	BRCLR3	BCLR3	BEQ	ASR					–	TAX	–	STA(+1)					7
8	BRSET4	BSET4	BHCC	LSL/ASL					–	CLC	EOR						8
9	BRCLR4	BCLR4	BHCS	ROL					–	SEC	ADC						9
A	BRSET5	BSET5	BPL	DEC					–	CLI	ORA						A
B	BRCLR5	BCLR5	BMI	–					–	SEI	ADD						B
C	BRSET6	BSET6	BMC	INC					–	RSP	–	JMP(–1)					C
D	BRCLR6	BCLR6	BMS	TST					–	NOP	BSR*	JSR(–3)					D
E	BRSET7	BSET7	BIL	–					–	–	LDX						E
F	BRCLR7	BCLR7	BIH	CLR					–	TXA	–	STX(+1)					F
	3/10	2/7	2/4	2/6	1/4	1/4	2/7	1/6	1/*	1/2	2/2	2/4	3/5	3/6	2/5	1/4	

(NOTE) 1. Undefined opcodes are marked with "–".
2. The number at the bottom of each column denote the number of bytes and the number of cycles required (Bytes/Cycles). Mnemonics followed by a "*" require a different number of cycles as follows:

RTI	9
RTS	6
SWI	11
BSR	8

3. ( ) indicate that the number in parenthesis must be added to the cycle count for that instruction.

HD6805W0

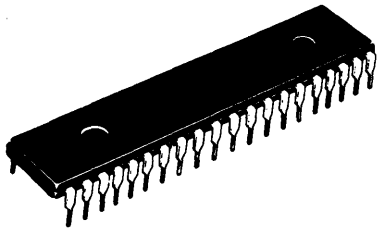
MCU (Microcomputer Unit)

—PRELIMINARY—

The HD6805W0 is an 8-bit microcomputer unit (MCU) which contains a CPU, on-chip clock, ROM, RAM, standby RAM, A/D Converter, I/O and two timers. This MCU is a member of the HD6805 family but compared with HD6805S, it is a single-chip microcomputer with strengthened internal functions of standby RAM, A/D Converter, timers and I/O.

The following are some of the hardware and software highlights of the MCU.

HD6805W0P



(DP-40)

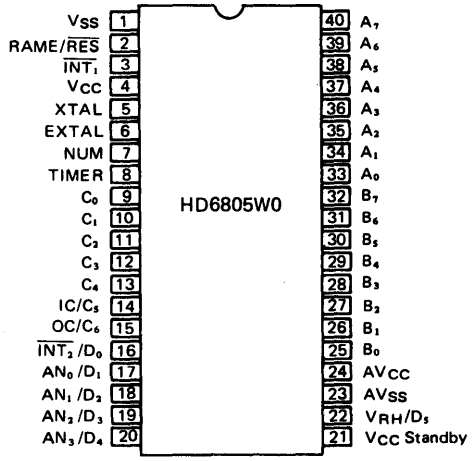
■ HARDWARE FEATURES

- 8-Bit Architecture
- 96 Bytes of RAM
  (8 bytes are standby RAM functions)
- Memory Mapped I/O
- 3834 Bytes of User ROM
- Internal 8-Bit Timer (Timer 1) with 7-Bit Prescaler
- Internal 8-Bit Programmable Timer (Timer 2)
- Interrupts – 2 External and 4 Timers
- 23 TTL/CMOS compatible I/O Lines; 8 Lines LED Direct Drive
- 8-Bit, 4-channel Internal A/D Converter
- Internal Clock Circuit
- Self-Check Mode
- Master Reset
- Low Voltage Inhibit
- Complete Development System Support by Evaluation Kit
- 5 Vdc Single Supply

■ SOFTWARE FEATURES

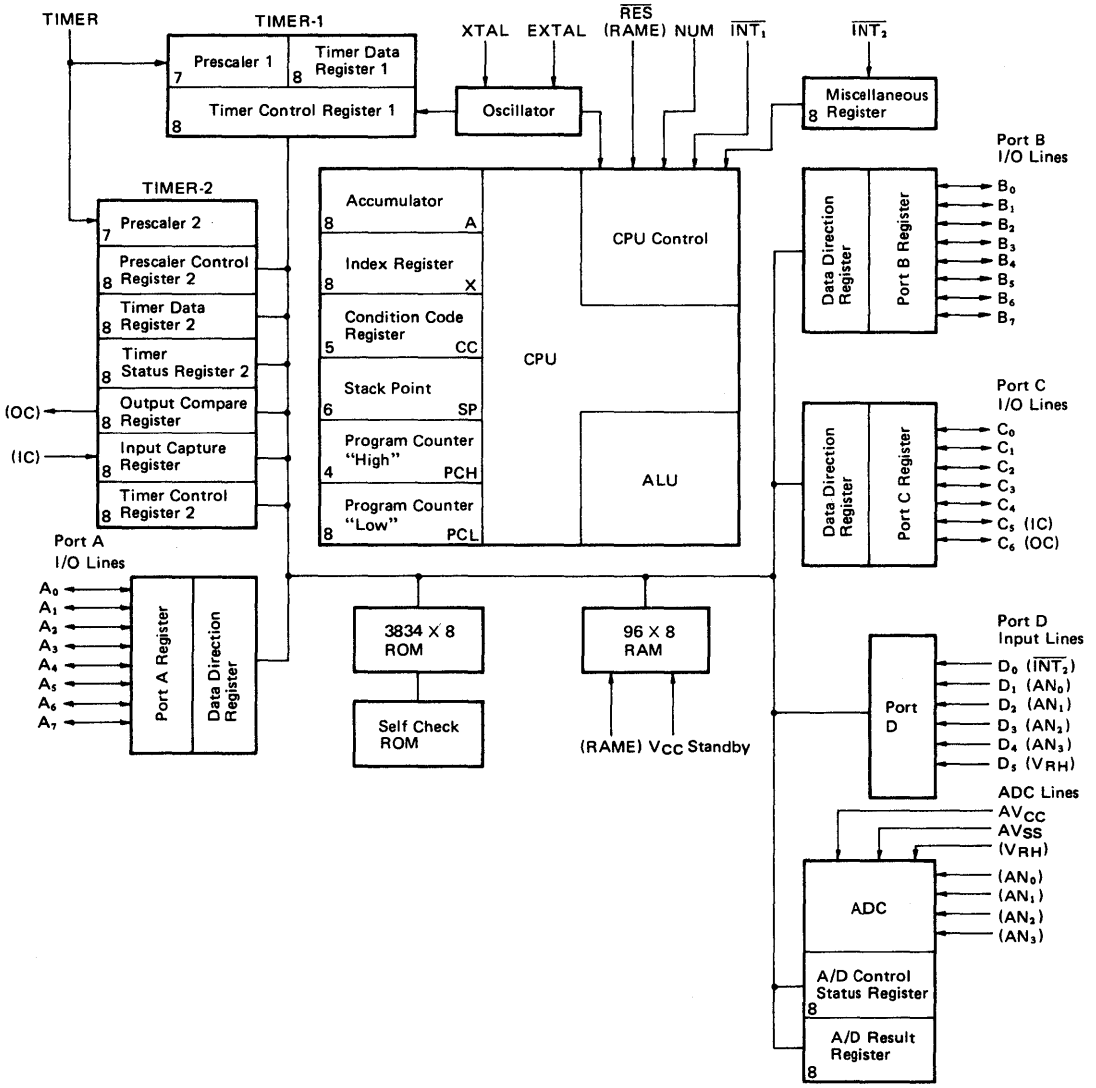
- Similar to HD6800
- Byte Efficient Instruction Set
- Easy to Program
- True Bit Manipulation
- Bit Test and Branch Instructions
- Versatile Interrupt Handing
- Powerful Indexed Addressing for Tables
- Full Set of Conditional Branches
- Memory Usable as Registers/Flags
- Single Instruction Memory Examine/Change
- 10 Powerful Addressing Modes
- All Addressing Modes Apply to ROM, RAM and I/O
- Compatible with MC6805P2, HD6805S1 and HD6805V1

■ PIN ARRANGEMENT



(Top View)

■ BLOCK DIAGRAM



(NOTE) The contents of ( ) items can be changed by software.

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC}	-0.3 ~ +7.0	V
Input Voltage (EXCEPT TIMER)	V_{in}	-0.3 ~ +7.0	V
Input Voltage (TIMER)		-0.3 ~ +15.0	V
Operating Temperature	T_{opr}	0 ~ +70	°C
Storage Temperature	T_{stg}	-55 ~ +150	°C

(NOTE) This device has an input protection circuit for high quiescent voltage and field, however, be careful not to impress a high input voltage than the insulation maximum value to the high input impedance circuit. To insure normal operation, the following are recommended for V_{in} and V_{out}:
$V_{SS} \leqq (V_{in}$ or $V_{out}) \leqq V_{CC}$

■ ELECTRICAL CHARACTERISTICS

● DC CHARACTERISTICS (V_{CC} = 5.25V ±0.5V, V_{SS} = GND, Ta = 0 ~ +70°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Input "High" Voltage	$\overline{RES}$	V_{IH}		4.0	–	V_{CC}	V
	$\overline{INT}$			3.0	–	V_{CC}	V
	All Others			2.0	–	V_{CC}	V
Input "High" Voltage Timer	Timer Mode			2.0	–	V_{CC}	V
	Self-Check Mode			9.0	–	15.0	V
Input "Low" Voltage	$\overline{RES}$	V_{IL}		-0.3	–	0.8	V
	$\overline{INT}$			-0.3	–	0.8	V
	All Others (except XTAL)			-0.3	–	0.8	V
Power Dissipation		P_D		–	–	850	mW
Low Voltage Recover		LVR		–	–	4.75	V
Low Voltage Inhibit		LVI		–	4.0	–	V
Input Leak Current	TIMER	I_{IL}	V_{in}=0.4V~V_{CC}	-20	–	20	μA
	$\overline{INT}$			-50	–	50	μA
	XTAL(Crystal Mode)			-1200	–	0	μA
Standby Voltage	Nonoperation Time	V_{SBB}		4.0	–	5.75	V
	Operation Time	V_{SB}		4.75	–	5.75	
Standby Current	Nonoperation Time	I_{SBB}	V_{SBB}=4.0V	–	–	3	mA

● AC CHARACTERISTICS (V_{CC} = 5.25V ±0.5V, V_{SS} = GND, Ta = 0 ~ +70°C, unless otherwise noted.)

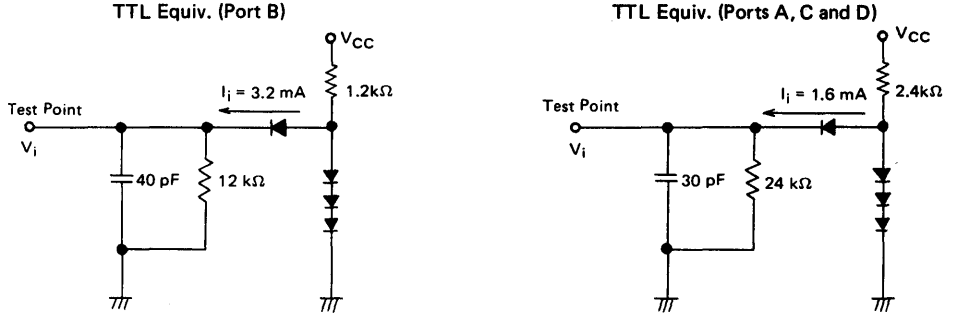
Item		Symbol	Test Condition	min	typ	max	Unit
Clock Frequency		f_{cl}		0.4	–	4.0	MHz
Cycle Time		t_{cyc}		1.0	–	10	μs
Oscillation Frequency (External Resister Mode)		f_{EXT}	R_{CP}=15.0kΩ±1%	2.7	–	4.0	MHz
$\overline{INT}$ Pulse Width		t_{IWL}		t_{cyc}+250	–	–	ns
$\overline{RES}$ Pulse Width		t_{RWL}		t_{cyc}+250	–	–	ns
TIMER Pulse Width		t_{TWL}		t_{cyc}+250	–	–	ns
Oscillation Start-up Time (Crystal Mode)		t_{OSC}	C_L=22pF±20% R_S=60Ω max.	–	–	100	ms
Delay Time Reset		t_{RHL}	External Cap. = 2.2μF	100	–	–	ms
Input Capacitance	EXTAL	C_{in}	V_{in}=0V	–	25	30	pF
	All Others			–	6	10	pF

- **PORT ELECTRICAL CHARACTERISTICS (V_{CC} = 5.25V ±0.5V, V_{SS} = GND, Ta = 0 ~ +70°C, unless otherwise noted.)**

Item		Symbol	Test Condition	min	typ	max	Unit
Output "High" Voltage	Port A	V_{OH}	I_{OH} = -10 μA	3.5	–	–	V
			I_{OH} = -100 μA	2.4	–	–	V
	Port B		I_{OH} = -200 μA	2.4	–	–	V
			I_{OH} = -1 mA	1.5	–	–	V
	Port C		I_{OH} = -100 μA	2.4	–	–	V
Output "Low" Voltage	Ports A and C	V_{OL}	I_{OL} = 1.6 mA	–	–	0.4	V
	Port B		I_{OL} = 3.2 mA	–	–	0.4	V
			I_{OL} = 10 mA	–	–	1.0	V
Input "High" Voltage	Ports A, B, C and D	V_{IH}		2.0	–	V_{CC}	V
Input "Low" Voltage		V_{IL}		-0.3	–	0.8	V
Input Leak Current	Port A	I_{IL}	V_{in} = 0.8V	-500	–	–	μA
			V_{in} = 2V	-300	–	–	μA
	Ports B, C and D		V_{in} = 0.4V~V_{CC}	-20	–	20	μA

- **A/D CONVERTER ELECTRICAL CHARACTERISTICS (V_{CC} = 5.25V, ±0.5V, V_{SS} = AV_{SS} = GND, Ta = 0~+70°C, unless otherwise noted.)**

Item	Symbol	Test Condition	min	typ	max	Unit
Analog Power Supply Voltage	AV_{CC}		4.75	5.0	5.25	V
Analog Input Voltage	AV_{in}		0	–	5.0	V
Reference Voltage	VRH		–	5	AV_{CC}+0.25	V
Reference Voltage	AV_{SS}		-0.1	0	–	V
Analog Multiplexer Input Capacitance			–	–	7.5	pF
Ladder Resistance		VRH = 5.0V, VRL = 0V, Ta = 25°C	2.5	–	12.5	kΩ
Resolution Power			–	8	–	Bit
Conversion Time		at 4MHz	–	76	–	μs
Input Channels			–	4	–	Channel
Non-Linearity Error		Ta = 25°C, AV_{CC} = 5.0V, VRH = 5.0V	–	–	±1/2	LSB
Offset Error		Ta = 25°C, AV_{CC} = 5.0V, VRH = 5.0V	–	±1/4	±3/8	LSB
Full-scale Error		Ta = 25°C, AV_{CC} = 5.0V, VRH = 5.0V	–	±1/4	±3/8	LSB
Quantum Error		Ta = 25°C, AV_{CC} = 5.0V, VRH = 5.0V	–	–	±1/2	LSB
Absolute Accuracy		Ta = 25°C, AV_{CC} = 5.0V, VRH = 5.0V	–	–	1.0	LSB
Off-channel Leak Current		AV_{in} = 5.0V, AV_{CC} = 4.75V, Ta = 25°C, On-channel AV_{in} = 0V	–	10	100	nA
Off-channel Leak Current		AV_{in} = 0V, AV_{CC} = 4.75V, Ta = 25°C, On-channel AV_{in} = 5V	-100	-10	–	nA



(NOTE) 1. Load capacitance includes the floating capacitance of the probe and the jig etc.
2. All diodes are 1S2074Ⓗ or equivalent.

Figure 1 Bus Timing Test Loads

■ SIGNAL DESCRIPTION

The input and output signals for the MCU shown in PIN ARRANGEMENT are described in the following paragraphs.

• V_{CC} and V_{SS}

Voltage is supplied to the MCU using these two pins. V_{CC} is 5.25V ±0.5V. V_{SS} is the ground connection.

• $\overline{INT_1}/\overline{INT_2}$

This pin provides the capability for applying an external interrupt to the MCU. Refer to INTERRUPTS for additional information.

• XTAL and EXTAL

These pins provide control input for the on-chip clock circuit. A crystal (AT cut, 4MHz maximum) or a resistor can be connected to these pins to provide the internal oscillator with varying degrees of stability. Refer to INTERNAL OSCILLATOR OPTIONS for recommendations about these inputs.

• TIMER

This pin allows an external input to be used to count for the internal timer circuitry. Refer to TIMER for additional information about the timer circuitry.

• $\overline{RES}$

This pin allows resetting of the MCU at times other than the automatic resetting capability already in the MCU. Refer to RESETS for additional information.

• NUM

This pin is not for user application and should be connected to ground.

• I/O Lines ($A_0 \sim A_7$, $B_0 \sim B_7$, $C_0 \sim C_6$)

There 23 lines are arranged as three ports (A, B and C). All lines are programmable as I/O under software control of the data direction registers. Refer to the section on INPUTS/OUTPUTS for details.

• Input Lines ($D_0 \sim D_5$)

Since the input for these 6 lines is TTL compatible, $003 address is read and they become Port D function.

• V_{CC} Standby

When source voltage V_{CC} is down, source voltage 5V ±5% is impressed to this pin to maintain standby RAM. The content of the low order 8 bytes are maintained when source voltage is off (3 mA max.). The circuit in Figure 2 is an example of a circuit for maintaining V_{CC} standby voltage when source voltage is off. To maintain the RAM contents when source voltage is off, the following hardware and software procedures are necessary.

(1) Software
- Write "0" into the RAM enable bit (RAME). RAME is bit 6 of the RAM control register location $01F. Since this operation disables the RAM standby part, contents are present with power source off.
- Maintain V_{CC} standby voltage above V_{SBB} (min.).

(2) Hardware
- Set RAME pin to "0" before setting V_{CC} to off.
- Maintain V_{CC} standby voltage above V_{SBB} (main).

When standby RAM is not needed, standby is connected to V_{CC}.

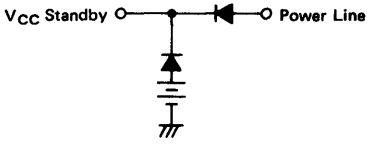


Figure 2 Battery Backup for V_{CC} Standby

• RAME

This pin, like $\overline{RES}$, is for the external control of standby RAM.

After power up, RAME is set to "0" and V_{CC} is set off. Since RAM is disabled, the RAM contents are maintained. The same type of function as this can be realized by software by setting the RAM control register ($01F) RAME (bit 6) to "0".

- **AV_{CC}**

This pin is used for the power supply of the A/D converter. When high accuracy is requested, a different power source than V_{CC} is impressed as

$AV_{CC} = 5.25 \pm 0.5V$

Connect to V_{CC} for all other cases.

- **$AN_0 \sim AN_3$**

This pin is used for A/D converter analog input. An analog signal is used for input during measure. These signals are switched by the internal multiplexer and the analog input selection uses bit 0 through bit 1 of the A/D control status register (ADCSR: $00E).

- **V_{RH} and AV_{SS}**

The input terminal reference voltage for the A/D converter is "High" (V_{RH}) or "Low" (AV_{SS}). AV_{SS} is fixed at 0V.

- **Input Capture (IC)**

The timer data register ($01C) contents are managed by input capture register ($01E) due to the positive or negative edge of this pin.

Specification of the positive or negative edge is set by bit 1 of the timer control register 2 ($01B). The specification is for positive edge when bit 1 is "1" and for the negative edge when bit 1 is set to "0". In this case, the DDR of port C_5 is set to "0".

- **Output Compare (OC)**

When the output compare register ($01D) and timer data register 2 ($01C) contents are the same, this pin is used for data output.

Data desired for output is specified by bit 0 of timer control register 2 ($01B). When bit 0 is "1", OC pin output is "1" and when bit 0 is "0", OC pin output is "0". In this case, DDR of port C_6 is set to "1".

■ MEMORY

The MCU memory is configured as shown in Figure 3. During the interrupt processing, the contents of the MCU registers are pushed onto the stack in the order shown in Figure 4. Since the stack pointer decrements during pushes, the low order byte (PCL) of the program counter is stacked first; then the high order three bits (PCH) are stacked. This ensures that the program counter is loaded correctly as the stack pointer increments when it pulls data from the stack. A subroutine call will cause only the program counter (PCH, PCL) contents to be pushed onto the stack.

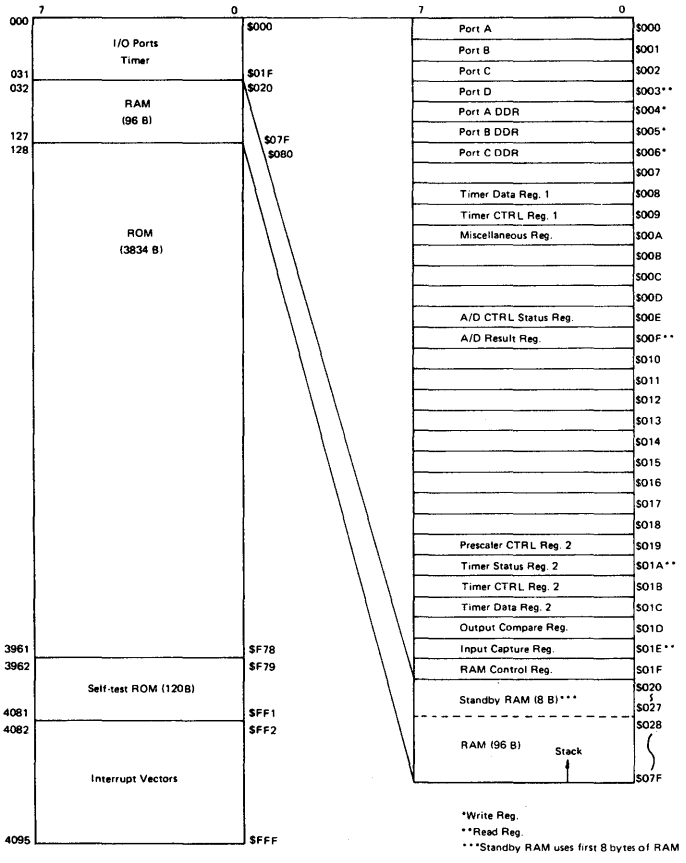


Figure 3 MCU Memory Structure

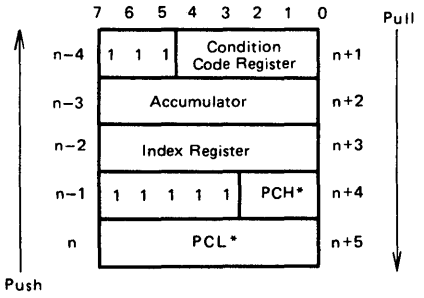


Figure 4 Interrupt Stacking Order

■ REGISTERS

The MCU has five registers available to the programmer, as shown in Figure 5 and explained below.

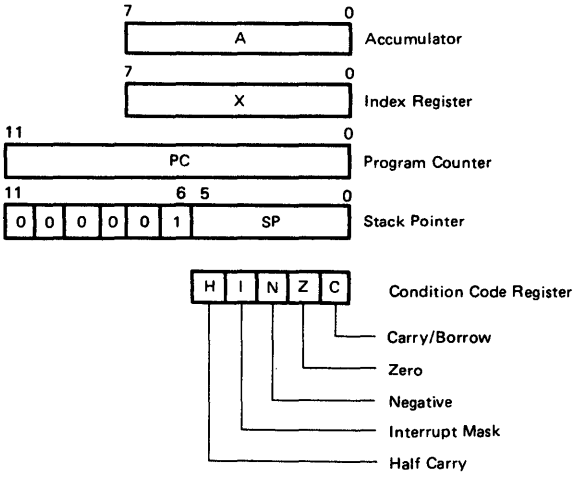


Figure 5 Programming Model

• Accumulator (A)

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.

• Index Register (X)

The index register is an 8-bit register used for the indexed addressing mode and contains an 8-bit address that may be added to an offset value to create an effective address. The index register can also be used for limited calculations or data manipulations when using read/modify/write instructions. When not required by a code sequence being executed, the index register can be used as a temporary storage area.

• Program Counter (PC)

The program counter is a 12-bit register that contains the address of the next instruction to be executed.

• Stack Pointer (SP)

The stack pointer is a 12-bit register that contains the address of the next free location on the stack. Initially, the stack pointer is set to location $07F and is decremented as data is being pushed onto the stack and incremented while data is being pulled from the stack. The six most significant bits of the stack pointer are permanently set to 000001. During an MCU reset or reset stack pointer (RSP) instruction, the stack pointer is set to location $07F. Subroutines and interrupts may be nested down to location $041 which allows the programmer to use up to 31 levels of subroutine calls.

• Condition Code Register (CC)

The condition code register is a 5-bit register in which each bit is used to indicate or flag the results of the instruction just executed. These bits can be individually tested by a program and specific action taken as a result of their state. Each individual condition code register bit is explained below.

• Half Carry (H)

The half carry bit is used during arithmetic operations (ADD or ADC) to indicate that a carry occurred between bits 3 and 4.

• Interrupt (I)

This bit is set to mask everything. If an interrupt occurs while this bit is set, it is latched and will be processed as soon as the interrupt bit is reset.

• Negative (N)

The negative bit is used to indicate that the result of the last arithmetic, logical or data manipulation was negative (bit 7 in a result equal to a logical one).

• Zero (Z)

Zero is used to indicate that the result of the last arithmetic, logical or data manipulation was zero.

• Carry/Borrow (C)

Carry/borrow is used to indicate that a carry or borrow out of the arithmetic logic unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions, shifts and rotates.

■ TIMER 1

The MCU timer circuitry is shown in Figure 6. The 8-bit counter is loaded under program control and counts down toward zero as soon as the clock input is applied. When the timer data register 1 (TDR1) reaches zero, the timer interrupt request bit (bit 7) in the timer control register 1 is set. The MCU responds to this interrupt by saving the present MCU state in the stack, fetching the timer 1 interrupt vector from locations $FF8 and $FF9 and executing the interrupt routine. The timer 1 interrupt can be masked by setting the timer interrupt mask bit (bit 6) in the timer control register 1. The interrupt bit (I bit) in the condition code register will also prevent a timer 1 interrupt from being processed.

The clock input to the timer 1 can be from an external source applied to the TIMER input pin or it can be the internal ϕ_2 signal. Note that when ϕ_2 is used as the source, it can be gated by an input applied to the TIMER input pin allowing the user to easily perform pulse-width measurements. The timer 1 continues to count past zero and its present count can be monitored at any time by monitoring the timer data register 1 This allows a program to determine the length of time since a

timer interrupt has occurred and not disturb the counting process.

At power up or reset, the prescaler and counter are initialized with all logical ones; the timer 1 interrupt request bit (bit 7) is cleared and the timer 1 interrupt request mask bit (bit 6) is set. To erase the timer 1 interrupt bit, "0" is written into TIF by software.

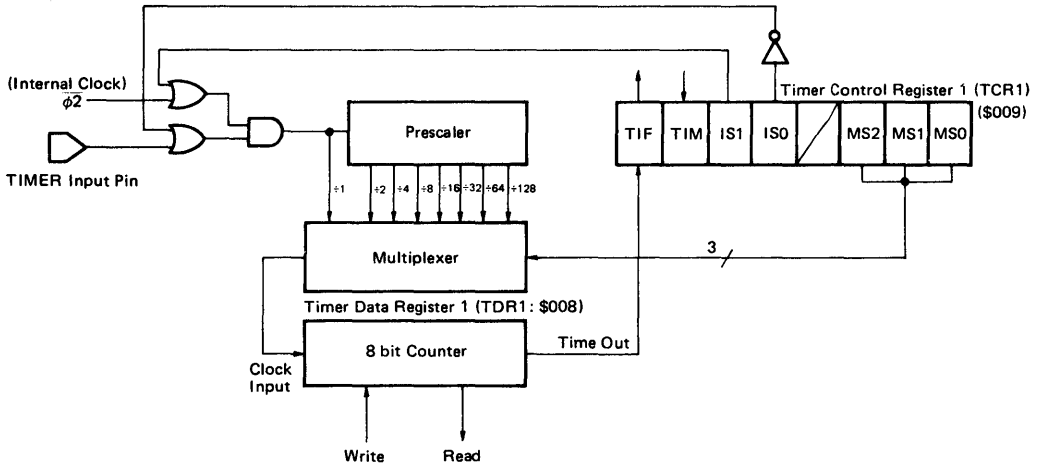
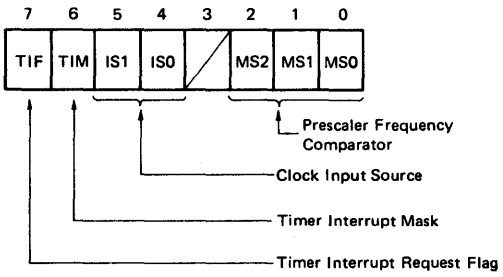


Figure 6 Timer Clock

- **Timer Control Register 1 (TCR1: $009)**

Clock input source selection, prescaler frequency compare selection and timer interrupt control are controlled by software written into timer control register 1 (TCR1: $009).

Timer Control Register 1 (TCR1: $009)



As shown in Table 1, clock input source selection is made by the three inputs; IS0 or IS1 (bits 4 and 5) of timer control register 1 (TCR1). After reset, internal clock ϕ_2 (bit 4 = 1 and bit 5 = 0) timer control is selected.

As shown in Table 2, prescaler frequency compare is selected by timer control register 1 (TCR1) three bits; MS0 through MS2 (bit 0 through bit 2).

Table 1 Clock Input Source Selection

TCR1		Clock Input Source
Bit 5	Bit 4	
0	0	Internal Clock, ϕ_2
0	1	Timer Pin Control ϕ_2
1	0	
1	1	Event Input From Timer Pin

Table 2 Prescaler Frequency Comparator Selection

TCR1			Prescaler Frequency Comparator
Bit 2	Bit 1	Bit 0	
0	0	0	÷ 1
0	0	1	÷ 2
0	1	0	÷ 4
0	1	1	÷ 8
1	0	0	÷ 16
1	0	1	÷ 32
1	1	0	÷ 64
1	1	1	÷ 128

■ TIMER 2

The HD6805W0 contains an on-chip programmable timer (timer 2) which may be used to perform measurements on an input waveform while independently generating an output waveform. Pulse widths for both input and output signals may vary from a few microseconds to many seconds. The timer 2 hardware consists of

- an 8-bit control register 2
- an 8-bit status register 2
- an 8-bit timer data register 2
- an 8-bit output compare register
- an 8-bit input capture register
- a 5-bit prescaler control register 2
- a 7-bit prescaler 2

A block diagram of the timer 2 is shown in Figure 7.

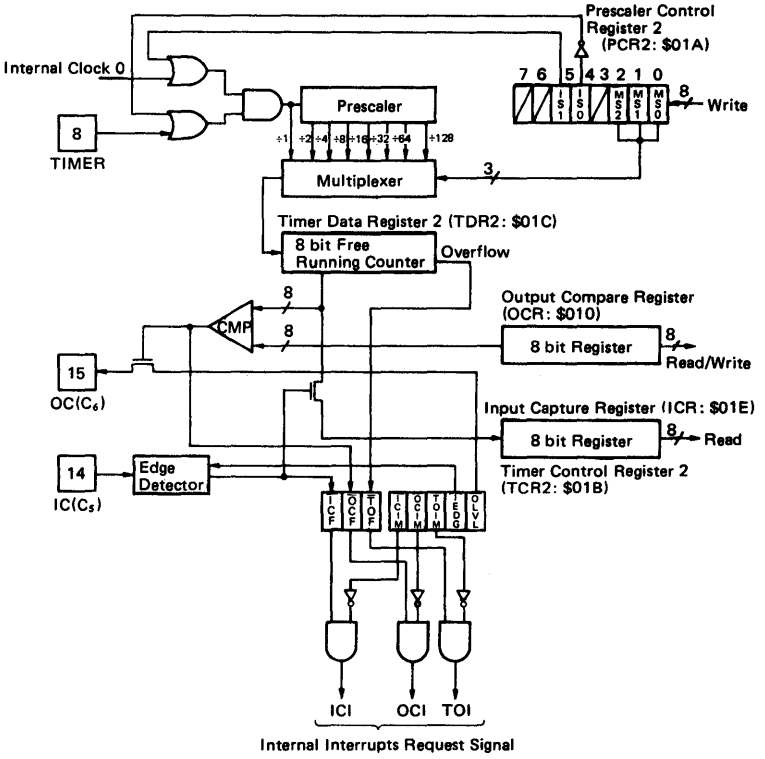


Figure 7 Timer 2 Block Diagram

• Timer Data Register 2 ($01C)

The key element in timer 2 is an 8-bit timer data register 2 free running counter which is driven to increasing values by the internal clock ϕ_2 or by timer input. The counter value may be read by the CPU software at any time. The timer data register 2 is cleared by reset. This counter can also read and write.

• Output Compare Register ($01D)

The output compare register is an 8-bit read/write register which is used to control an output waveform. The contents of this register are constantly compared with the current value of timer data register 2. When a match is found, a flag is set (OCF) in the timer status register 2 (TSR2) and the current value of the output level bit (OLVL) in TSR2 is clocked to port C bit 6. Providing the data direction register for port C, bit 6 contains a "1" (output), the output level register value will appear on the pin for port C, bit 6. The values in the output compare register and output level bit may then be changed to control the output level on the next compare value. The output compare register is set to $FF by reset.

• Input Capture Register ($01E)

The input capture register is an 8-bit read-only register used to store the current value of the timer data register 2 when the proper transition of an external input signal occurs. The input transition change required to trigger the counter transfer is controlled by the input edge bit (IEDG) in the TCR2. The data direction register bit for port C bit 5 should* be clear (zero) in

order to gate in the external input signal to the edge detect unit in the timer. In all cases, it is necessary to make the input pulse width at least 2 enable cycles in order to mantain the input capture.

*With port C bit 5 configured as an output and set to "1", the external input will still be seen by the edge detect unit.

- **Timer Control Register 2 (TCR2: $01B)**

The timer control register 2 consists of an 5-bit register of which all 5 bits are readable and writeable.

Timer Control Register 2 (TCR2: $01B)

7	6	5	4	3	2	1	0
			ICIM	OCIM	TOIM	IEDG	OLVL

Bit 0 OLVL Output Level

When this value is compared with the counter value and output compare register value, it is moved to bit 6 of Port C.

If DDR is set to "1" according to bit 6 of Port C, this value is output from the line of bit 6 of Port C.

Bit 1 IEDG Input Edge

This bit determines the trigger to change either polarity of input line bit 5 of Port C for data transmission to input capture register from timer/counter 2. When this function is used, it is necessary to clear DDR beforehand, according to Port C bit 5. When IEDG = 0, the negative edge triggers ("High" to "Low" transition). When IEDG = 1, the positive edge triggers ("Low" to "High" transition).

Bit 2 TOIM Timer Overflow Interrupt Mask

When this bit is cleared, internal interrupt (TOI) is enabled by TOF interrupt but when it is set, interrupt is inhibited.

Bit 3 OCIM Output Compare Interrupt Mask

When this bit is cleared, internal interrupt (OCI) by OCF interrupt occurs. When this bit is set, interrupt is inhibited.

Bit 4 ICIM Input Capture Interrupt Mask

When this bit is cleared, internal interrupt (ICI) by ICF interrupt occurs. When this bit is set, interrupt is inhibited.

- **Timer Status Register 2 (TSR2: $01A)**

The timer status register is a 3-bit read-only register which indicates that:

(1) A proper transition has taken place on the input pin with a subsequent transfer of the current counter value to the input capture register (ICF).
(2) A match has been found between the value in the counter and the output compare register (OCF).
(3) When $00 is in the counter (TOF)

Each of the flags in the TSR2 has interrupt and inhibit bits, by which the interrupt output request is controlled. If the I bit in the condition code register is cleared, priority vectors are generated in response to clearing these flags. Each bit is discussed below.

Timer Status Register 2 (TSR2: $01A)

7	6	5	4	3	2	1	0
ICF	OCF	TOF					

Bit 5 TOF Timer Overflow Flag

This read-only bit is set when the counter contains $00. It is cleared by a read of the TSR2, followed by a read of timer/counter 2 ($01B).

Bit 6 OCF Output Compare Flag

This read-only bit is set when a match is found between the output compare register and the free running counter. It is cleared by a read of the TSR2 followed by a CPU write to the output compare register ($01D).

Bit 7 ICF Input Capture Flag

This read-only bit is set by a change in IEDG specification detected by the edge detection circuit and cleared by a read of the TSR2 followed by a CPU read of the input capture register ($01E).

Although TCR2 and TSR2 are as described above, output compare output can be written into C_6 port by software. Accordingly, after C_6 port has been written into by software, simultaneous cyclic pulse control with a short width is easy.

- **Prescaler Control Register 2 (PCR2: $019)**

The selections of clock input source and prescaler frequency compare are performed by prescaler control register 2 (PCR2: $019).

Prescaler Control Register 2 (PCR2: $019)

7	6	5	4	3	2	1	0
		IS1	IS0		MS2	MS1	MS0

Prescaler Frequency Comparator

Clock Input Source

Selection of clock input source is performed in three different ways by bits 4 and 5 of prescaler control register 2 (PCR2), as shown in Table 3. After reset, internal clock ϕ_2 (bit 4 = 1 and bit 5 = 0) is selected by timer pin control. The prescaler frequency compare is selected by three bits in the prescaler control register 2 (bits 0 through 2), as shown in Table 4. The frequency compare can be selected in seven ways (÷1, ÷2, ÷4, ÷8, ÷16, ÷32, ÷64 and ÷128). After reset, ÷1 (bit 0 = bit 1 = bit 2 = 0) is set.

When writing into prescaler control register 2, or when writing into time counter 2, prescaler is initialized to $FF.

Table 3 Clock Input Source Selection

PCR2		Clock Input Source
Bit 5	Bit 4	
0	0	Internal Clock ϕ_2
0	1	Timer Pin Control ϕ_2
1	0	
1	1	Timer Pin Input

Table 4 Prescaler Frequency Comparator Selection

PCR2			Prescaler Frequency Comparator
Bit 2	Bit 1	Bit 0	
0	0	0	÷ 1
0	0	1	÷ 2
0	1	0	÷ 4
0	1	1	÷ 8
1	0	0	÷ 16
1	0	1	÷ 32
1	1	0	÷ 64
1	1	1	÷ 128

■ SELF CHECK

The self check capability of the MCU provides an internal check to determine if the part is functional. Connect the MCU as shown in Figure 8 and monitor the output of port C bit 3 for and oscillation of approximately three hertz, if the LSI is normal.

■ RESETS

The MCU can be reset by the external reset input ($\overline{RES}$) as shown in Figure 9. The reset conditions are the same as all of the input ports of the MCU (the contents of DDR are cleared).

Upon power up, reset input requires a minimum of 100 milliseconds to stabilize the internal oscillator, then it is necessary to go "Low". As shown in Figure 10, a sufficient delay occurs by connecting a capacitor to the $\overline{RES}$ input.

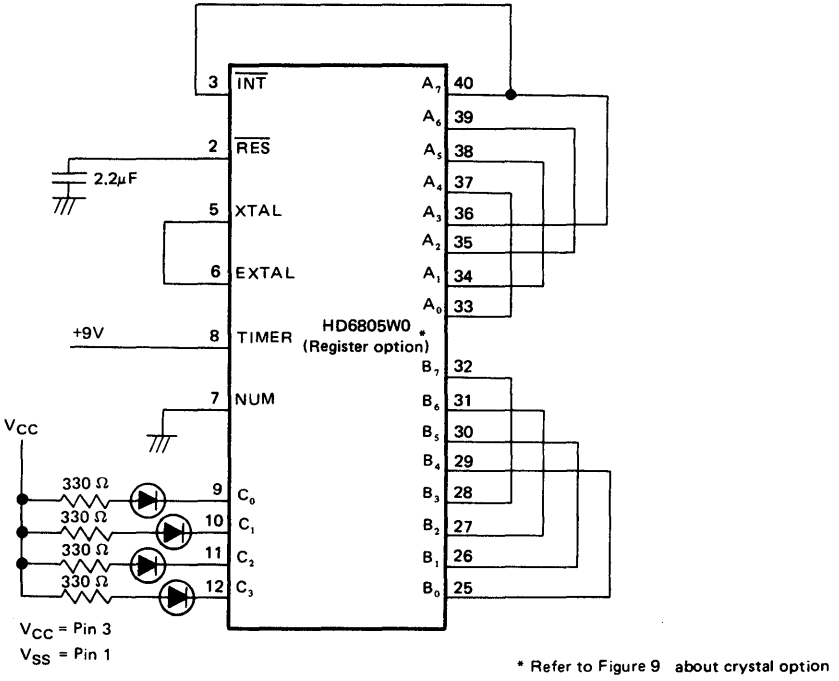


Figure 8 Self Check Connections

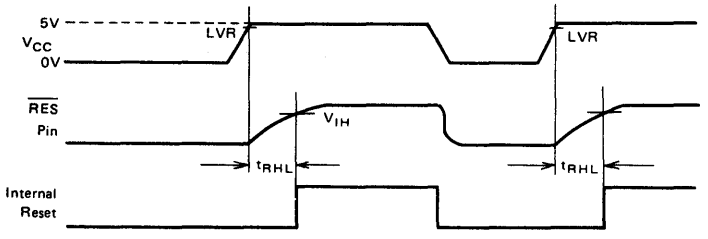


Figure 9 Power Up and Reset Timing

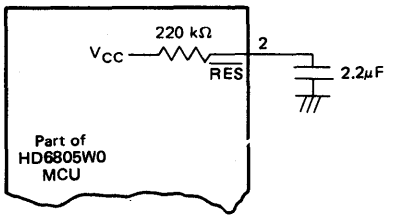


Figure 10 Power Up Reset Delay Circuit

■ INTERNAL OSCILLATOR OPTIONS

The internal oscillator circuit has been designed to require a minimum of external components. The use of a crystal (AT cut, 4 MHz max) or a resistor is sufficient to drive the internal oscillator with varying degrees of stability. Furthermore, a mask option manufactured separately from the LSI is available to provide better matching between the external components and the internal oscillator. Four different connection methods are shown in Figure 11. Crystal specifications are given in Figure 12. A resistor selection graph is shown in Figure 13.

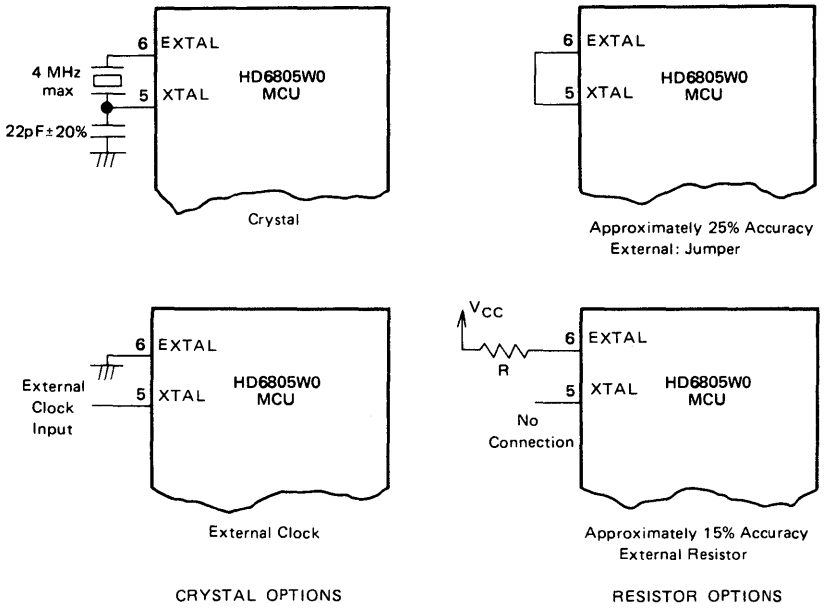
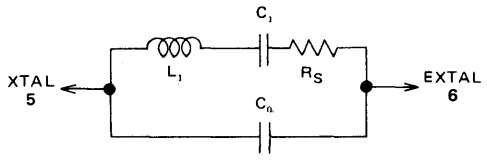


Figure 11 Internal Oscillator Options



AT – Cut Parallel Resonance Crystal
C_0 = 7 pF max.
f = 4 MHz (C_1 =22pF±20%)
R_S = 60 Ω max.

Figure 12 Crystal Parameters

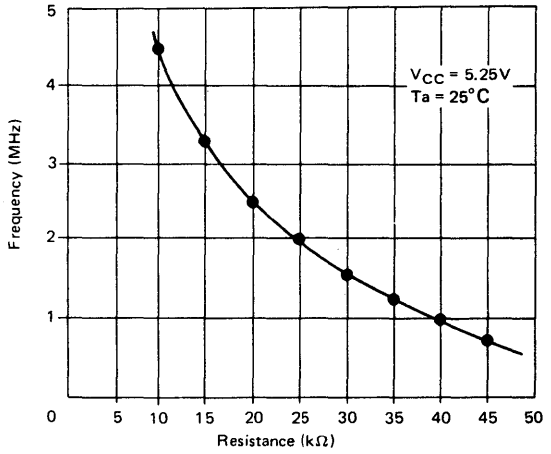


Figure 13 Typical Resistor Selection Graph

■ INTERRUPTS

The CPU can be interrupted in seven different ways: through external interrupt ($\overline{INT_1}$ and $\overline{INT_2}$), internal timer interrupt request (Timer 1, ICI, OCI and OFI) or on command (SWI). Among these, $\overline{INT_2}$ and Timer 1 are generated by the same vector address. When interrupt occurs, processing of the program is suspended, the present CPU state is pushed onto the stack. Moreover, the interrupt mask bit (I) of condition code register is set and the external routine priority address is achieved from the special external vector address. After that, the external interrupt routine is executed. The interrupt service routines normally end with a return from interrupt (RTI) instruction which allows the CPU to resume processing of the program prior to the interrupt. The priority interrupts are shown in Table 5 with the vector address that contains the starting address of the appropriate interrupt routine. The interrupt sequence is shown as a flowchart in Figure 14.

Table 5 Interrupt Priorities

Interrupt	Priority	Vector Address
$\overline{RES}$	1	$FFE, $FFF
SWI	2	$FFC, $FFD
$\overline{INT_1}$	3	$FFA, $FFB
Timer/$\overline{INT_2}$	4	$FF8, $FF9
ICI	5	$FF6, $FF7
OCI	6	$FF4, $FF5
OFI	7	$FF2, $FF3

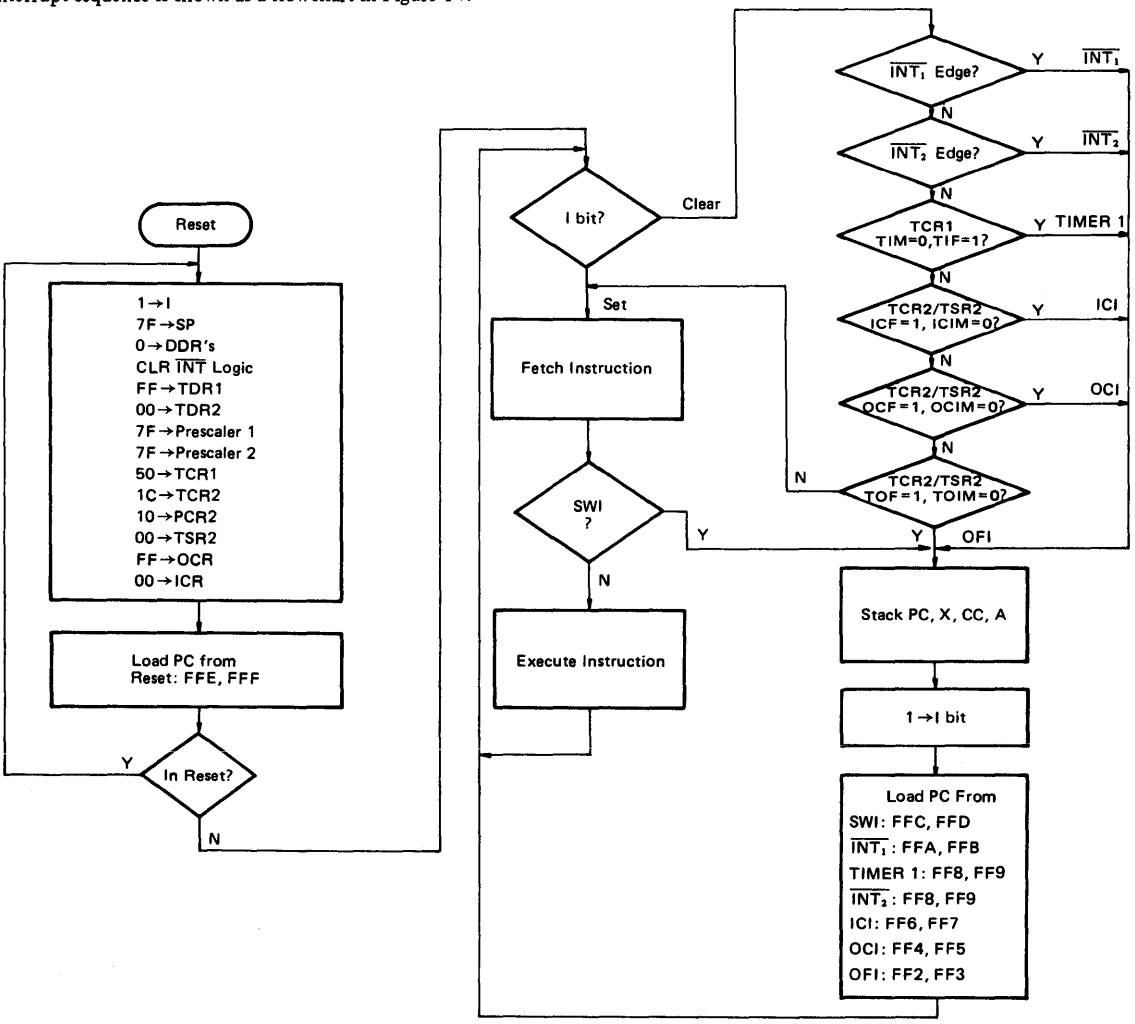
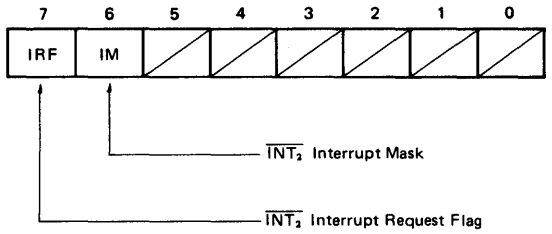


Figure 14 Interrupt Flowchart

- **Miscellaneous Register (MR: $00A)**

As shown in Table 5, the interrupt vector address of external interrupt vector $\overline{INT_2}$ is the same as Timer 1 interrupt vector address. For that reason, the control register for $\overline{INT_2}$ is the miscellaneous register (MR: $00A).

Miscellaneous Register (MR: $00A)



The interrupt of $\overline{INT_2}$ is caused by the negative edge to enable latch. Moreover, interrupt inhibit is the same as the (I) bit for other condition code register (CC).

Miscellaneous register (MR) bit 7 is the $\overline{INT_2}$ interrupt request flag. When interrupt occurs, bit 7 is set at "1". Bit 7 is checked by software in the vector address ($FF8, $FF9) interrupt routine and indicated by interrupt of $\overline{INT_2}$. Bit 7 is reset by software (BCLR instruction).

Bit 6 is the interrupt mask bit for $\overline{INT_2}$. When bit 6 is "1" $\overline{INT_2}$ interrupt is inhibited.

While read-modify-write instruction is processing IRF, if $\overline{INT_2}$ interrupt occurs, $\overline{INT_2}$ interrupt request can be received. Accordingly, miscellaneous register set/reset should not be used for read-modify-write instructions (COM, ROL, ROR, LSL, LSR and ASL). The details of these instructions are shown in Table 8.

Since IRF can read/write, IRF is written with software and "1" should not be written in. Accordingly, interrupt request should not be written in with software.

■ INPUT/OUTPUT

There are 23 input/output pins. All of the pins are controlled by the data direction register and both input and output are programmable. When output is programmed, the I/O port is read, for example, even if the output load of output level changes, the latched logical level data is read as shown in Figure 15. When Port B is output by program, the current from each pin is capable is sinking 10 mA (V_{OL} max. = 1V). Furthermore, port A pin is CMOS compatible at output. Ports B and C are CMOS compatible at input. Several examples of the Port distributions are shown in Figure 16.

On port C, C_5 and C_6 are timer 2 and are selected by software.

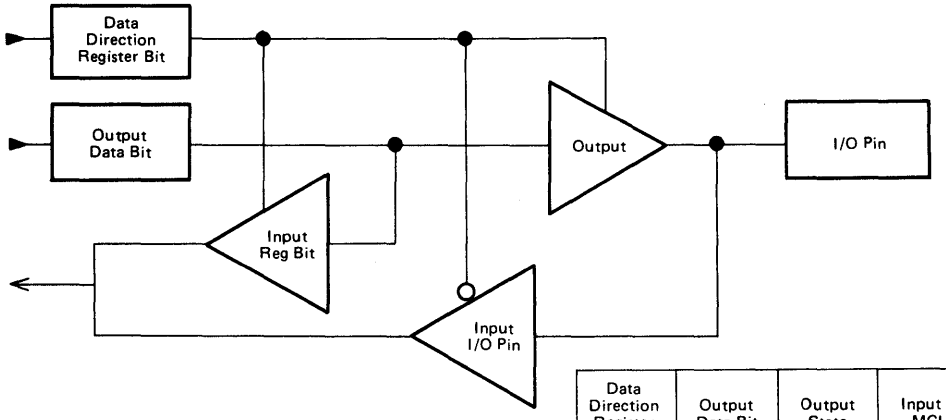
When all of port C is used for input, all of port C DDR ($006) should be "0". When port C is used for output, all of port C DDR should be "1". At this time the data input uses port C ($002).

When port C is used as timer 2 input capture (IC), port C bit 5 is reset to "0" (input) and timer control register 2 (TCR2: $01B) bit 4 (ICIM) is reset to "0".

For either edge, the value in TCR2 bit 1 (IEDG) is input to input capture register as timer data register 2 value.

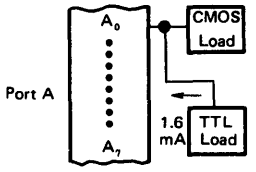
When timer 2 output compare (OC) is used, port C, DDR ($006) bit 6 is set to "1" (output). In this case, timer data register 2 matches the content of output compare register and the value of bit 0 of timer control register 2 is output to C_6.

In this case, data can also be written to C_6 by software. When the output compare match is written with software, the software write has priority and the output compare write is ignored. Moreover, when C_6 is output-only, output matched output is written at C_6 and caution must be taken.

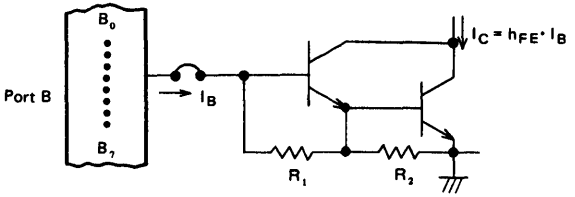


Data Direction Register Bit	Output Data Bit	Output State	Input to MCU
1	0	0	0
1	1	1	1
0	x	3-State	Pin

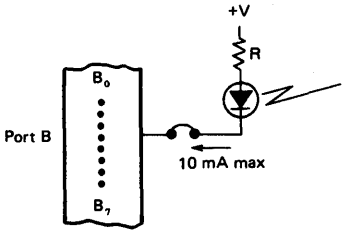
Figure 15 Typical Port I/O Circuitry



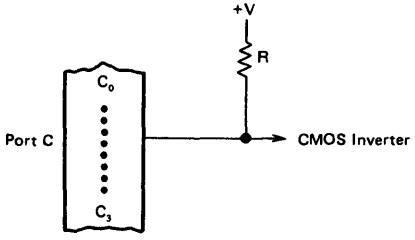
Port A Programmed as output(s) driving CMOS and TTL Load directly.
(a)



Port B Programmed as output(s) driving Darlington base directly.
(b)



Port B Programmed as output(s) driving LED(s) directly.
(c)



Port C Programmed as output(s) driving CMOS using external pull-up resistors.
(d)

Figure 16 Typical Port Connections

■ INPUT

Port D has 6 TTL compare compatible pins and 4-channel input A/D converter can be used. The conceptual structure of port D is shown in Figure 17.

When $003 is read in the input is 6 pin TTL compatible. For use as an A/D converter, refer to the section on A/D converters.

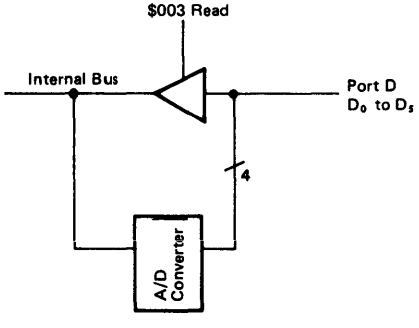


Figure 17 Port D

■ A/D CONVERTER

The HD6805W0 has an internal 8 bit A/D converter. The A/D converter is shown in Figure 18 and has 4 pins for analog input (AN_0 through AN_3), result register and control/status register (ADCSR).

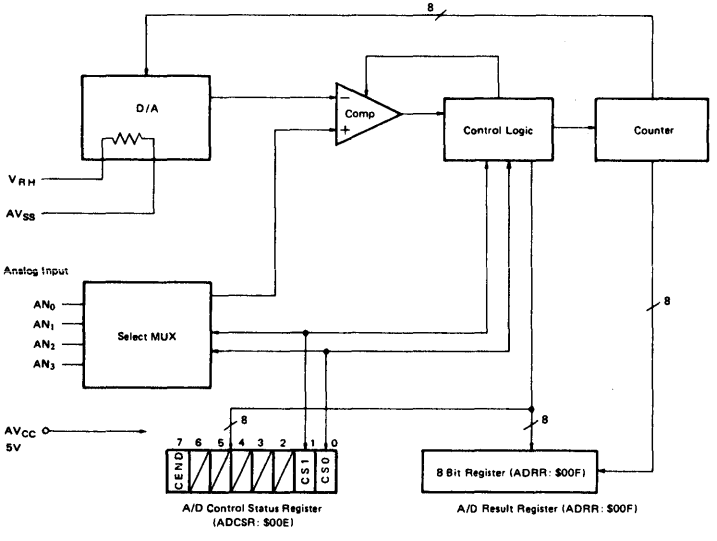


Figure 18 A/D Converter Block Diagram

- **Analog Input (AN_0 through AN_3)**

Analog pins AN_0 through AN_3 accept analog voltages of 0V through 5V. The resolution power is 8 bit (256 divisions) with a conversion time of 76 μs at 1 MHz. Analog conversion is selected by bits 0 through of control status register (ADCSR) analog input. Since the CPU is unnecessary for conversion, other user programs can be executed.

Table 6 Analog Input Selection

ADCSR		Analog Input Signal
Bit 1	Bit 0	
0	0	AN_0
0	1	AN_1
1	0	AN_2
1	1	AN_3

- **A/D Control Status Register (ADCSR: $00E)**

Control status register ($00E) is used to select analog input or A/D conversion termination.

Analog input selection is as shown in Table 6 and is selected by bits 0 through 1 of bit 2. The analog input signal is selected from among AN_0 through AN_3.

A/D conversion begins when data is writen into bits 0 through 1 of control status register. When A/D conversion ends, bit 7 (CEND flag) is set. Bit 7 is reset after the A/D result register is read. Moreover, when bit 7 is set A/D conversion execution continues. To end the A/D conversion, A/D result register is set to the most recent value. During A/D conversion execution, data is written into the new A/D control status register which selects the input channel and A/D conversion execution at that time is suspended. CEND flag is set and new A/D conversion begins.

- **A/D Result Register (ADRR: $00F)**

When A/D conversion ends, the result is set in the A/D conversion result register ($00F). When the ADCSR CEND flag is set, converted result is obtained from A/D result register. Furthermore, ADCSR CEND flag is reset.

- **STANDBY RAM**

The 8 bit RAMs $020 through $027 are used for standby RAM. When used as standby RAM, V_{CC} standby is impressed after V_{CC} OFF. Consequently, power supply is connected only to standby RAM and other parts are not connected with power supply. Accordingly, there is a small savings in power supply but the standby RAM data is maintained.

Standby RAM control is performed by RAM control register or RAME signal.

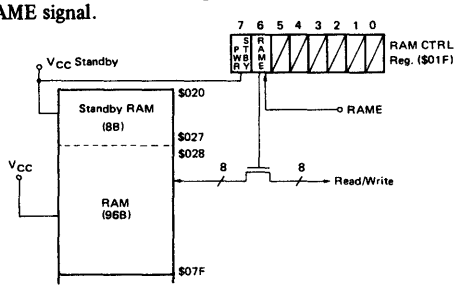


Figure 19 Standby RAM

- **RAM Control Register (RCR: $01F)**

This register, which is addressed at $01F, gives status information about the standby RAM. A "0" in the RAM enable bit (RAME) will disable the standby RAM, thereby protecting it at power down in V_{CC} standby is held greater than V_{SBB} volts as explained previously in the signal description for V_{CC} standby.

RAM Control Register

	7	6	5	4	3	2	1	0
$01F	STBY PWR	RAME	X	X	X	X	X	X

Bit 0 Not Used
Bit 1 Not Used
Bit 2 Not Used
Bit 3 Not Used
Bit 4 Not Used
Bit 5 Not Used

Bit 6 RAM Enable

The RAM enable control bit allows the user to disable the standby RAM. The bit which resets the CPU is set to "1" and standby RAM is enabled. This bit can be written to "1" or "0" under program control.

When the RAM is disabled, (logic "0") the RAM address is ineffective.

Bit 7 Standby bit

The standby bit is cleared when the standby voltage is removed. This bit is a read/write status flag that the user can read which indicates that the standby RAM voltage has been applied, and the data in the standby RAM is valid.

- **RAME Signal**

The RAME signal can control the RAME bit in the RAM control register to be "0" (RAME is disabled). When in a hardware standby mode, V_{CC} standby remains on and the RAME signal falls, then V_{CC} is set to off, as shown in Figure 20.

With V_{CC} on, RAME as "0" causes the signal to fall and RCR RAME bit is reset at "0", standby RAM is disabled. After this, V_{CC} becomes off and RAME is maintained by the V_{CC} standby power source. Moreover, RAME rises to "1" after V_{CC} on and RCR RAME bit is "1". Standby RAM is then enabled. In this way, without software, RAME signal is used externally to control the RAM.

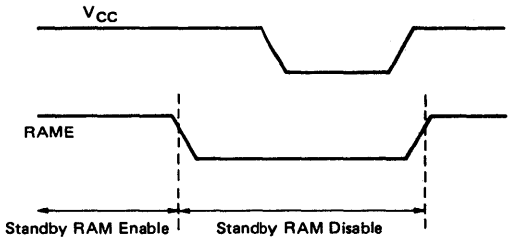


Figure 20 RAM Control Signal (RAME)

■ BIT MANIPULATION

The MCU has the ability to set or clear any single RAM or input/output port (except the data direction registers) with a single instruction (BSET and BCLR). Any bit in the page zero read only memory can be tested by using the BRSET and BRCLR instructions, and the program branches as a result of its state. This capability to work with any bit in RAM, ROM or I/O allows the user to have individual flags in RAM or to handle single I/O bits as control lines. The example in Figure 21 shows the usefulness of the bit manipulation and test instructions. Assume that bit 0 of port A is connected to a zero crossing detector circuit and that bit 1 of port A is connected to the trigger of a TRIAC which powers the controlled hardware.

This program, which uses only seven bytes of ROM provides turn-on of the TRIAC within 14 microseconds of the zero crossing. The timer is also incorporated to provide turn-on at some later time which permits pulse-width modulation of the controlled power.

```
SELF 1    BRCLR0, PORTA, SELF 1
          BSET 1, PORTA
          BCLR 1, PORTA
```

Figure 21 Bit Manipulation Example

■ ADDRESSING MODES

The MCU has ten addressing modes available for use by the programmer. These modes are explained and illustrated briefly in the following paragraphs.

• Immediate

Refer to Figure 22. The immediate addressing mode accesses constants which do not change during program execution. Such instructions are two bytes long. The effective address (EA) is the PC and the operand is fetched from the byte following the opcode.

• Direct

Refer to Figure 23. In direct addressing, the address of the operand is contained in the secondbyte of the instruction. Direct addressing allows the user to directly address the lowest 256 bytes in memory. All RAM space, I/O registers and 128 bytes of ROM are located in page zero to take advantage of this efficient memory addressing mode.

• Extended

Refer to Figure 24. Extended addressing is used to reference any location in memory space. The EA is the contents of the two bytes following the opcode. Extended addressing instructions are three bytes long.

• Relative

Refer to Figure 25. The relative addressing mode applies only to the branch instructions. In this mode the contents of the byte following the opcode is added to the program counter when the branch is taken. EA = (PC) + 2 + Rel. Rel is the contents of the location following the instruction opcode with bit 7 being the sign bit. If the branch is not taken, Rel = 0, when a branch takes place, the program goes to somewhere within the range of +129 bytes to -127 bytes of the present instruction. These instructions are two bytes long.

• Indexed (No Offset)

Refer to Figure 26. This mode of addressing accesses the lowest 256 bytes of memory. These instructions are one byte long and their EA is the contents of the index register.

• Indexed (8-bit Offset)

Refer to Figure 27. The EA is calculated by adding the contents of the byte following the opcode to the contents of the index register. In this mode, 511 low memory locations are accessable. These instructions occupy two bytes.

• Indexed (16-bit Offset)

Refer to Figure 28. This addressing mode calculates the EA by adding the contents of the two bytes following the opcode to the index register. Thus, the entire memory space may be accessed. Instructions which use this addressing mode are three bytes long.

• Bit Set/Clear

Refer to Figure 29. This mode of addressing applies to instructions which can set or clear any bit on page zero. The lower three bits in the opcode specify the bit to be set or cleared while the byte following the opcode specifies the address in page zero.

• Bit Test and Branch

Refer to Figure 30. This mode of addressing applies to instructions which can test any bit in the first 256 locations ($00 through $FF) and branch to any location relative to the PC. The byte to be tested is addressed by the byte following the opcode. The individual bit within that byte to be tested is addressed by the lower three bits of the opcode. The third byte is the relative address to be added to the program counter if the branch condition is met. These instructions are three bytes long. The value of the bit to be tested is written to the carry bit in the condition code register.

• Implied

Refer to Figure 31. The implied mode of addressing has no EA. All of the information necessary to execute an instruction is contained in the opcode. Direct operations on the accumulator and the index register are included in this mode of addressing. In addition, control instructions such as SWI and RTI belong to this group. All implied addressing instructions are one byte long.

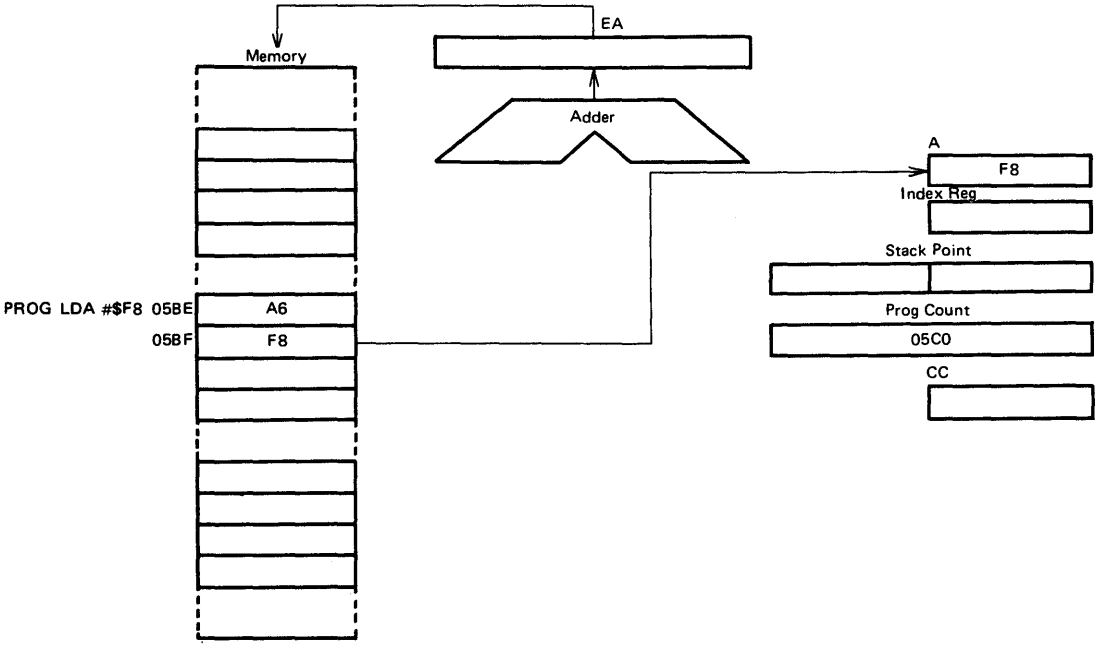


Figure 22 Immediate Addressing Example

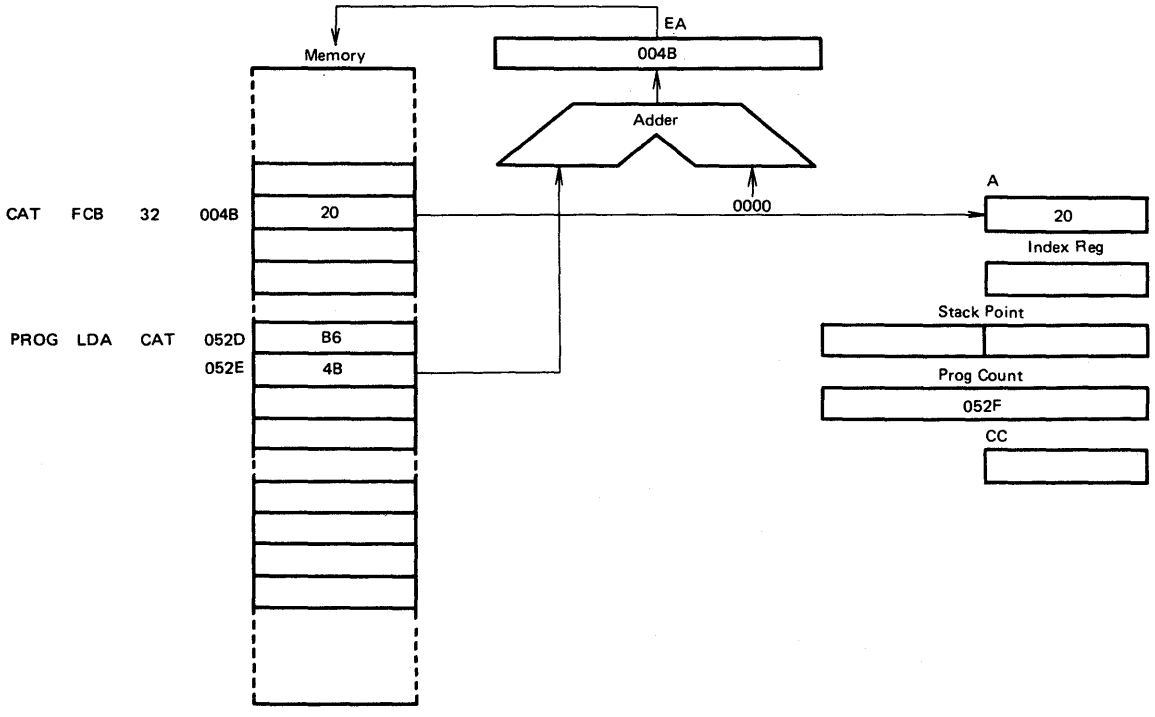


Figure 23 Direct Addressing Example

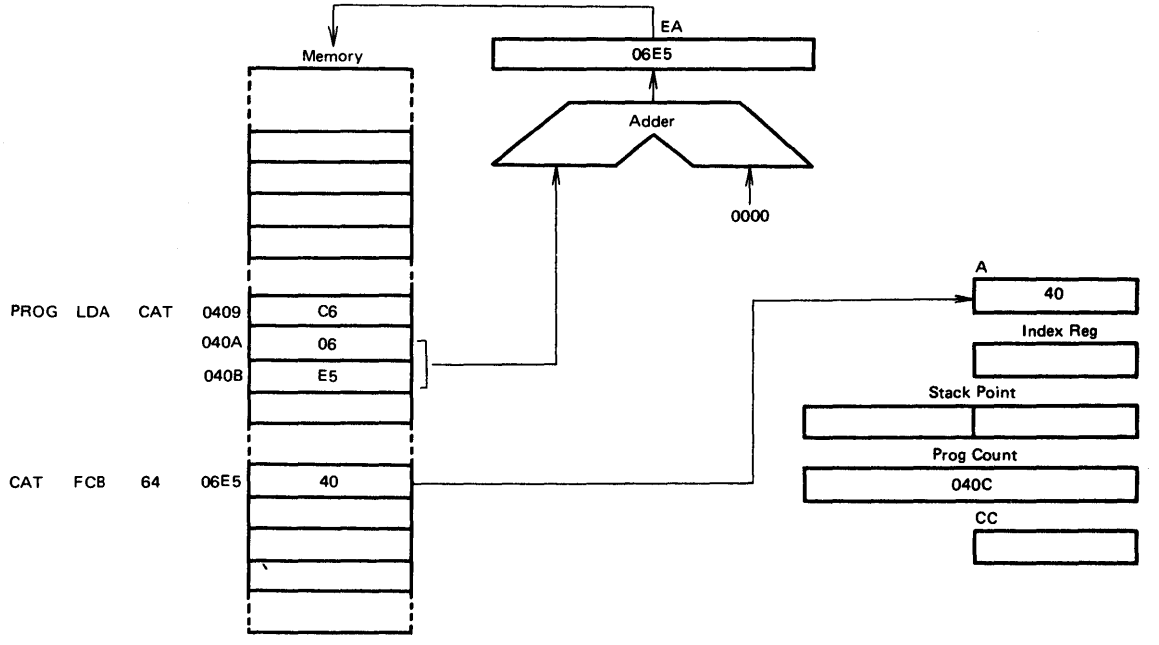


Figure 24 Extended Addressing Example

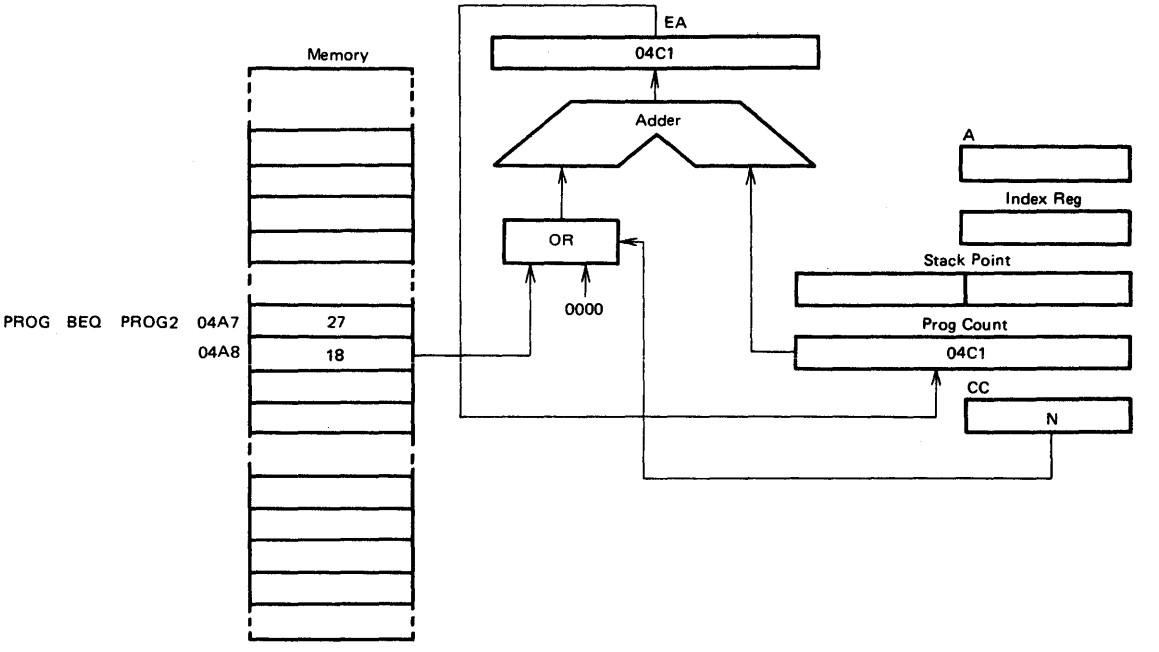


Figure 25 Relative Addressing Example

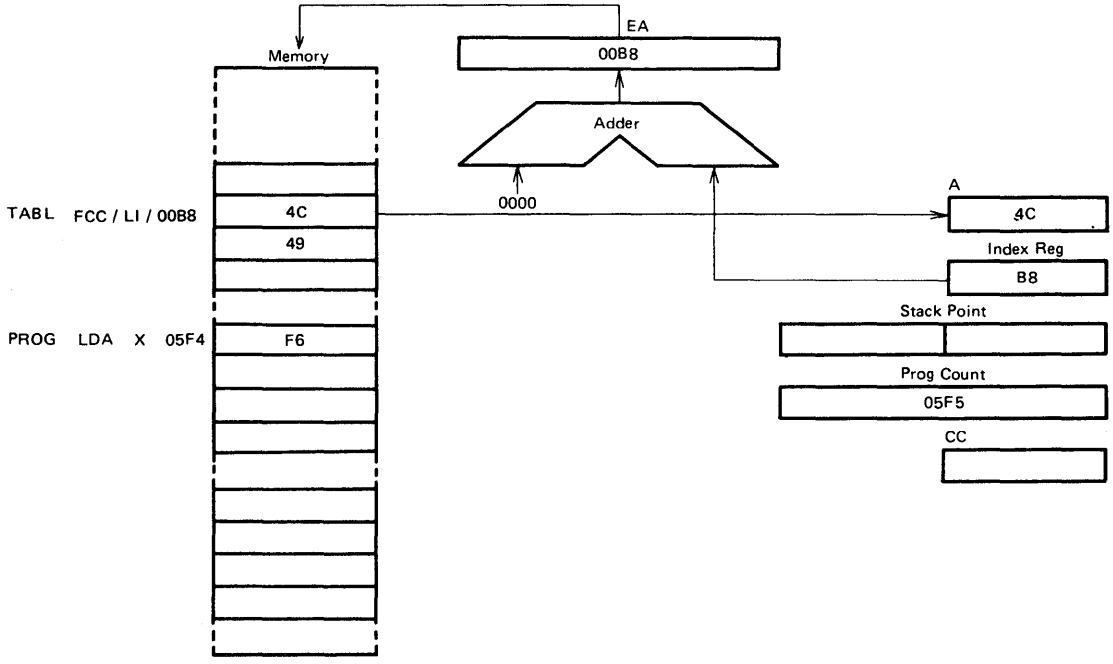


Figure 26 Indexed (No Offset) Addressing Example

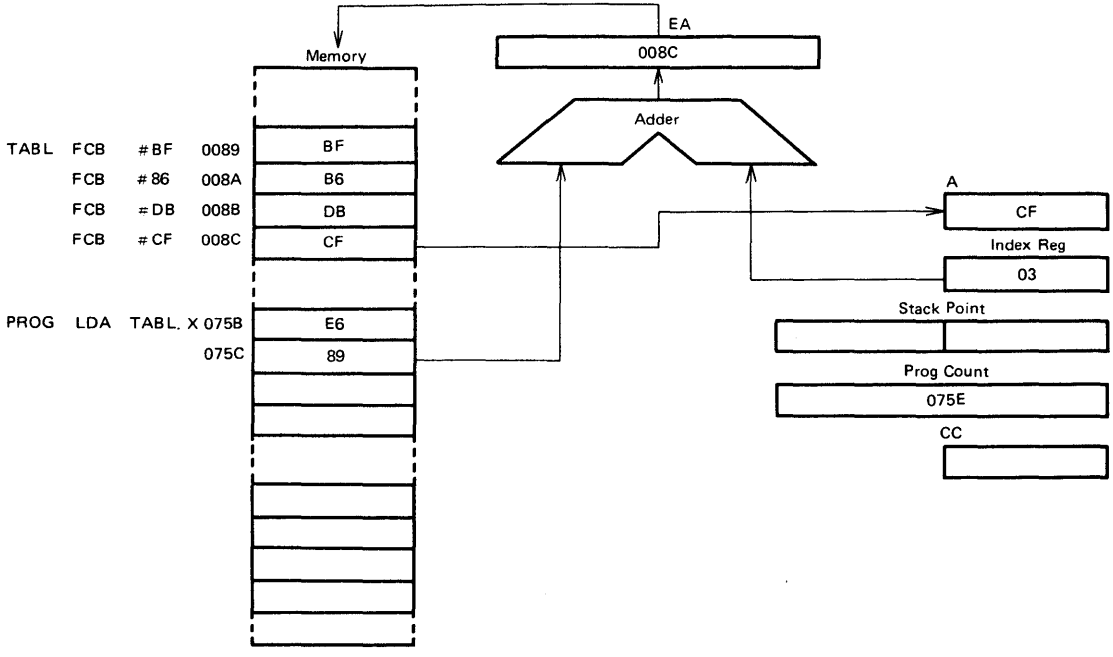


Figure 27 Indexed (8-Bit Offset) Addressing Example

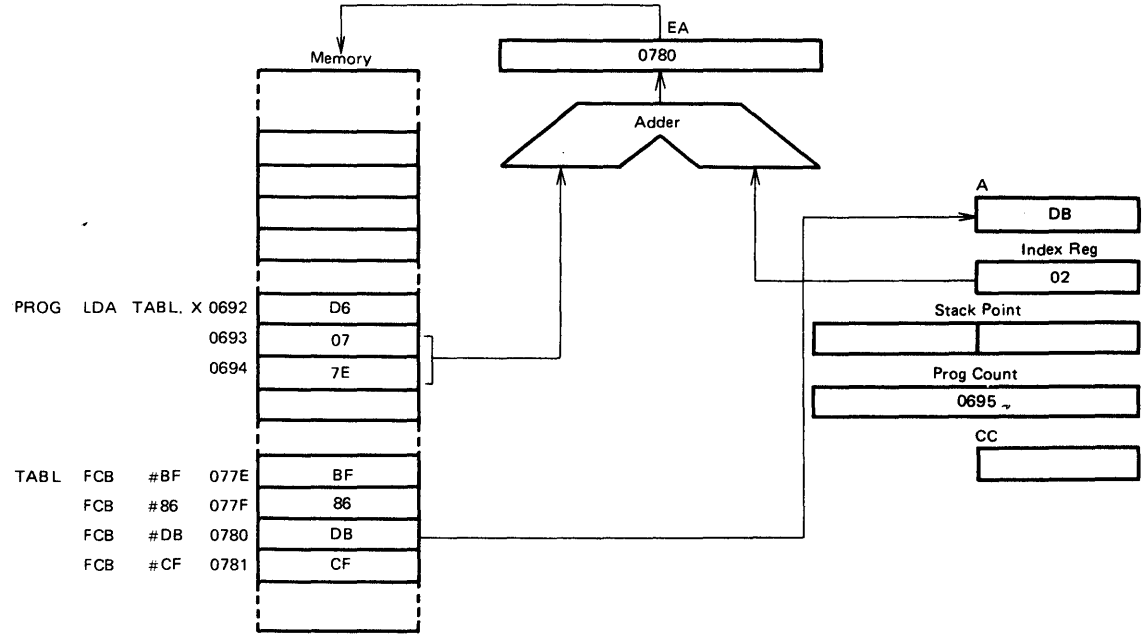


Figure 28 Indexed (16-Bit Offset) Addressing Example

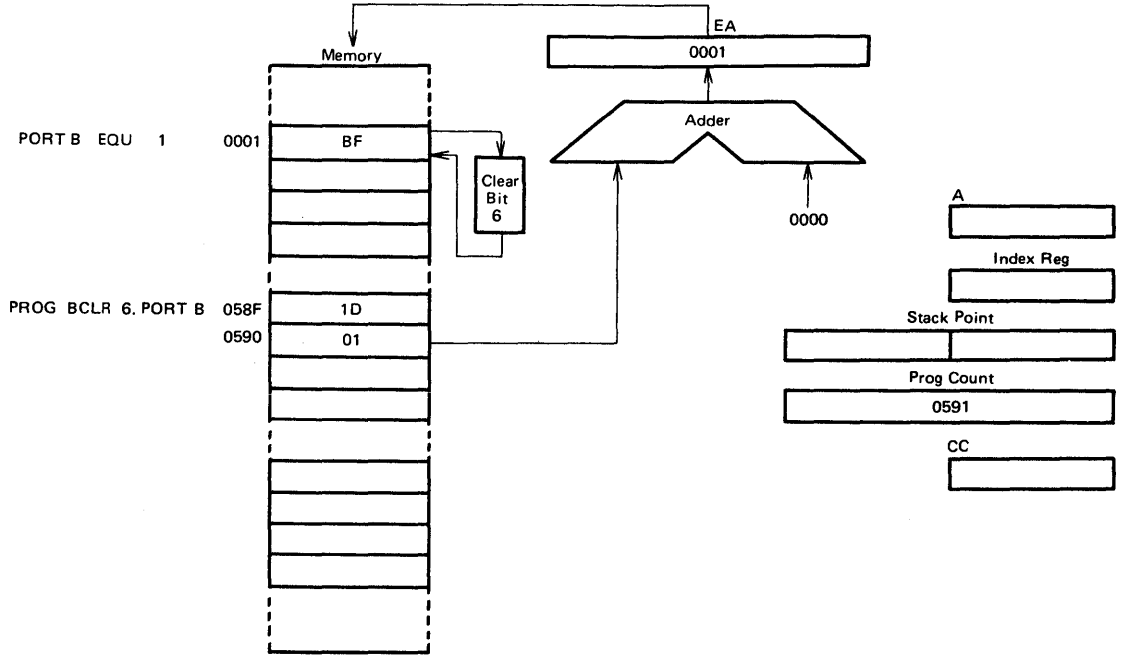


Figure 29 Bit Set/Clear Addressing Example

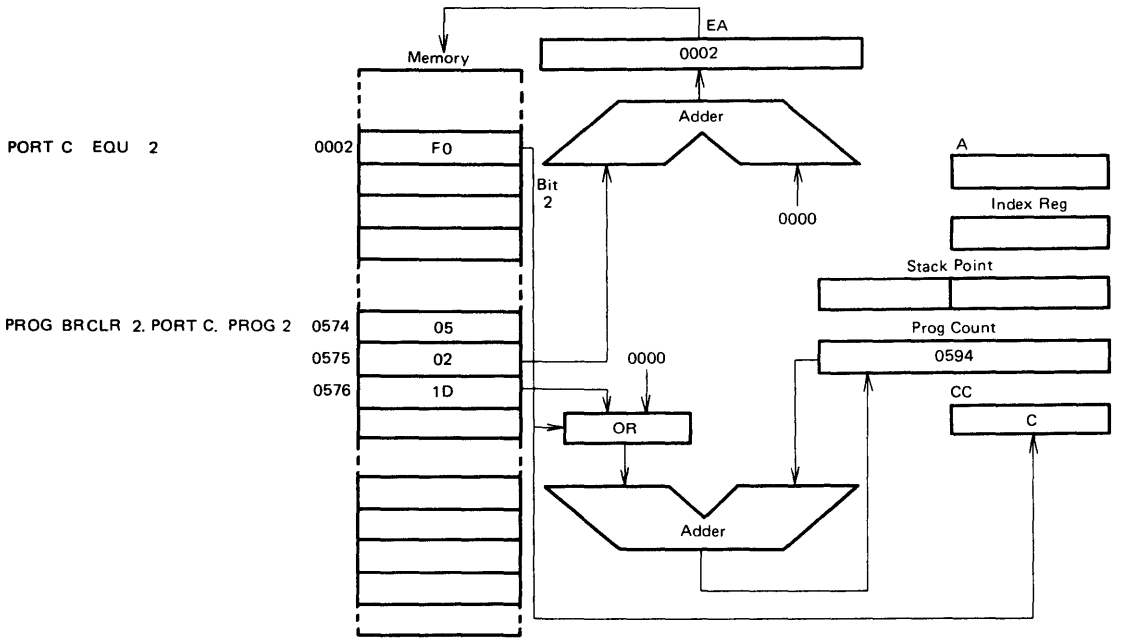


Figure 30 Bit Test and Branch Addressing Example

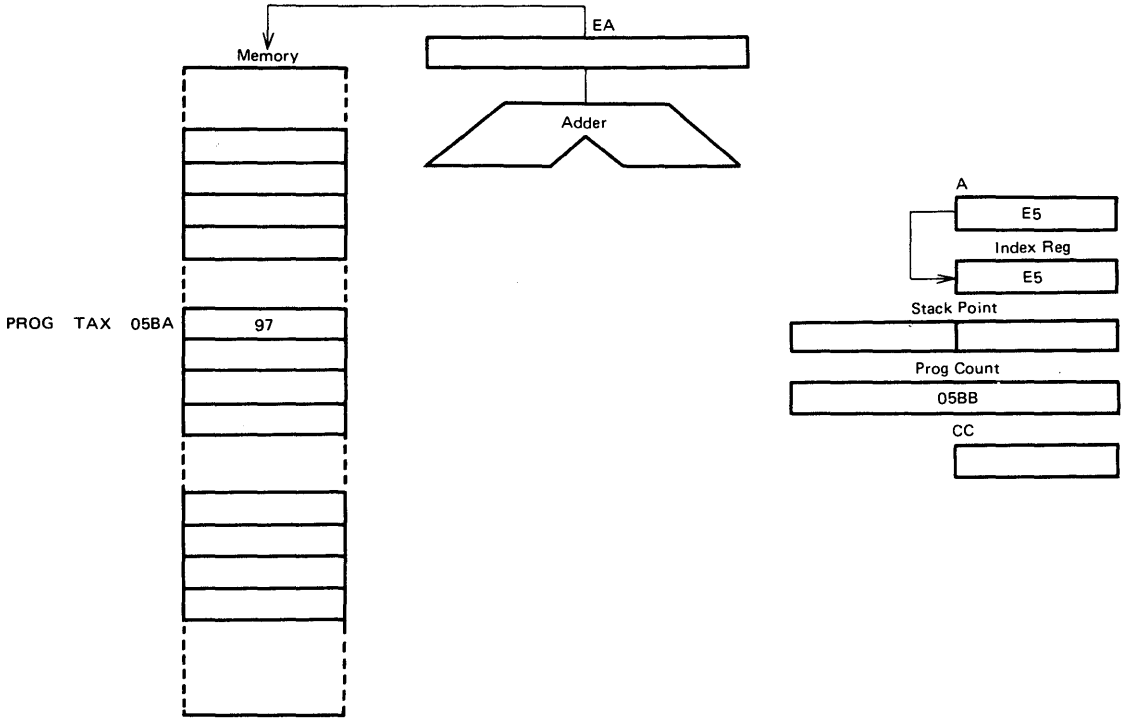


Figure 31 Implied Addressing Example

■ INSTRUCTION SET

The MCU has a set of 59 basic instructions. These instructions can be divided into five different types; register/memory, read/modify/write, branch, bit manipulation and control. Each instruction is breifly explained below. All of the instructions within a given type are presented in individual tables.

• Register/Memory Instructions

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory by using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to Table 7.

• Read/Modify/Write Instructions

These instructions read a memory location or a register, modify or test its contents and write the modified value back to the memory or register. The TST instruction for test of negative or zero is an exception to the read/modify/write instructions since it does not perform the write. Refer to Table 8.

• Branch Instructions

The branch instructional cause a branch from a program when a certain condition is met. Refer to Table 9.

• Bit Manipulation Instructions

These instructions are used on any bit in the first 256 bytes of the memory. One group either sets or clears. The other group performs the bit test and branch operations. Refer to Table 10.

• Control Instructions

The control instructions control the MCU operations during program execution. Refer to Table 11.

• Alphabetical Listing

The complete instruction set is given in alphabetical order in Table 12.

• Opcode Map

Table 13 is an opcode map for the instructions used on the MCU.

Table 7 Register/Memory Instructions

Function	Mnemonic	Addressing Modes																	
		Immediate			Direct			Extended			Indexed (No Offset)			Indexed (8-Bit Offset)			Indexed (16-Bit Offset)		
		Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Load A from Memory	LDA	A6	2	2	B6	2	4	C6	3	5	F6	1	4	E6	2	5	D6	3	6
Load X from Memory	LDX	AE	2	2	BE	2	4	CE	3	5	FE	1	4	EE	2	5	DE	3	6
Store A in Memory	STA	–	–	–	B7	2	5	C7	3	6	F7	1	5	E7	2	6	D7	3	7
Store X in Memory	STX	–	–	–	BF	2	5	CF	3	6	FF	1	5	EF	2	6	DF	3	7
Add Memory to A	ADD	AB	2	2	BB	2	4	CB	3	5	FB	1	4	EB	2	5	DB	3	6
Add Memory and Carry to A	ADC	A9	2	2	B9	2	4	C9	3	5	F9	1	4	E9	2	5	D9	3	6
Subtract Memory	SUB	A0	2	2	B0	2	4	C0	3	5	F0	1	4	E0	2	5	D0	3	6
Subtract Memory from A with Borrow	SBC	A2	2	2	B2	2	4	C2	3	5	F2	1	4	E2	2	5	D2	3	6
AND Memory to A	AND	A4	2	2	B4	2	4	C4	3	5	F4	1	4	E4	2	5	D4	3	6
OR Memory with A	ORA	AA	2	2	BA	2	4	CA	3	5	FA	1	4	EA	2	5	DA	3	6
Exclusive OR Memory with A	EOR	A8	2	2	B8	2	4	C8	3	5	F8	1	4	E8	2	5	D8	3	6
Arithmetic Compare A with Memory	CMP	A1	2	2	B1	2	4	C1	3	5	F1	1	4	E1	2	5	D1	3	6
Arithmetic Compare X with Memory	CPX	A3	2	2	B3	2	4	C3	3	5	F3	1	4	E3	2	5	D3	3	6
Bit Test Memory with A (Logical Compare)	BIT	A5	2	2	B5	2	4	C5	3	5	F5	1	4	E5	2	5	D5	3	6
Jump Unconditional	JMP	–	–	–	BC	2	3	CC	3	4	FC	1	3	EC	2	4	DC	3	5
Jump to Subroutine	JSR	–	–	–	BD	2	7	CD	3	8	FD	1	7	ED	2	8	DD	3	9

Symbols:
Op: Operation Abbreviation
: Instruction Statement

Table 8 Read/Modify/Write Instructions

Function	Mnemonic	Addressing Modes														
		Implied (A)			Implied (X)			Direct			Indexed (No Offset)			Indexed (8-Bit Offset)		
		Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Increment	INC	4C	1	4	5C	1	4	3C	2	6	7C	1	6	6C	2	7
Decrement	DEC	4A	1	4	5A	1	4	3A	2	6	7A	1	6	6A	2	7
Clear	CLR	4F	1	4	5F	1	4	3F	2	6	7F	1	6	6F	2	7
Complement	COM	43	1	4	53	1	4	33	2	6	73	1	6	63	2	7
Negate (2's Complement)	NEG	40	1	4	50	1	4	30	2	6	70	1	6	60	2	7
Rotate Left Thru Carry	ROL	49	1	4	59	1	4	39	2	6	79	1	6	69	2	7
Rotate Right Thru Carry	ROR	46	1	4	56	1	4	36	2	6	76	1	6	66	2	7
Logical Shift Left	LSL	48	1	4	58	1	4	38	2	6	78	1	6	68	2	7
Logical Shift Right	LSR	44	1	4	54	1	4	34	2	6	74	1	6	64	2	7
Arithmetic Shift Right	ASR	47	1	4	57	1	4	37	2	6	77	1	6	67	2	7
Arithmetic Shift Left	ASL	48	1	4	58	1	4	38	2	6	78	1	6	68	2	7
Test for Negative or Zero	TST	4D	1	4	5D	1	4	3D	2	6	7D	1	6	6D	2	7

Symbols:
Op: Operation Abbreviation
: Instruction Statement

Table 9 Branch Instructions

Function	Mnemonic	Relative Addressing Mode		
		Op Code	# Bytes	# Cycles
Branch Always	BRA	20	2	4
Branch Never	BRN	21	2	4
Branch IF Higher	BHI	22	2	4
Branch IF Lower or Same	BLS	23	2	4
Branch IF Carry Clear	BCC	24	2	4
(Branch IF Higher or Same)	(BHS)	24	2	4
Branch IF Carry Set	BCS	25	2	4
(Branch IF Lower)	(BLO)	25	2	4
Branch IF Not Equal	BNE	26	2	4
Branch IF Equal	BEQ	27	2	4
Branch IF Half Carry Clear	BHCC	28	2	4
Branch IF Half Carry Set	BHCS	29	2	4
Branch IF Plus	BPL	2A	2	4
Branch IF Minus	BMI	2B	2	4
Branch IF Interrupt Mask Bit is Clear	BMC	2C	2	4
Branch IF Interrupt Mask Bit is Set	BMS	2D	2	4
Branch IF Interrupt Line is Low	BIL	2E	2	4
Branch IF Interrupt Line is High	BIH	2F	2	4
Branch to Subroutine	BSR	AD	2	8

Symbols: Op: Operation Abbreviation #: Instruction Statement

Table 10 Bit Manipulation Instructions

Function	Mnemonic	Addressing Modes					
		Bit Set/Clear			Bit Test and Branch		
		Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Branch IF Bit n is set	BRSET n (n=0 ..... 7)	–	–	–	2•n	3	10
Branch IF Bit n is clear	BRCLR n (n=0 .....7)	–	–	–	01+2•n	3	10
Set Bit n	BSET n (n=0 ..... 7)	10+2•n	2	7	–	–	–
Clear bit n	BCLR n (n=0 ..... 7)	11+2•n	2	7	–	–	–

Symbols: Op: Operation Abbreviation #: Instruction Statement

Table 11 Control Instructions

Function	Mnemonic	Implied		
		Op Code	# Bytes	# Cycles
Transfer A to X	TAX	97	1	2
Transfer X to A	TXA	9F	1	2
Set Carry Bit	SEC	99	1	2
Clear Carry Bit	CLC	98	1	2
Set Interrupt Mask Bit	SEI	9B	1	2
Clear Interrupt Mask Bit	CLI	9A	1	2
Software Interrupt	SWI	83	1	11
Return from Subroutine	RTS	81	1	6
Return from Interrupt	RTI	80	1	9
Reset Stack Pointer	RSP	9C	1	2
No-Operation	NOP	9D	1	2

Symbols: Op: Operation Abbreviation #: Instruction Statement

Table 12 Instruction Set

Mnemonic	Addressing Modes										Condition Code				
	Implied	Immediate	Direct	Extended	Relative	Indexed (No Offset)	Indexed (8 Bits)	Indexed (16 Bits)	Bit Set/ Clear	Bit Test & Branch	H	I	N	Z	C
ADC		x	x	x		x	x	x			∧	•	∧	∧	∧
ADD		x	x	x		x	x	x			∧	•	∧	∧	∧
AND		x	x	x		x	x	x			•	•	∧	∧	•
ASL	x		x			x	x				•	•	∧	∧	∧
ASR	x		x			x	x				•	•	∧	∧	∧
BCC					x						•	•	•	•	•
BCLR									x		•	•	•	•	•
BCS					x						•	•	•	•	•
BEQ					x						•	•	•	•	•
BHCC					x						•	•	•	•	•
BHCS					x						•	•	•	•	•
BHI					x						•	•	•	•	•
BHS					x						•	•	•	•	•
BIH					x						•	•	•	•	•
BIL					x						•	•	•	•	•
BIT		x	x	x		x	x	x			•	•	∧	∧	•
BLO					x						•	•	•	•	•
BLS					x						•	•	•	•	•
BMC					x						•	•	•	•	•
BMI					x						•	•	•	•	•
BMS					x						•	•	•	•	•
BNE					x						•	•	•	•	•
BPL					x						•	•	•	•	•
BRA					x						•	•	•	•	•
BRN					x						•	•	•	•	•
BRCLR										x	•	•	•	•	∧
BRSET										x	•	•	•	•	∧
BSET									x		•	•	•	•	•
BSR					x						•	•	•	•	•
CLC	x										•	•	•	•	0
CLI	x										•	0	•	•	•
CLR	x		x			x	x				•	•	0	1	•
CMP		x	x	x		x	x	x			•	•	∧	∧	∧
COM	x		x			x	x				•	•	∧	∧	1
CPX		x	x	x		x	x	x			•	•	∧	∧	∧
DEC	x		x			x	x				•	•	∧	∧	•
EOR		x	x	x		x	x	x			•	•	∧	∧	•
INC	x		x			x	x				•	•	∧	∧	•
JMP			x	x		x	x	x			•	•	•	•	•
JSR			x	x		x	x	x			•	•	•	•	•
LDA		x	x	x		x	x	x			•	•	∧	∧	•
LDX		x	x	x		x	x	x			•	•	∧	∧	•

Condition Code Symbols:
- H Half Carry (From Bit 3)
- I Interrupt Mask
- N Negative (Sign Bit)
- Z Zero
- C Carry Borrow
- ∧ Test and Set if True, Cleared Otherwise
- • Not Affected

(to be continued)

Table 12 Instruction Set

Mnemonic	Addressing Modes										Condition Code				
	Implied	Immediate	Direct	Extended	Relative	Indexed (No Offset)	Indexed (8 Bits)	Indexed (16 Bits)	Bit Set/Clear	Bit Test & Branch	H	I	N	Z	C
LSL	x		x			x	x				•	•	∧	∧	∧
LSR	x		x			x	x				•	•	0	∧	∧
NEQ	x		x			x	x				•	•	∧	∧	∧
NOP	x										•	•	•	•	•
ORA		x	x	x		x	x	x			•	•	∧	∧	•
ROL	x		x			x	x				•	•	∧	∧	∧
ROR	x		x			x	x				•	•	∧	∧	∧
RSP	x										•	•	•	•	•
RTI	x										?	?	?	?	?
RTS	x										•	•	•	•	•
SBC		x	x	x		x	x	x			•	•	∧	∧	∧
SEC	x										•	•	•	•	1
SEI	x										•	1	•	•	•
STA			x	x		x	x	x			•	•	∧	∧	•
STX			x	x		x	x	x			•	•	∧	∧	•
SUB		x	x	x		x	x	x			•	•	∧	∧	∧
SWI	x										•	1	•	•	•
TAX	x										•	•	•	•	•
TST	x		x			x	x				•	•	∧	∧	•
TXA	x										•	•	•	•	•

Condition Code Symbols:

H Half Carry (From Bit 3)
I Interrupt Mask
N Negative (Sign Bit)
Z Zero
C Carry/Borrow
∧ Test and Set if True, Cleared Otherwise
• Not Affected
? Load CC Register From Stack

Table 13 Opcode Map

	Bit Manipulation		Brnch	Read/Modify/Write					Control		Register/Memory						
	Test & Branch	Set/Clear	Rel	DIR	A	X	,X1	,X0	IMP	IMP	IMM	DIR	EXT	,X2	,X1	,X0	
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	←HIGH
0	BRSET0	BSET0	BRA	NEQ					RTI*	–	SUB						0
1	BRCLR0	BCLR0	BRN	–					RTS*	–	CMP						1
2	BRSET1	BSET1	BHI	–					–	–	SBC						2
3	BRCLR1	BCLR1	BLS	COM					SWI*	–	CPX						3 L
4	BRSET2	BSET2	BCC	LSR					–	–	AND						4 O
5	BRCLR2	BCLR2	BCS	–					–	–	BIT						5 W
6	BRSET3	BSET3	BNE	ROR					–	–	LDA						6
7	BRCLR3	BCLR3	BEQ	ASR					–	TAX	–	STA(+1)					7
8	BRSET4	BSET4	BHCC	LSL/ASL					–	CLC	EOR						8
9	BRCLR4	BCLR4	BHCS	ROL					–	SEC	ADC						9
A	BRSET5	BSET5	BPL	DEC					–	CLI	ORA						A
B	BRCLR5	BCLR5	BMI	–					–	SEI	ADD						B
C	BRSET6	BSET6	BMC	INC					–	RSP	–	JMP(-1)					C
D	BRCLR6	BCLR6	BMS	TST					–	NOP	BSR*	JSR(-3)					D
E	BRSET7	BSET7	BIL	–					–	–	LDX						E
F	BRCLR7	BCLR7	BIH	CLR					–	TXA	–	STX(+1)					F
	3/10	2/7	2/4	2/6	1/4	1/4	2/7	1/6	1/*	1/2	2/2	2/4	3/5	3/6	2/5	1/4	

[NOTE] 1. Undefined opcodes are marked with "–".
2. The number at the bottom of each column denote the number of bytes and the number of cycles required (Bytes/Cycles).
Mnemonics followed by a "*" require a different number of cycles as follows:

RTI	9
RTS	6
SWI	11
BSR	8

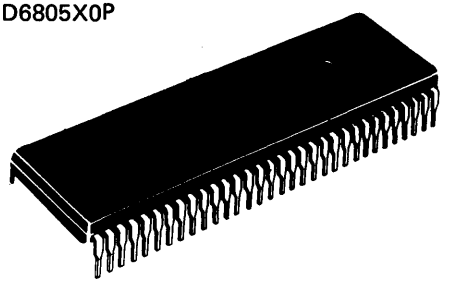
3. ( ) indicate that the number in parenthesis must be added to the cycle count for that instruction.

HD6805X0
MCU (Microcomputer Unit)

—ADVANCE INFORMATION—

The HD6805X0 is the 8-bit single chip Microcomputer Unit (MCU) which contains a CPU, on-chip clock, ROM, RAM, I/O and timer. The HD6805X0 is a member of HD6805 family and has more I/O ports and serial I/O than HD6805V1.

HD6805X0P



(DP-64)

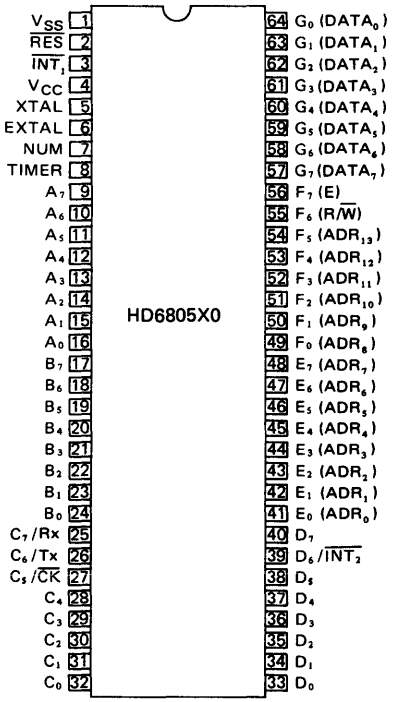
■ HARDWARE FEATURES

- 8-Bit Architecture
- 96 Bytes of RAM
- Memory Mapped I/O
- 4096 Bytes of ROM
- Internal 8-Bit Timer with 7-Bit Prescaler
- Vectored Interrupts – 2 External, Timer, Soft and Serial I/O
- 56 I/O Lines (16 Lines LED Compatible)
- On-Chip Serial I/O (Usable as Timer)
- On-Chip Clock Circuit
- Self-Check Mode
- Master Reset
- Low Voltage Inhibit
- Complete Development System Support by Evaluation Kit
- 5 Vdc Single Supply

■ SOFTWARE FEATURES

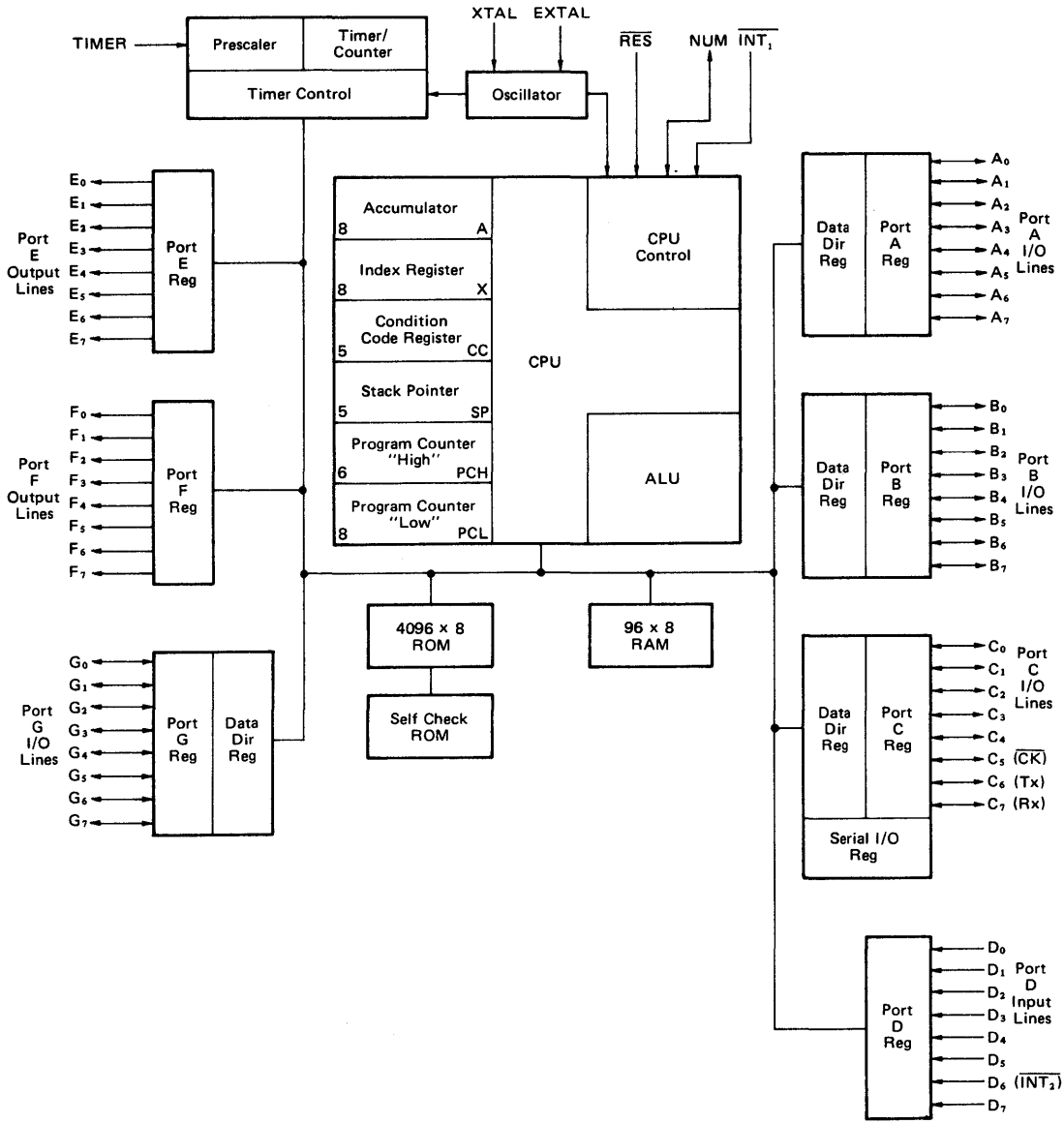
- Similar to HD6800
- Byte Efficient Instruction Set
- Easy to Program
- True Bit Manipulation
- Bit Test and Branch Instruction
- Versatile Interrupt Handing
- Powerful Indexed Addressing for Tables
- Full Set of Conditional Branches
- Memory Usable as Registers/Flags
- Single Instruction Memory Examine/Change
- 10 Powerful Addressing Mode
- All Addressing Modes Apply to ROM, RAM and I/O
- Compatible with MC6805P2
- Compatible with HD6805V1, HD6805U1

■ PIN ARRANGEMENT



(Top View)

■ BLOCK DIAGRAM

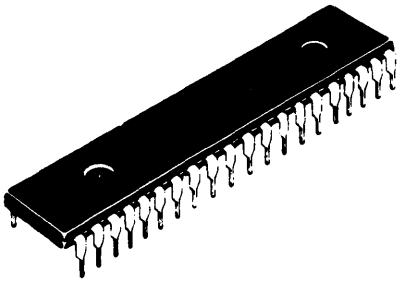


HD6301V0, HD63A01V0, HD63B01V0 CMOS MCU (Microcomputer Unit)

—PRELIMINARY—

The HD6301V0 is an 8-bit CMOS single-chip microcomputer unit, Object Code compatible with the HD6801. 4kB ROM, 128 bytes RAM, Serial Communication Interface (SCI), parallel I/O terminals as well as three functions of timer on chip are incorporated in the HD6301V0. It is bus compatible with HMCS6800, provided with some additional functions such as an improved execution time of key instruction plus several new instructions of operation to increase system throughput. The HD6301V0 can be expanded up to 65k words. Like the HMCS6800 family, I/O level is TTL compatible with +5.0V single power supply. By using the Hitachi's 3μm CMOS process, low power consumption is realized. And as lower power dissipation mode, HD6301V0 has Sleep Mode and Stand-By Mode. So flexible low power consumption application is possible.

HD6301V0P, HD63A01V0P, HD63B01V0P

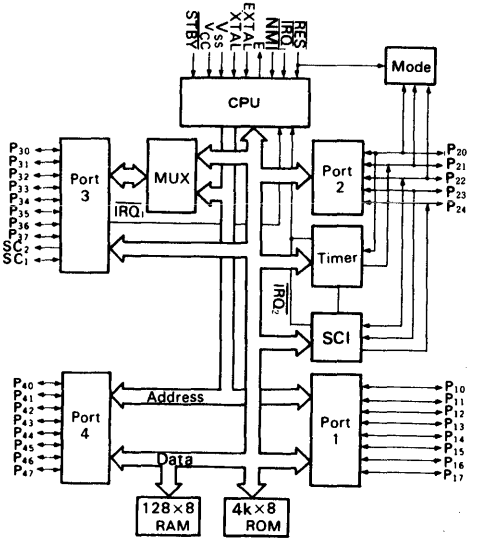


(DP-40)

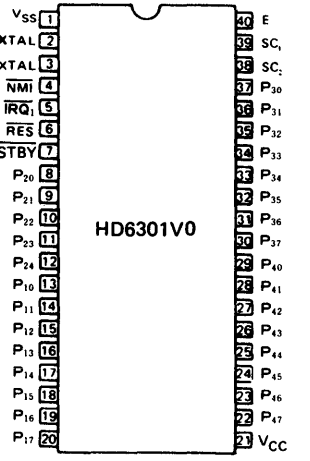
■ FEATURES

- Object Code Upward Compatible with HD6801 Family
- Abundant On-Chip Functions Compatible with HD6801V0; 4kB ROM,128 Bytes RAM,29 Parallel I/O Lines, 2 Lines of Data Strobe, 16-bit Timer, Serial Communication Interface
- Low Power Consumption Mode: Sleep Mode, Standby Mode
- Minimum Instruction Cycle Time
  1μs (f=1MHz), 0.67μs (f=1.5MHz), 0.5μs (f=2MHz)
- Bit Manipulation, Bit Test Instruction
- Protection from System Burst: Address Trap, Op-Code Trap
- Up to 65k Words Address Space
- Wide Operation Range
  V_{CC}=3 to 6V (f=0.5MHz), f=0.1 to 1.5MHz (V_{CC}=5V ±10%), f=0.1 to 2.0MHz (V_{CC}=5V ±5%)

■ BLOCK DIAGRAM



■ PIN ARRANGEMENT



(Top View)

■ TYPE OF PRODUCTS

Type No.	Bus Timing
HD6301V0	1 MHz
HD63A01V0	1.5 MHz
HD63B01V0	2 MHz

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC}	-0.3 ~ +7.0	V
Input Voltage	V_{in}	-0.3 ~ V_{CC}+0.3	V
Operating Temperature	T_{opr}	0 ~ +70	°C
Storage Temperature	T_{stg}	-55 ~ +150	°C

(NOTE) This product has protection circuits in input terminal from high static electricity voltage and high electric field. But be careful not to apply overvoltage more than maximum ratings to these high input impedance protection circuits. To assure the normal operation, we recommend V_{in}, V_{out} : $V_{SS} \leqq (V_{in}$ or $V_{out}) \leqq V_{CC}$.

■ ELECTRICAL CHARACTERISTICS

• DC CHARACTERISTICS (V_{CC} = 5.0V±10% --- f = 1.0~1.5MHz, V_{CC} = 5.0V±5% --- f = 2.0MHz, V_{SS} = 0V, Ta = 0~+70°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Input "High" Voltage	$\overline{RES}$, $\overline{STBY}$	V_{IH}		V_{CC}-0.5	–	V_{CC} +0.3	V
	EXTAL			V_{CC}x0.7	–		
	Other Inputs			2.0	–		
Input "Low" Voltage	All Inputs	V_{IL}		-0.3	–	0.8	V
Input Leakage Current	$\overline{NMI}$, $\overline{IRQ_1}$, $\overline{RES}$, $\overline{STBY}$	$\|I_{in}\|$	V_{in} = 0.4~5.1V	–	–	1.0	μA
Three State (off-state) Leakage Current	P_{10}~P_{17}, P_{20}~P_{24}, P_{30}~P_{37}, P_{40}~P_{47}, $\overline{IS3}$	$\|I_{TSI}\|$	V_{in} = 0.4~5.1V	–	–	1.0	μA
Output "High" Voltage	All Outputs	V_{OH}	I_{OH} = -200μA	2.4	–	–	V
			I_{OH} = -10μA	V_{CC}-0.7	–	–	V
Output "Low" Voltage	All Outputs	V_{OL}	I_{OL} = 1.6mA	–	–	0.55	V
Input Capacitance	All Inputs	C_{in}	V_{in}=0V, f=1.0MHz, Ta = 25°C	–	–	12.5	pF
Standby Current	Non Operation	I_{CC}		–	2.0	15.0	μA
Current Dissipation*		I_{CC}	Operating (f=1MHz**)	–	6.0	10.0	mA
			Sleeping (f=1MHz**)	–	1.0	2.0	
RAM Stand-By Voltage		V_{RAM}		2.0	–	–	V

* V_{IH} min = V_{CC}-1.0V, V_{IL} max = 0.8V

** Current Dissipation of the operating or sleeping condition is proportional to the operating frequency. So the typ. or max. values about Current Dissipations of the when of f = x MHz operation are decided according to the following formula;

typ. value (f = x MHz) = typ. value (f = 1MHz) x x
max. value (f = x MHz) = max. value (f = 1MHz) x x
(both the sleeping and operating)

- **AC CHARACTERISTICS** (V_{CC} = 5.0V±10% --- f = 1.0~1.5MHz, V_{CC} = 5.0V±5% --- f = 2.0MHz, V_{SS} = 0V, T_a = 0~+70°C, unless otherwise noted.)

BUS TIMING

Item		Symbol	Test Condition	HD6301V0			HD63A01V0			HD63B01V0			Unit
				min	typ	max	min	typ	max	min	typ	max	
Cycle Time		t_{cyc}		1	–	10	0.666	–	10	0.5	–	10	μs
Address Strobe Pulse Width "High"		PW_{ASH}		200	–	–	135	–	–	120	–	–	ns
Address Strobe Rise Time		t_{ASr}		–	–	35	–	–	35	–	–	35	ns
Address Strobe Fall Time		t_{ASf}		–	–	35	–	–	35	–	–	35	ns
Address Strobe Delay Time		t_{ASD}		40	–	–	TBD	–	–	TBD	–	–	ns
Enable Rise Time		t_{Er}		–	–	35	–	–	35	–	–	35	ns
Enable Fall Time		t_{Ef}		–	–	35	–	–	35	–	–	35	ns
Enable Pulse Width "High" Level		PW_{EH}		450	–	–	TBD	–	–	TBD	–	–	ns
Enable Pulse Width "Low" Level		PW_{EL}		450	–	–	TBD	–	–	TBD	–	–	ns
Address Strobe to Enable Delay Time		t_{ASED}		60	–	–	TBD	–	–	TBD	–	–	ns
Address Delay Time		t_{AD1}		–	–	250	–	–	TBD	–	–	TBD	ns
		t_{AD2}	Fig. 1	–	350	–	–	TBD	–	–	TBD	–	ns
Address Delay Time for Latch		t_{ADL}	Fig. 2	–	–	250	–	–	220	–	–	140	ns
Data Set-up Time	Write	t_{DSW}		230	–	–	TBD	–	–	TBD	–	–	ns
	Read	t_{DSR}		80	–	–	TBD	–	–	TBD	–	–	ns
Data Hold Time	Read	t_{HR}		0	–	–	0	–	–	0	–	–	ns
	Write	t_{HW}		20	–	–	20	–	–	20	–	–	ns
Address Set-up Time for Latch		t_{ASL}		60	–	–	TBD	–	–	TBD	–	–	ns
Address Hold Time for Latch		t_{AHL}		20	–	–	TBD	–	–	TBD	–	–	ns
Address Hold Time		t_{AH}		20	–	–	20	–	–	20	–	–	ns
A_0 ~ A_7 Set-up Time Before E		t_{ASM}		200	–	–	TBD	–	–	TBD	–	–	ns
Peripheral Read Access Time	Non-Multiplexed Bus	(t_{ACCN})		–	–	(635)	–	–	TBD	–	–	TBD	ns
	Multiplexed Bus	(t_{ACCM})		–	–	(635)	–	–	TBD	–	–	TBD	ns
Oscillator stabilization Time		t_{RC}	Fig. 10	20	–	–	20	–	–	20	–	–	ms
Processor Control Set-up Time		t_{PCS}	Fig. 11	250	–	–	250	–	–	250	–	–	ns

PERIPHERAL PORT TIMING

Item		Symbol	Test Condition	HD6301V0			HD63A01V0			HD63B01V0			Unit
				min	typ	max	min	typ	max	min	typ	max	
Peripheral Data Set-up Time	Port 1, 2, 3, 4	t_{PDSU}	Fig. 3	200	–	–	200	–	–	200	–	–	ns
Peripheral Data Hold Time	Port 1, 2, 3, 4	t_{PDH}	Fig. 3	200	–	–	200	–	–	200	–	–	ns
Delay Time, Enable Positive Transition to $\overline{OS3}$ Negative Transition		t_{OSD1}	Fig. 5	–	–	300	–	–	300	–	–	300	ns
Delay Time, Enable Positive Transition to $\overline{OS3}$ Positive Transition		t_{OSD2}	Fig. 5	–	–	300	–	–	300	–	–	300	ns
Delay Time, Enable Negative Transition to Peripheral Data Valid	Port 1, 2*, 3, 4	t_{PWD}	Fig. 4	–	–	300	–	–	300	–	–	300	ns
Input Strobe Pulse Width		t_{PWIS}	Fig. 6	200	–	–	200	–	–	200	–	–	ns
Input Data Hold Time	Port 3	t_{IH}	Fig. 6	150	–	–	150	–	–	150	–	–	ns
Input Data Setup Time	Port 3	t_{IS}	Fig. 6	0	–	–	0	–	–	0	–	–	ns

* Except P_{21}

TIMER, SCI TIMING

Item	Symbol	Test Condition	HD6301V0			HD63A01V0			HD63B01V0			Unit
			min	typ	max	min	typ	max	min	typ	max	
Timer Input Pulse Width	t_{PWT}		2.5	–	–	TBD	–	–	TBD	–	–	t_{cyc}
Delay Time, Enable Positive Transition to Timer Out	t_{TOD}	Fig. 7	–	–	400	–	–	400	–	–	400	ns
SCI Input Clock Cycle	t_{Scyc}		2.0	–	–	TBD	–	–	TBD	–	–	μs
SCI Input Clock Pulse Width	t_{PWSCK}		0.4	–	0.6	0.4	–	0.6	0.4	–	0.6	t_{cyc}

MODE PROGRAMMING

Item	Symbol	Test Condition	HD6301V0			HD63A01V0			HD63B01V0			Unit
			min	typ	max	min	typ	max	min	typ	max	
$\overline{RES}$ "Low" Pulse Width	PW_{RSTL}	Fig. 8	3	–	–	3	–	–	3	–	–	t_{cyc}
Mode Programming Set-up Time	t_{MPS}		2	–	–	2	–	–	2	–	–	t_{cyc}
Mode Programming Hold Time	t_{HMP}		150	–	–	150	–	–	150	–	–	ns

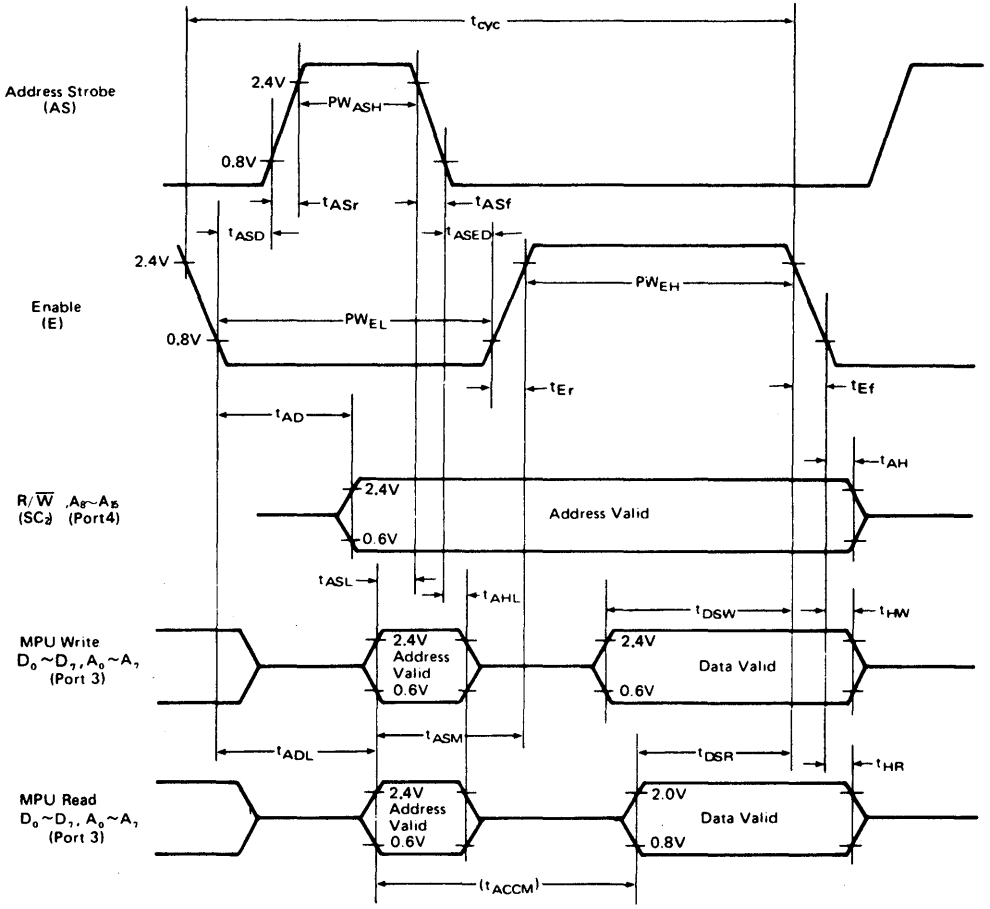


Figure 1 Expanded Multiplexed Bus Timing

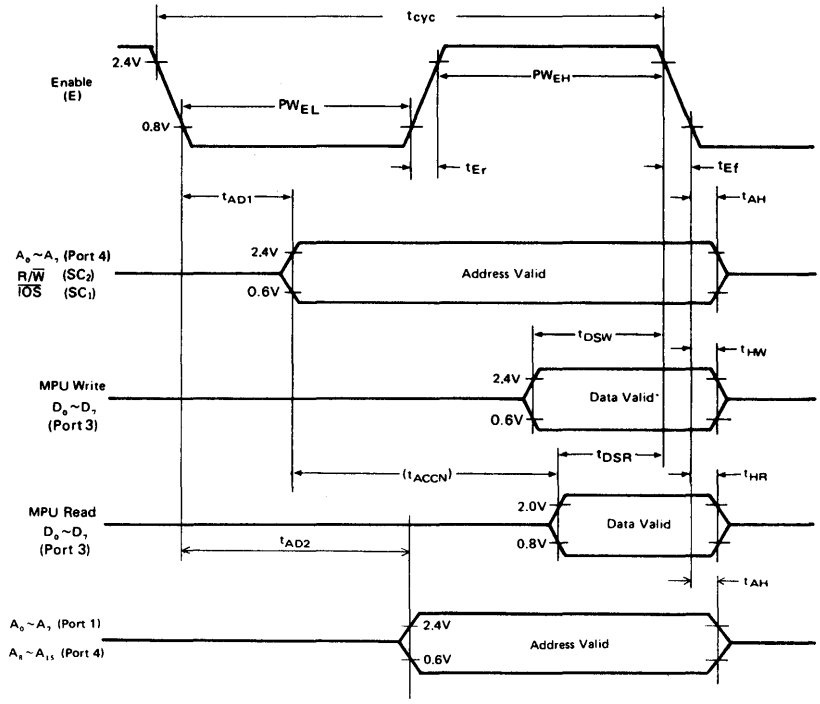
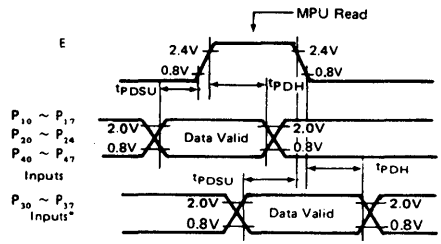
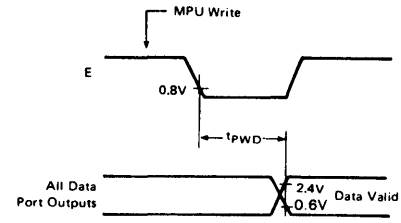


Figure 2 Expanded Non-Multiplexed Bus Timing



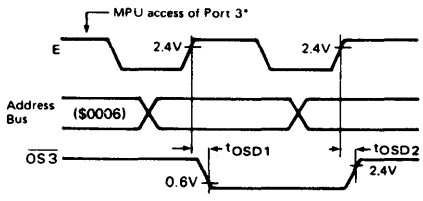
*Port 3 Non-Latched Operation

Figure 3 Port Data Set-up and Hold Times (MPU Read)



Note) Port 2: Except P_{21}

Figure 4 Port Data Delay Times (MPU Write)



*Access matches Output Strobe Select (OSS = 0, a read; OSS = 1, a write)

Figure 5 Port 3 Output Strobe Timing (Single Chip Mode)

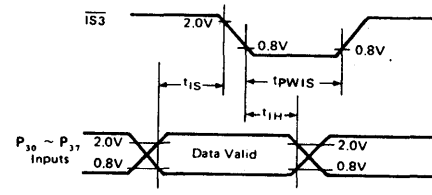


Figure 6 Port 3 Latch Timing (Single Chip Mode)

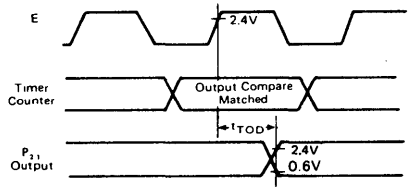


Figure 7 Timer Output Timing

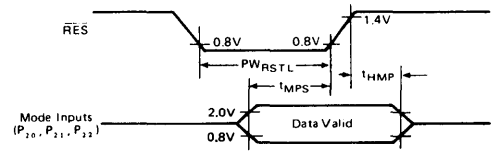
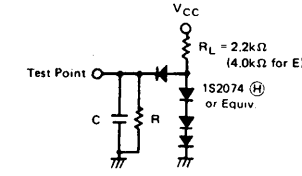


Figure 8 Mode Programming Timing



C = 90 pF for P_{10}~P_{37}, P_{40}~P_{47}, SC_1, SC_2
= 30 pF for P_{10}~P_{17}, P_{20}~P_{24}
= 40 pF for E
R = 12 kΩ for P_{10}~P_{17}, P_{20}~P_{24}, P_{30}~P_{37}, P_{40}~P_{47}, E, SC_1, SC_2

Figure 9 Bus Timing Test Loads (TTL Load)

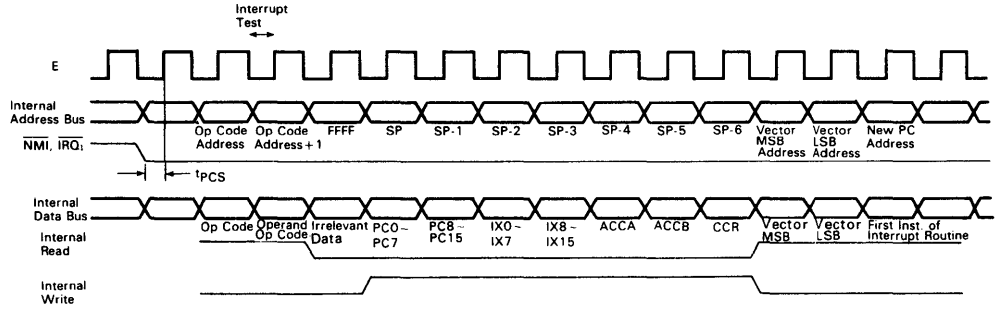


Figure 10 Interrupt Sequence

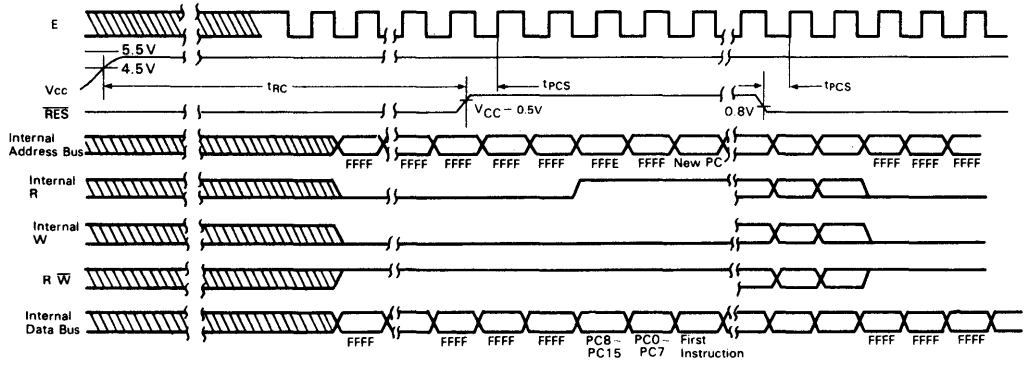


Figure 11 Reset Timing

■ FUNCTIONAL PIN DESCRIPTION

• V_{CC}, V_{SS}

These two pins are used for power supply and GND. Recommended power supply voltage is 5V ± 10% (HD6301V0, HD63A01V0), 5V ±5% (HD63B01V0) or 3 to 6V other than for high speed operation (500kHz).

• XTAL, EXTAL

These two pins are connected with parallel resonant fundamental crystal, AT cut. For instance, in order to obtain the system clock 1MHz, a 4MHz resonant fundamental crystal is useful because the devide by 4 circuitry is included. EXTAL accepts an external clock input of duty 50% (±10%) to drive, then internal clock is a quarter the frequency of an external clock. External driving frequency will be less than 4 times as maximum internal clock. For external driving, no XTAL should be connected. An example of connection circuit is shown in Fig. 12.

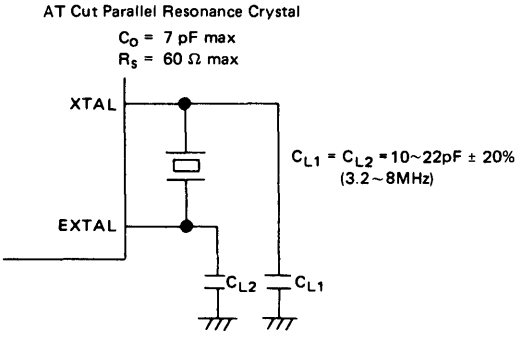


Figure 12 Crystal interface

• $\overline{STBY}$

This pin is used to place the MCU in the Stand-by mode. Setting to "Low" level, the internal condition is reset with inactive oscillation and fixed internal clock. In order to retain information in RAM during stand-by, write "0" into RAM enable bit (RAME). RAME is bit 6 of the RAM Control Register at address $0014. This disables the RAM, so the contents of RAM is guaranteed. For details of the stand-by mode, see the STAND-BY section.

• Reset ($\overline{RES}$)

This input is used to reset the MCU and start it from a power off condition. $\overline{RES}$ must be held "Low" for at least 20ms when power is on. To reset the MCU during system operation, it must be held "Low" at least 3 system clock cycles. From the third cycle on, all address buses become "High" with $\overline{RES}$ at "Low" level. Detecting "High" level, MPU does the following.

(1) I/O Port 2 bits 2,1,0 are latched into bits PC2, PC1, PC0 of program control register.
(2) The contents of the two Start Addresses, $FFFE, $FFFF are brought to the program counter, from which program starts (see Table 1).
(3) The interrupt mask bit is set. In order to have the MPU recognize the maskable interrupts $\overline{IRQ_1}$ and $\overline{IRQ_2}$, clear it beforehand.

• Enable (E)

With the internal oscillator in use, this supplies system clock for the rest of the system. Output is a single-phase, TTL compatible and 1/4 the crystal oscillation frequency. It will drive two LS TTL load and 40pF.

• Non maskable Interrupt ($\overline{NMI}$)

When the input signal of this pin is recognized to fall, NMI sequence starts. The current instruction may be continued to the last if $\overline{NMI}$ signal is detected as well as the following $\overline{IRQ_1}$ interrupt. Interrupt mask bit in Condition Code Register has no effect on NMI. In response to NMI interrupt, the information of Program Counter, Index Register, Accumulators, and Condition Code Register are stored on the stack. On completion of this sequence, vectoring address $FFFC and FFFD will occur to load the contents to the program counter and branch to a non maskable interrupt service routine.

Inputs $\overline{IRQ_1}$, and $\overline{NMI}$ are hardware interrupt lines sampled by internal clock. After the execution of instructions, start the interrupt routine in synchronization with E.

• Interrupt Request ($\overline{IRQ_1}$)

This level sensitive input requests that an interrupt sequence be generated within the machine. The MPU will wait receiving the request until it completes the current instruction that being executed before it recognizes the request. At that time, if the interrupt mask bit in Condition Code Register is not set, MPU begins interrupt sequence; otherwise, interrupt request is neglected.

Once the sequence has started, the information of Program Counter, Index Register, Accumulator, Condition Code Register are stored on the stack. Then the MPU sets the interrupt bit so that no further maskable interrupts may occur.

Table 1 Interrupt Vectoring memory map

	Vector		Interrupt
	MSB	LSB	
Highest Priority	FFFE	FFFF	$\overline{RES}$
	FFEE	FFEF	TRAP
	FFFC	FFFD	$\overline{NMI}$
	FFFA	FFFB	Software Interrupt (SWI)
	FFF8	FFF9	$\overline{IRQ_1}$ (or $\overline{IS3}$)
	FFF6	FFF7	ICF (Timer Input Capture)
	FFF4	FFF5	OCF (Timer Output Compare)
	FFF2	FFF3	TOF (Timer Overflow)
Lowest Priority	FFF0	FFF1	SCI (RDRF + ORFE + TDRE)

At the end of the cycle, the MPU generates 16 bit vectoring addresses indicating memory addresses $FFF8 and $FFF9, and locates the contents in Program Counter to branch to an interrupt service routine.

The Internal Interrut will generate signal ($\overline{IRQ_2}$) which is quite the same as $\overline{IRQ_1}$ except that it will use the vector address $FFF0 to $FFF7.

When $\overline{IRQ_1}$ and $\overline{IRQ_2}$ are generated at the same time, the former precede the latter. Interrupt Mask Bit in the condition code register, if being set, will keep the both interrupts off.

On occurrence of following Address Error or Op-code error, TRAP interrupt is invoked. This interrupt has priority next to $\overline{RES}$. Independently of the Mask Bit condition, the MPU will start an interrupt sequence. The vector for this interrupt will be FFEE, FFEF.

The following pins are available only for Port 3 in single chip mode.

- **Input Strobe ($\overline{IS3}$) (SC_1)**

This signal controls $\overline{IS3}$ interrupt and the latch of Port 3. When detected the signal fall, the flag of Port 3 Control Status Register is set.

For respective bits of Port 3 Control Status Register, see the I/O PORT 3 CONTROL STATUS REGISTER section.

- **Output Strobe ($\overline{OS3}$) (SC_2)**

This signal is used to strobe to an external device, indicating effective data is on the I/O pins. The timing chart for Output Strobe are shown in figure 5.

The following pins are available for Expanded Modes.

- **Read/Write (R/$\overline{W}$) (SC_2)**

This TTL compatible output signal indicates peripheral and memory devices whether MCU is in Read ("High"), or in Write ("Low"). The normal stand-by state is Read ("High"). Its output will drive one TTL load and 90pF.

- **I/O Strobe ($\overline{IOS}$) (SC_1)**

In expanded non multiplexed mode 5 of operation, $\overline{IOS}$ decodes internally A_9 to A_{15} as zero's and A_8 as a one. This allows external access up to 256 addresses from $0100 to $01FF in memory. The timing chart is shown in Figure 2.

- **Address Strobe (AS) (SC_1)**

In the expanded multiplexed mode, address strobe appears at this pin. It is used to latch the lower 8 bits addresses multiplexed with data at Port 3 and to control the 8-bit latch by address strobe as shown in Figure 18. Thereby, I/O Port 3 can become data bus during E pulse. The timing chart of this signal is shown in Figure 1.

■ PORTS

There are four I/O ports on HD6301V MCU (three 8-bit ports and one 5-bit port). 2 control pins are connected to one of the 8-bit port. Each port has an independent write-only data direction register to program individual I/O pins for input or output.*

When the bit of associated Data Direction Register is "1", I/O pin is programmed for output, if "0", then programmed for an input.

There are four ports.: Port 1, Port 2, Port 3, and Port 4. Addresses of each port and associated Data Direction Register are shown in Table 2.

* Only one exception is bit 1 of Port 2 which becomes either a data input or a timer output. It cannot be used as an output port.

Table 2 Port and Data Direction Register Addresses

Ports	Port Address	Data Direction Register Address
I/O Port 1	$0002	$0000
I/O Port 2	$0003	$0001
I/O Port 3	$0006	$0004
I/O Port 4	$0007	$0005

- **I/O Port 1**

This is an 8-bit port, each bit being defined individually as input or outputs by associated Data Direction Register. The 8-bit output buffers have three-state capability, maintaining in high impedance state when they are used for input. In order to be read accurately, the voltage on the input lines must be more than 2.0V for logic "1" and less than 0.8V for logic "0".

These are TTL compatible. After the MCU has been reset, all I/O lines are configured as inputs in all modes except mode 1. In all modes other than expanded non multiplexed mode, mode 1, Port 1 is always parallel I/O. In mode 1, Port 1 will be output line for lower order address lines (A_0 to A_7).

- **I/O Port 2**

This port has five lines, whose I/O direction depends on its data direction register. The 5-bit output buffers have three-state capability, going high impedance state when used as inputs. In order to be read accurately, the voltage on the input lines must be more than 2.0V for logic "1" and less than 0.8V for logic "0". After the MCU has been reset, I/O lines are configured as inputs. These pins on Port 2 (pins 10, 9, 8 of the chip) are used to program the mode of operation during reset. The values of these three pins during reset are latched into the upper 3 bits (bit 7, 6 and 5), which is expanded in the MODE SELECTION section.

In all modes, Port 2 can be configured as I/O lines. This port also provides access to the Serial I/O and the Timer. However, note that bit 1 (P_{21}) is the only pin restricted to data input or Timer output.

- **I/O Port 3**

This is an 8-bit port which can be configured as I/O lines, a data bus, or an address bus multiplexed with data bus. Its function depends on hardware operation mode programmed by the user using 3 bits of Port 2 during Reset. Port 3 as a data bus is bi-directional. For an input from peripherals, regular TTL level must be supplied, that is greater than 2.0V for a logic "1" and less than 0.8V for a logic "0". This TTL compatible three-state buffer can drive one TTL load and 90pF. In the expanded Modes, data direction register will be inhibited after Reset and data flow will be dependent on the state of the R/$\overline{W}$ line. Port 3 in each mode assumes the following characteristics.

Single Chip Mode (Mode 7): Parallel Inputs/Outputs as programmed by its corresponding Data Direction Register.

There are two control lines associated with this port in this mode, an input strobe ($\overline{IS3}$) and an output strobe ($\overline{OS3}$), both being used for handshaking. They are controlled by I/O Port 3 Control/Status Register. Additional 3 characteristics of Port 3 are summarized as follows:

(1) Port 3 input data can be latched using $\overline{IS3}$ (SC_1) as a control signal.
(2) $\overline{OS3}$ can be generated by MPU read or write to Port 3's data register.
(3) $\overline{IRQ_1}$ interrupt can be generated by an $\overline{IS3}$ negative edge.

Port 3 strobe and latch timing is shown in Figs. 5 and 6, respectively.

I/O Port 3 Control/Status Register

	7	6	5	4	3	2	1	0
	$\overline{IS3}$	$\overline{IS3}$	X	OSS	LATCH	X	X	X
$000F	FLAG	$\overline{IRQ_1}$ ENABLE			ENABLE			

Bit 0 Not used.
Bit 1 Not used.

Bit 2 Not used.

Bit 3 LATCH ENABLE.

Bit 3 is used to control the input latch of Port 3. If the bit is set at "1", the input data on Port 3 is latched by the falling edge of $\overline{IS3}$. The latch is cleared by the MCU read to Port 3; it can now be latched again. Bit 3 is cleared by a reset.

Bit 4 OSS (Output Strobe Select)

This bit identifies the cause of output strobe generation: a write operation or read operation to I/O Port 3. When the bit is cleared, the strobe will be generated by a read operation to Port 3. When the bit is not cleared, the strobe will be generated by a write operation. Bit 4 is cleared by a reset.

Bit 5 Not used.

Bit 6 $\overline{IS3}$ ENABLE.

If the $\overline{IS3}$ flag (bit 7) is set with bit 6 set, an interrupt is enabled. Clearing the flag causes the interrupt to be disabled. The bit is cleared by a reset.

Bit 7 $\overline{IS3}$ FLAG.

Bit 7 is a read-only bit which is set by the falling edge of $\overline{IS3}$ (SC_1). It is cleared by a read of the Control/Status Register followed by a read/write of I/O Port 3. The bit is cleared by reset.

Expanded non multiplexed mode (mode 1,5)

In this mode, Port 3 becomes data bus. (D_0 to D_7)

Expanded Multiplexed Mode (mode 0, 4, 6)

Port 3 becomes both the data bus ($D_0 \sim D_7$) and lower bits of the address bus ($A_0 \sim A_7$). An address strobe output is true when the address is on the port.

- **I/O Port 4**

This is an 8-bit port that becomes either I/O or address outputs depending on the operation mode selected. In order to be read accurately, the voltage at the input lines must be greater than 2.0V for a logic "1", and less than 0.8V for a logic "0". For outputs, each line is TTL compatible and can drive one TTL load and 90pF. After reset, this port becomes inputs. To use these pins as addresses, they should be programmed as outputs.

In each mode, Port 4 assumes following characteristics.

Single Chip Mode (Mode 7): Parallel Inputs/Outputs as programmed by its associated data direction register.

Expanded Non Multiplexed Mode (Mode 5): In this mode, Port 4 becomes the lower address lines (A_0 to A_7) by writing "1"s on the data direction register.

When all of the eight bits are not required as addresses, the remaining lines can be used as I/O lines (Inputs only) starting with the MSB.

Expanded Non Multiplexed Mode (Mode 1): In this mode, Port 4 becomes output for upper order address lines (A_8 to A_{15}).

Expanded Multiplexed Mode (Mode 0, 4): In this mode, Port 4 becomes output for upper order address lines (A_8 to A_{15}) regardless of the value of data direction register. The relation between each mode and I/O Port 1 to 4 is summarized in Table 3.

■ MODE SELECTION

The operation mode after the rest must be determined by the user wiring the 10, 9, and 8 externally. These three pins are lower order bits; I/O 0, I/O 1, I/O 2 of Port 2. They are latched into the programmed control bits PC0, PC1, PC2 in I/O Port 2 register when reset goes "High", I/O Port 2 Register is shown below.

Port 2 DATA REGISTER

	7	6	5	4	3	2	1	0
$0003	PC2	PC1	PC0	I/O 4	I/O 3	I/O 2	I/O 1	I/O 0

An example of external hardware used for Mode Selection is shown in Fig. 13. During reset, the HD14053B is available to seperate the peripheral device from the MCU. It is necessary where the data conflict can occur between peripheral device and Mode generation circuit.

No mode can be changed through software because the bits 5, 6, and 7 of Port 2 are for read only. The mode selection of the HD6301V0 is shown in Table 4.

The HD6301V0 operates in three basic modes: (1) Single Chip Mode, (2) Expanded Multiplexed Mode (compatible with the HMCS6800 peripheral family), (3) Expanded Non Multiplexed Mode (compatible with HMCS6800 peripheral family)

- **Single Chip Mode**

In the Single Chip Mode, all ports will become I/O. This is shown in figure 15. In this mode, SC1, SC2 pins are configured for control lines of Port 3 and can be used as input strobe ($\overline{IS3}$) and output strobe ($\overline{OS3}$) for handshaking data.

- **Expanded Multiplexed Mode**

In this mode, Port 4 is configured for I/O (inputs only) or address lines. The data bus and the lower order address bus are multiplexed in Port 3 and can be seperated by an output called Address Strobe.

Port 2 is configured for 5 parallel I/O or Serial I/O, or Timer, or any combination thereof. Port 1 is configured for 8 parallel I/O. In this mode, HD6301V0 is expandable to 65k words (See Fig. 16).

- **Expanded Non Multiplexed Mode**

In this mode, the HD6301V0 can directly address HMCS6800 peripherals with no external logic. In mode 5, Port 3 becomes a data bus. Port 4 becomes A_0 to A_7 address bus or partial address bus and I/O (inputs only). Port 2 is configured for a parallel I/O, Serial I/O, Timer or any combination thereof. Port 1 is configured as a parallel I/O only.

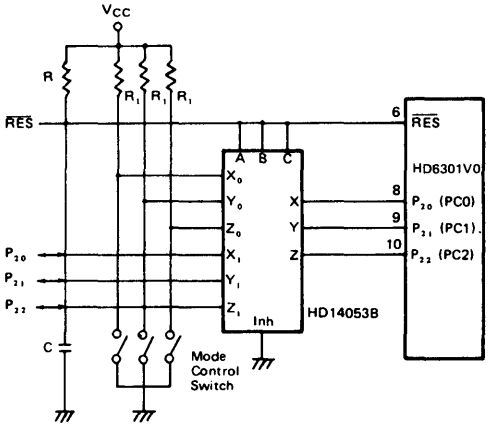
In this mode, HD6301V0 is expandable to 256 locations. In the application system enough with fewer addresses, idle pins of Port 4 can be used as I/O lines (inputs only)(See Fig. 17).

In mode 1, Port 3 becomes a data bus and Port 1 becomes A_0 to A_7 address bus, and Port 4 becomes A_8 to A_{15} address bus.

In this mode, the HD6301V0 is expandable to 65k words with no external logic.

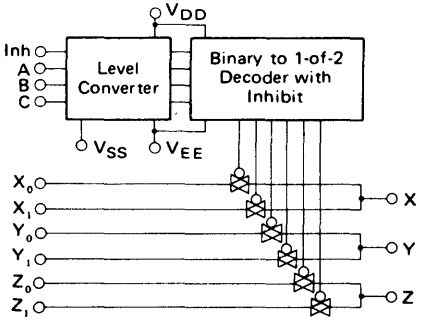
- **Lower Order Address Bus Latch**

Because the data bus is multiplexed with the lower order address bus in Port 3 in the expanded multiplexed mode, address bits must be latched outside the board. It requires the 74LS373 Transparent octal D-type to latch the LSB. Latch connection of the HD6301V0 is shown in Figure 18.



Note 1) Figure of Mode 7
2) RC≈Reset Constant
3) R_1=10kΩ

Figure 13 Recommended Circuit for Mode Selection



Truth Table

Control Input				On Switch		
Inhibit	Select					
	C	B	A	HD14053B		
0	0	0	0	Z_0	Y_0	X_0
0	0	0	1	Z_0	Y_0	X_1
0	0	1	0	Z_0	Y_1	X_0
0	0	1	1	Z_0	Y_1	X_1
0	1	0	0	Z_1	Y_0	X_0
0	1	0	1	Z_1	Y_0	X_1
0	1	1	0	Z_1	Y_1	X_0
0	1	1	1	Z_1	Y_1	X_1
1	X	X	X	–		

Figure 14 HD14053B Multiplexers/De-Multiplexers

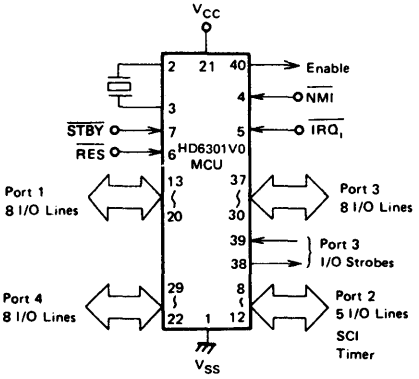


Figure 15 HD6301V0 MCU Single-Chip Mode

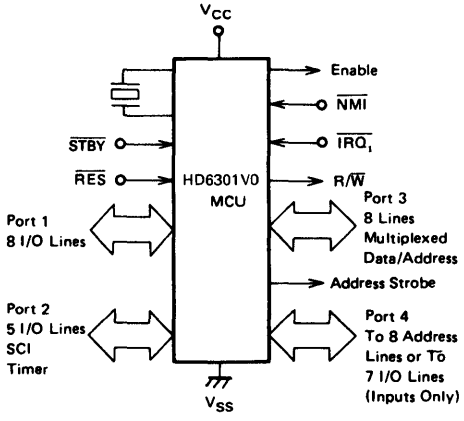


Figure 16 HD6301V0 MCU Expanded Multiplexed Mode

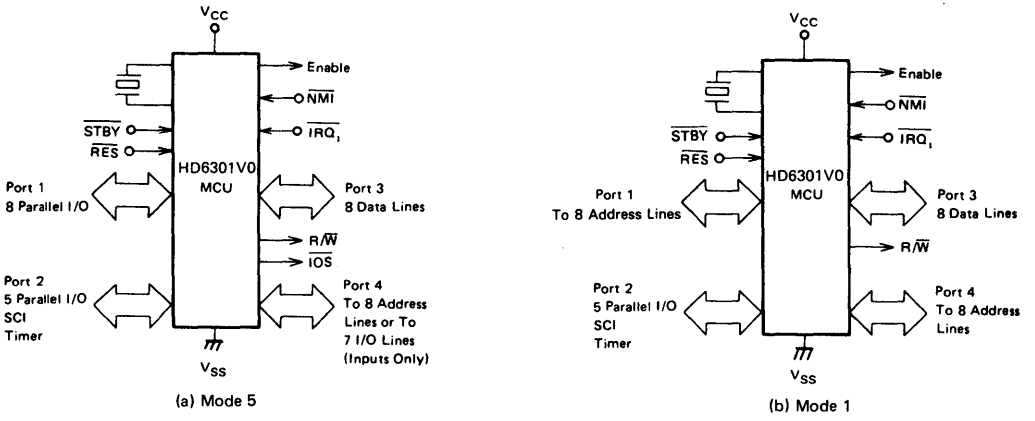
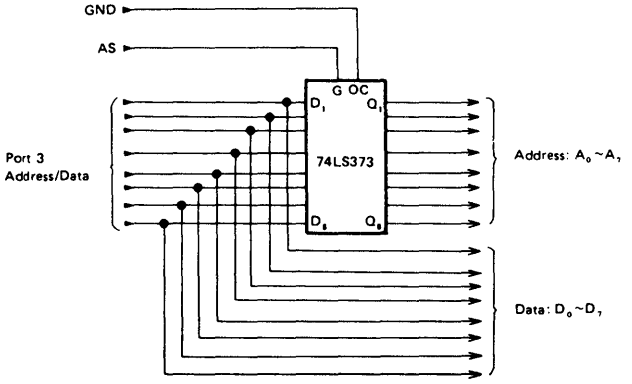


Figure 17 HD6301V0 MCU Expanded Non Multiplexed Mode



Function Table

Output Control	Enable G	Enable D	Output Q
L	H	H	H
L	H	L	L
L	L	X	Q_0
H	X	X	Z

Figure 18 Latch Connection

- **Mode and Port Summary MCU Signal Description**

This section gives a description of the MCU signals for the various modes. SC_1 and SC_2 are signals which vary with the mode that the chip is in.

Table 3 Feature of each mode and lines

MODE		PORT 1 Eight Lines	PORT 2 Five Lines	PORT 3 Eight Lines	PORT 4 Eight Lines	SC_1	SC_2
SINGLE CHIP		I/O	I/O	I/O	I/O	$\overline{IS3}$ (I)	$\overline{OS3}$ (O)
EXPANDED MUX		I/O	I/O	ADDRESS BUS (A_0~A_7) DATA BUS (D_0~D_7)	ADDRESS BUS* (A_8~A_{15})	AS(O)	R/$\overline{W}$(O)
EXPANDED	Mode 5	I/O	I/O	DATA BUS (D_0~D_7)	ADDRESS BUS* (A_0~A_7)	$\overline{IOS}$(O)	R/$\overline{W}$(O)
NON-MUX	Mode 1	ADDRESS BUS (A_0~A_7)	I/O	DATA BUS (D_0~D_7)	ADDRESS BUS (A_8~A_{15})	Not Used	R/$\overline{W}$(O)

*These lines can be substituted for I/O (Input Only) starting with the MSB (except Mode 0, 4).

I = Input
O = Output
R/$\overline{W}$ = Read/Write
$\overline{IS3}$ = Input Strobe
$\overline{OS3}$ = Output Strobe
$\overline{IOS}$ = I/O Select
SC = Strobe Control
AS = Address Strobe

Table 4 Mode Selection Summary

Mode	P_{22} (PC2)	P_{21} (PC1)	P_{20} (PC0)	ROM	RAM	Interrupt Vectors	Bus Mode	Operating Mode
7	H	H	H	I	I	I	I	Single Chip
6	H	H	L	I	I	I	MUX(4)	Multiplexed/Partial Decode
5	H	L	H	I	I	I	NMUX(4)	Non-Multiplexed/Partial Decode
4	H	L	L	E(2)	I(1)	E	MUX	Multiplexed/RAM
3	L	H	H	—	—	—	—	Not Used
2	L	H	L	—	—	—	—	Not Used
1	L	L	H	E(2)	I	I	NMUX	Non-Multiplexed
0	L	L	L	I	I	I(3)	MUX	Multiplexed Test

LEGEND :
I — Internal
E — External
MUX — Multiplexed
NMUX — Non-Multiplexed
L — Logic "0"
H — Logic "1"

(NOTES)
1) Internal RAM is addressed at $0080.
2) Internal ROM is disabled.
3) Reset vector is external for 3 or 4 cycles after $\overline{RES}$ goes "high".
4) Idle lines of Port 4 address outputs can be assigned to Input Port.

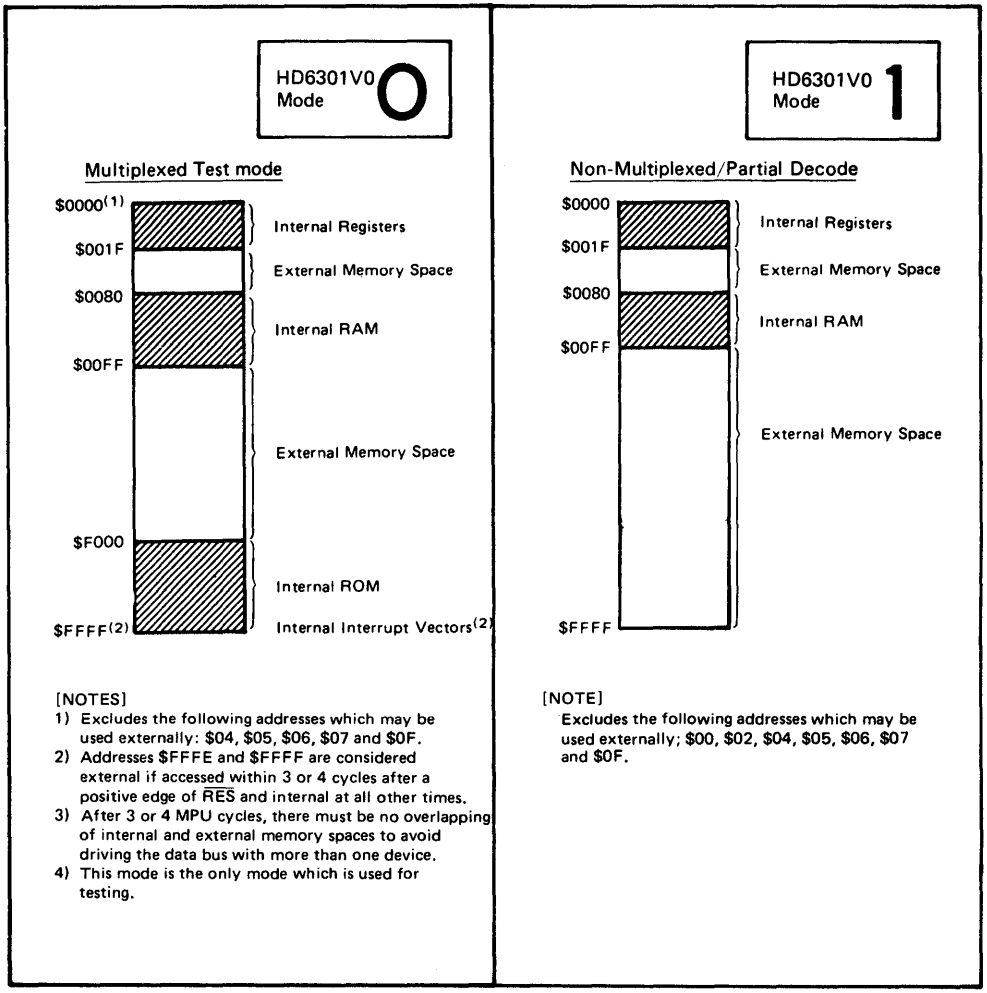
■ Memory Map

The MCU can provide up to 65k byte address space depending on the operating mode. Fig. 19 shows a memory map for each operating mode. The first 32 locations of each map are for the MCU's internal register only, as shown in Table 5.

Table 5 Internal Register Area

Register	Address
Port 1 Data Direction Register****	00*
Port 2 Data Direction Register****	01
Port 1 Data Register	02*
Port 2 Data Register	03
Port 3 Data Direction Register****	04**
Port 4 Data Direction Register****	05***
Port 3 Data Register	06**
Port 4 Data Register	07***
Timer Control and Status Register	08
Counter (High Byte)	09
Counter (Low Byte)	0A
Output Compare Register (High Byte)	0B
Output Compare Register (Low Byte)	0C
Input Capture Register (High Byte)	0D
Input Capture Register (Low Byte)	0E
Port 3 Control and Status Register	0F**
Rate and Mode Control Register	10
Transmit/Receive Control and Status Register	11
Receive Data Register	12
Transmit Data Register	13
RAM Control Register	14
Reserved	15-1F

* External address in Mode 1
** External address in modes 0, 1, 4, 6 ; cannot be accessed in Mode 5
*** External address in Modes 0, 1, 4
**** 1 = Output, 0 = Input

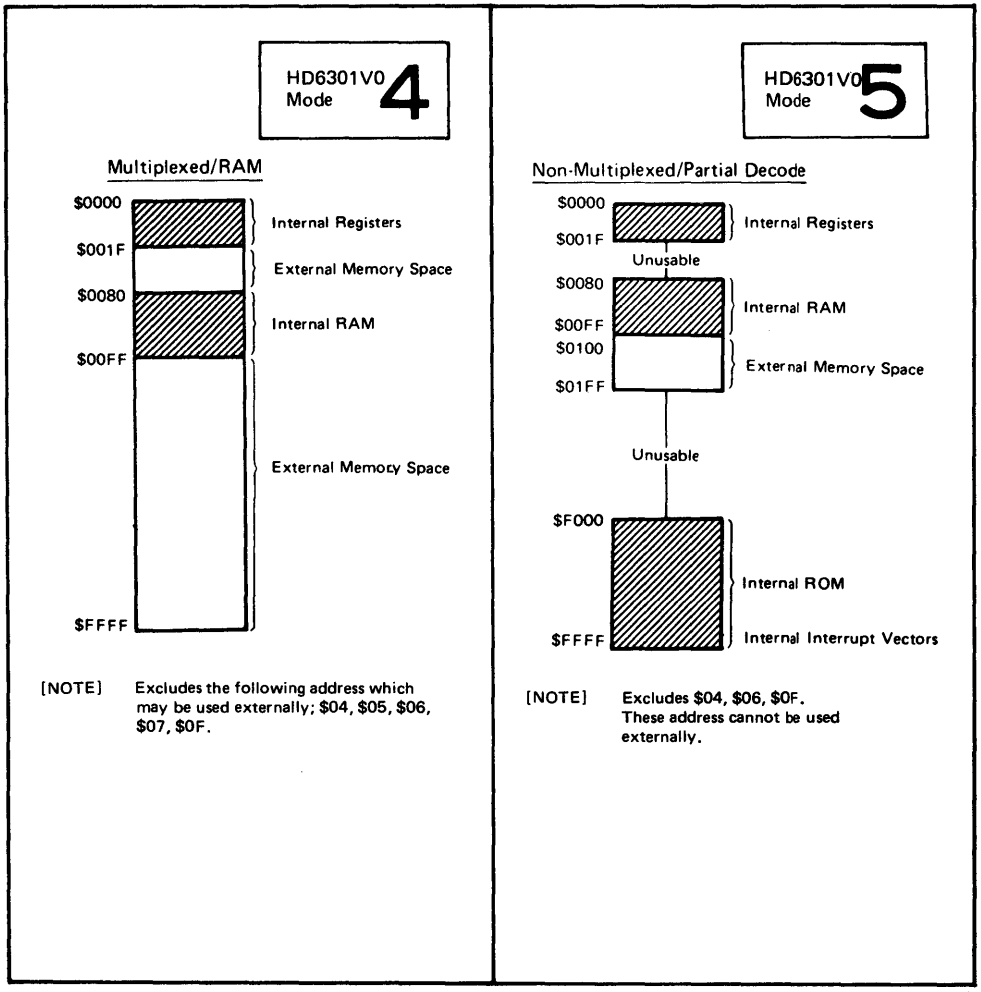


[NOTES]
1) Excludes the following addresses which may be used externally: $04, $05, $06, $07 and $0F.
2) Addresses $FFFE and $FFFF are considered external if accessed within 3 or 4 cycles after a positive edge of $\overline{RES}$ and internal at all other times.
3) After 3 or 4 MPU cycles, there must be no overlapping of internal and external memory spaces to avoid driving the data bus with more than one device.
4) This mode is the only mode which is used for testing.

[NOTE]
Excludes the following addresses which may be used externally; $00, $02, $04, $05, $06, $07 and $0F.

(to be continued)

Figure 19 HD6301V0 Memory Maps



(to be continued)

Figure 19 HD6301V0 Memory Maps

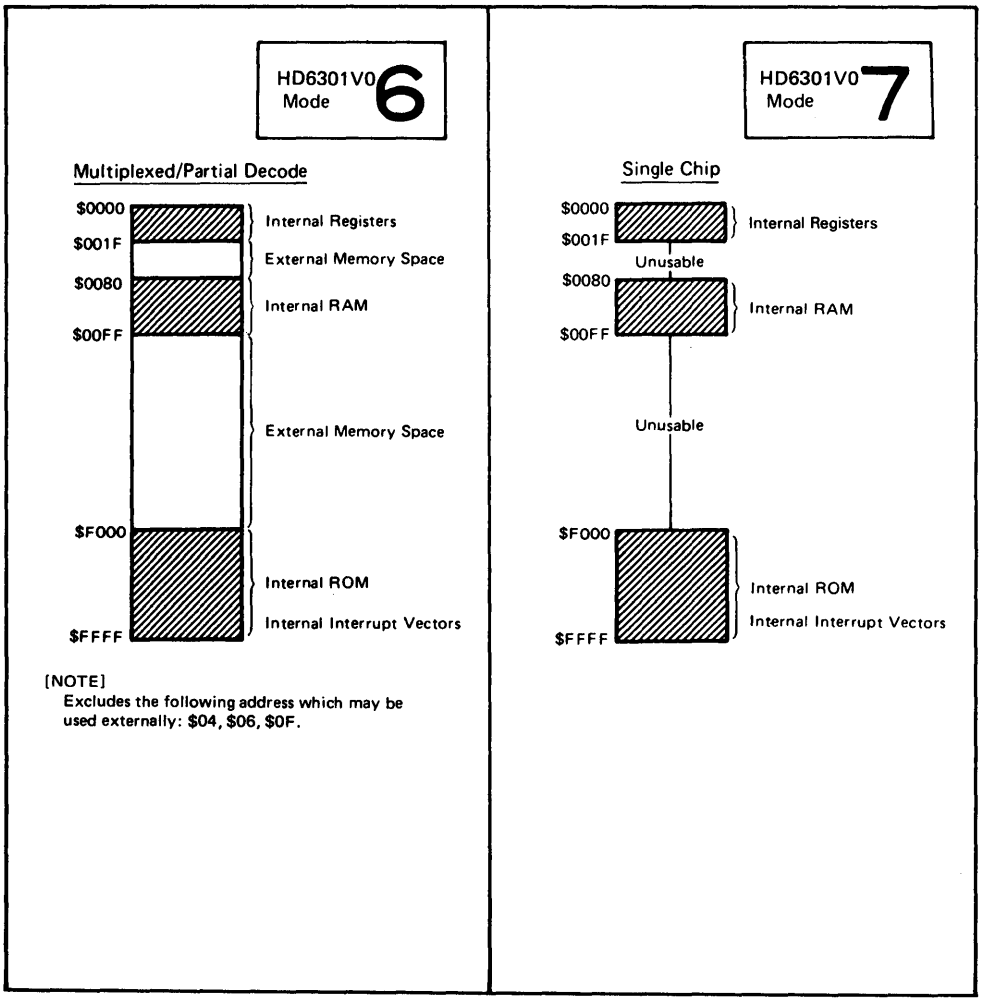


Figure 19 HD6301V0 Memory Maps

■ PROGRAMMABLE TIMER

The HD6301V0 contains 16-bit programmable timer and used to make measurement of input waveform. In addition independently it can generate an output waveform by itself. For both input and output waveform, the pulse width may vary from a few microseconds to many seconds.

The timer hardware consists of

- an 8-bit control and status register
- a 16-bit free running counter
- a 16-bit output compare register, and
- a 16-bit input capture register

A block diagram of the timr is shown in Figure 20.

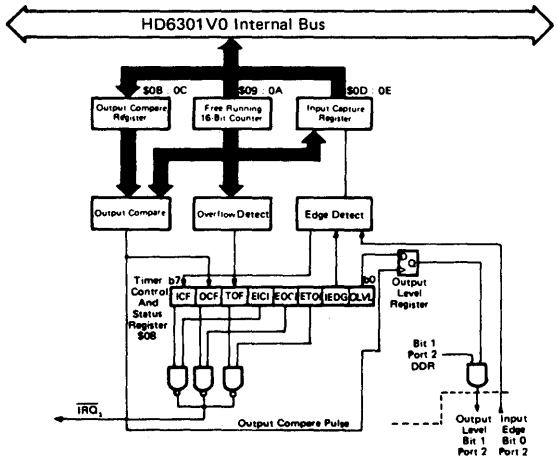


Figure 20 Programmable Timer Block Diagram

• Free Running Counter ($0009:000A)

The key element in the programmable timer is a 16-bit free running counter, that is driven by an E (Enable) clock to increment its values. The counter value will be read out by the MPU software at any time desired with no effects on the counter. Reset will clear the counter.

When the MPU writes arbitrary data to the MSB of $09, then value of $FFF8 is being pre set to the counte ($09, $0A) indepently of the write data value. When the MPU writes arbitrary data to the LSB ($0A), the data is set to the "Low", on the other hand, the data preceedingly written in "High" byte is set to "High".

The counter value written to the counter using the double store instruction is shown in Figure 21.

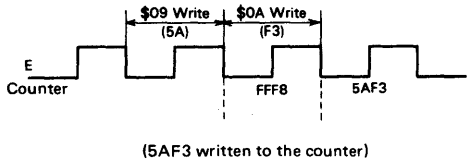


Figure 21 Counter Write Timing

* To write to the counter can disturb serial operations, so it should be inhibited during using the SCI.

• Output Compare Register ($000B:$000C)

This is a 16-bit read/write register which is used to control an output waveform. The contents of this register are constantly being compared with current value of the free running counter.

When the contents match with the value of the free running counter, a flag (OCF) in the timer control/status register (TCSR) is set and the current value of an output level Bit (OLVL) in the TCSR is transferred to Port 2 bit 1. When bit 1 of the Port 2 data direction register is "1" (output), the OLVL value will appear on the bit 1 of Port 2. Then, the value of Output Compare Register and Output level bit should be changed to control an output level again on the next compare values.

The output compare register is set to $FFFF during reset. The compare function is inhibited for one cycle immediately after writing of the output compare register to the upper bytes, or, a writing of the Counter to the upper bytes. This is because for one thing, the operation makes sure to set the values of 16 bits in the register before comparing, and for another, the counter is set to FFF8 in the cycle following writing to "High" of the counter.*

* For the data writing on Compare Register, 2-byte transfer instruction such as STX is available.

• Input Capture Register ($000D: $000E)

The input capture register is a 16-bit read-only register used to hold the current value of free running counter obtained when the proper transition of an external input signal occurs.

The input transition change required to trigger the counter transfer is controlled by the input Edge bit (IEDG).

To allow the external input signal to gate in the edge detect unit, the bit of the Data Direction Register corresponding to bit 0 of Port 2 must have been cleared (to zero).

To insure input capture in all cases, the width of an input pulse requires at least 2 Enable cycles.

• Timer Control/Status Register (TCSR) ($0008)

This is an 8-bit register. All 8 bits are readable and the lower 5 bits may be written. The upper 3 bits are read-only, indicating the timer status information below.

(1) A proper transition has taken place on the input pin with a subsequent transfer of the current counter value to the input capture register (ICF).
(2) A match has been found between the value in the free running counter and the output compare register (OCF).
(3) When counting up to $0000 (TOF).

Each flag may contains an individual enable bit in TCSR where controls whether or not an interrupt request may be output to internal interrupt signal ($\overline{IRQ_2}$). If the I-bit in Condition Code Register has been cleared, a priority vectored address occurs corresponding to each flag being set. A description of each bit is as follows.

Timer Control / Status Register

7	6	5	4	3	2	1	0	
ICF	OCF	TOF	EICI	EOCI	ETOI	IEDG	OLVL	$0008

Bit 0 OLVL (Output Level); When a match is found in the value between the counter and the output compare register, this bit is transferred to the Port 2 bit 1. If the DDR corresponding to Port 2 bit 1 is set "1", the value will appear on the output pin of Port 2 bit 1.

Bit 1 IEDG (Input Edge); This bit control which transition of an input of Port 2 bit 0 will trigger the data transfer from the counter to the input capture register. The DDR corresponding to Port 2 bit 0 must be clear in advance of using this function. When IEDG = 0, trigger takes place on a negative edge ("High"-to-"Low" transition). When IEDG = 1, trigger takes place on a leading edge ("Low"-to-"High" transition).

Bit 2 ETOI (Enable Timer Overflow Interrupt); When set, this bit enables TOF interrupt to generate the interrupt request ($\overline{IRQ_2}$) but when clear, the interrupt is inhibited.

Bit 3 EOCI (Enable Output Compare Interrupt); When set, this bit enables OCF interrupt to generate the interrupt request ($\overline{IRQ_2}$), when clear, the interrupt is inhibited.

Bit 4 EICI (Enable Input Capture Interrupt); When set, this bit enables ICF interrupt to generate the interrupt request ($\overline{IRQ_2}$) but when clear, the interrupt is inhibited.

Bit 5 TOF (Timer Over Flow Flag); This read-only bit is set when the counter value is $0000. It is cleared by MPU read of TCSR (with TOF set) following an MPU read of the counter ($0009).

Bit 6 OCF (Output Compare Flag); This read-only bit is set when a match is found in the value between the output compare register and the counter. It is cleared by a read of TCSR (with OCF set) following an MCU write to the output compare register ($000B or $000C).

Bit 7 ICF (Input Capture Flag); The read-only bit is set by a proper transition on the input, and is cleared by a read of TCSR (with ICF set) followed by an MPU read of Input Capture Register ($000D).

Reset will clear each bit of Timer Control and Status Register.

■ SERIAL COMMUNICATION INTERFACE

The HD6301V0 contains a full-duplex asynchronous Serial Communication Interface (SCI). SCI may select the several kinds of the data rate and comprises a transmitter and a receiver which operate independently on each other but with the same data format at the same data rate. Both of transmitter and receiver communicate with the MPU via the data bus and with the outside world, through Port 2 bit 2, 3 and 4. Description of hardware, software, register is as follows.

• Wake-Up Feature

In typical multiprocessor applications the software protocol will usually have the designated address at the initial byte of the message. The purpose of Wake-Up feature is to have the non-selected MPU neglect the remainder of the message. Thus the non-selected MPU can inhibit the all further interrupt process until the next message begins.

Wake-Up feature is triggered by a ten consecutive "1"s which indicates an idle transmit line. Therefore software protocol needs an idle period between the messages.

With this hardware feature, the non-selected MPU be re-enabled (or "wakes-up") for the appearing next message.

• Programmable Option

The HD6301V0 has the following optional features provided for its Serial I/O. They are all programmable.

- data format ; standard mark/space (NRZ)
- Clock Source ; external or internal
- baud rate ; one of 4 rates per given MPU E clock frequency or 1/8 of external clock
- wake-up feature ; Enabled or disabled
- Interrupt requests ; enabled or masked individually for transmitter and receive data registers
- Clock Output ; internal clock enabled or disabled to Port 2 bit 2
- Port 2 (bits 3, 4) ; dedicated or not dedicated to serial I/O individually for receiver and transmitter

• Serial Communication Hardware

The serial communications hardware is controlled by 4 registers as shown in Figure 22. The registers include:

- an 8-bit control/status register
- a 4-bit rate/mode control register (write-only)
- an 8-bit read-only receive data register
- an 8-bit write-only transmit data register

Besides these 4 registers, Serial I/O utilizes Port 2 bit 3 (input) and bit 4 (output). Port 2 bit 2 can be used when an option is selected for the internal-clock-out or the external-clock-in.

• Transmit/Receive Control Status Register (TRCSR)

TRCS Register consists of 8 bits which all may be read while only bits 0 to 4 may be written. The register is initialized to $20 on $\overline{RES}$. The bits of the TRCS register are defined as follows.

Transmit / Receiver Control Status Register

7	6	5	4	3	2	1	0	
RDRF	ORFE	TDRE	RIE	RE	TIE	TE	WU	ADDR: $0011

Bit 0 WU (Wake Up); Set by software and clear by hardware on receipt of ten consecutive "1"s. It should be noted that RE flag has already set in advance of WU flag's set.

Bit 1 TE (Transmit Enable); Set to produce preample of ten consecutive "1"s and to enable the data of transmitter to output subsequently to the Port 2 bit 4 independently of its corresponding DDR value. When clear, serial I/O affects nothing on Port 2 bit 4.

Bit 2 TIE (Transmit Interrupt Enable); When this bit is set with TDRE (bit 5) set, it will permit an $\overline{IRQ_2}$ interrupt. When clear, TDRE interrupt is masked.

Bit 3 RE (Receive Enable); When set, gates Port 2 bit 3 to input of receiver regardless of DDR value for this bit. When clear, the serial I/O affects nothing on Port 2 bit 3.

Bit 4 RIE (Receive Interrupt Enable); When this bit is set with bit 7 (RDRF) or a bit 6 (ORFE) set, it will permit an $\overline{IRQ_2}$. When clear, $\overline{IRQ_2}$ interrupt is masked.

Bit 5 TDRE (Transmit Data Register Empty); When the data transfer is made from the Transmit Data Register to Output Shift Register, it is set by hardware. The bit is cleared by reading the status register and followed by writing the next new data into the Transmit Data Register. TDRE is initialized to 1 by $\overline{RES}$.

Bit 6 ORFE (Over Run Framing Error); When overrun or

framing error occurs (receive only), it is set by hardware. Over Run Error occurs if the attempt is made to transfer the new byte to the receive data register with the RDRF set. Framing Error occurs when the bit counters are not synchronized with the boundary of the byte in the bit stream. The bit is cleared by reading the status register and followed by reading the receive data register, or by $\overline{RES}$.

Bit 7 RDRF (Receive Data Register Full); It is set by hardware when the data transfer is made from the receive shift register to the receive data register. It is cleared by reading the status register and followed by reading the receive data register, or by $\overline{RES}$.

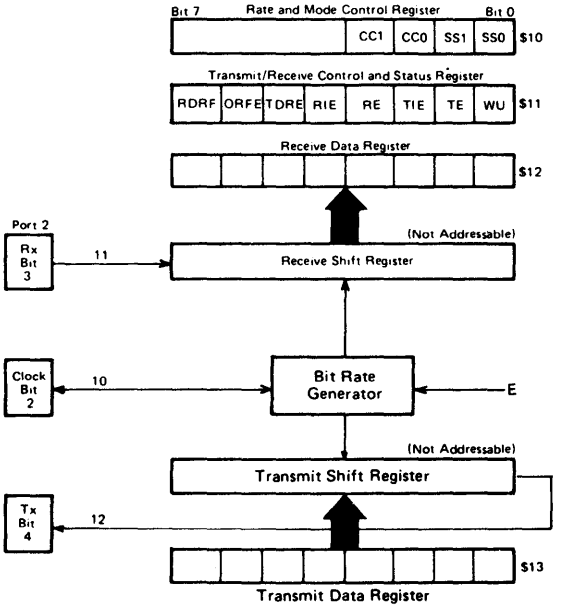
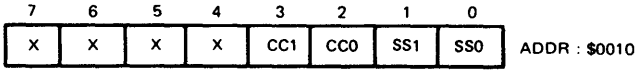


Figure 22 Serial I/O Register



Transfer Rate / Mode Control Register

Table 6 SCI Bit Times and Transfer Rates

SS1 : SS0	XTAL E	2.4576 MHz 614.4 kHz	4.0 MHz 1.0 MHz	4.9152MHz 1.2288MHz
0 0	E ÷ 16	26 μs/38,400 Baud	16 μs/62,500 Baud	13 μs/76,800Baud
0 1	E ÷ 128	208μs/4,800 Baud	128 μs/7812.5 Baud	104.2μs/ 9,600Baud
1 0	E ÷ 1024	1.67ms/600 Baud	1.024ms/976.6 Baud	833.3μs/ 1,200Baud
1 1	E ÷ 4096	6.67ms/150 Baud	4.096ms/244.1 Baud	3.333ms/ 300Baud

Table 7 SCI Format and Clock Source Control

CC1, CC0	Format	Clock Source	Port 2 Bit 2	Port 2 Bit 3	Port 2 Bit 4
00	–	–	–	–	–
01	NRZ	Internal	Not Used	**	**
10	NRZ	Internal	Output*	**	**
11	NRZ	External	Input	**	**

* Clock output is available regardless of values for bits RE and TE.
** Bit 3 is used for serial input if RE = "1" in TRCS.
Bit 4 is used for serial output if TE = "1" in TRCS.

- **Transfer rate/Mode Control Register (RMCR)**

The register controls the following serial I/O variables:

- Bauds rate
- data format
- clock source
- Port 2 bit 2 feature

It is 4-bit write-only register, cleared by $\overline{RES}$. The 4 bits are considered as a pair of 2-bit fields. The lower 2 bits control the bit rate of internal clock while the upper 2 bits control the format and the clock select logic.

Bit 0 SS0
Bit 1 SS1 } Speed Select

These bits select the Baud rate for the internal clock. The rates selectable are function of E clock frequency within the MPU. Table 6 lists the available Band Rates.

Bit 2 CC0
Bit 3 CC1 } Clock Control/Format Select

They control the data format and the clock select logic. Table 7 defines the bit field.

- **Internally Generated Clock**

If the user wish to employ externaly a internal clock for the serial I/O, the following requirements should be noted.

- The values of RE and TE have no effect.
- CC1, CC0 must be set to "10".
- The maximum clock rate will be E/16.
- The clock is once the bit rate.

- **Externally Generated Clock**

If the user wish to supply an external clock for the Serial I/O, the following requirements should be noted.

- The CC1, CC0, field in the Rate and Mode Control Register must be set to "11" (See Table 7).
- The external clock must be set to 8 times the desired baud rate.
- The maximum external clock frequency is the same as E clock.

- **Serial Operations**

The serial I/O hardware must be initialized by the HD6301V0 software prior to operation. The sequence will be normally as follows.

- Writing the desired operation control bits to the Rate and Mode Control Register.
- Writing the desired operation control bits to the TRCS register.

If using Port 2 bit 3, 4 for serial I/O exclusively, TE, RE bits may be preserved set. When TE, RE bit cleared during SCI operation, and subsequently set again, it should be noted that the setting of TE, RE must refrain for at least one bit time of the current baud rate. If set within one bit time, there may be the case where the initializing of internal function for transmit and receive does not take place.

- **Transmit Operation**

Data transmission is enabled by the TE bit in the TRCS register. When set, gates the output of the serial transmit shift register to Port 2 bit 4 which is unconditionally configured as an output irrespectively of corresponding DDR value.

Following $\overline{RES}$ the user should configure both the RMC register and the TRCS register for desired operation. Setting the TE bit during this procedure causes a transmission of ten-bit preamble of "1"s. Following the preamble, internal synchronization is established and the transmitter section is ready to operate. Then either of the following states exists.

(1) If the transmit data register is empty (TDRE = 1), the consecutive "1"s are transmitted indicating an idle lines.

(2) If the data has been loaded into the Transmit Data Register (TDRE = 0), it is transferred to the output shift register and data transmission begins.

During the data transfer, the 0 start bit is first transferred. Next the 8-bit data (beggining at bit0) and the stop bit. When the transmit data register has been empty, the hardware sets the TDRE flag bit: If the MCU fails to respond to the flag within the proper time, TDRE is preserved set and then a 1 will be sent (instead of a 0 at start bit time) and more 1s will be set successively until the data is supplied to the data register. While the TDRE remains a "1", no "0" will be sent.

- **Receive Operation**

The receive operation is enabled by the RE bit, gating the serial input through Port 2 bit 3. The receive section operation is conditioned by the contents of the TRCS and RMC register. In the normal non-biphase mode, the received bit stream is synchronized by the first "0" (space). During 10-bit time, the approximate center is strobed. If the tenth bit is not "1" (stop bit), the system assumes a framing error and the ORFE is set.

If the tenth bit is "1", the data is transferred to the receive data register, with the interrupt flag set. If the tenth bit of the next data is received, however, still RDRF is preserved set, then ORFE is set indicating that an overrun error has occurred.

After the MCU read of the status register as a response to RDRF flag or ORFE flag, following the MCU read of the receive data register, RDRF or ORFE will be cleared.

■ RAM CONTROL REGISTER

The register assigned to the address $0014 gives a status information about standby RAM.

RAM Control Register

	7	6	5	4	3	2	1	0
$0014	STBY PWR	RAME	x	x	x	x	x	x

Bit 0 Not used.
Bit 1 Not used.
Bit 2 Not used.
Bit 3 Not used.
Bit 4 Not used.
Bit 5 Not used.
Bit 6 RAM Enable.

Using this control bit, the user can disable the RAM. When the MPU is reset, "1" is set in the RAM Enable bit thus enabling the standby RAM. With the program control, it is capable of writing "1" or "0". With the disabled RAM (logic "0"), the RAM address becomes external address and the MPU may read the data from the outside memory.

Bit 7 Standby Bit

This bit is cleared when the V_{CC} voltage is removed. This bit is a read/write status flag that user can read. If this bit is preserved set, indicating that V_{CC} voltage is applied and the data in the RAM is valid.

■ GENERAL DESCRIPTION OF INSTRUCTION SET

The HD6301V0 has an upward object code compatible with the HD6801 to utilize all instruction sets of the HMCS6800. The execution time of the key instruction is reduced to increase the system through-put. In addition, the bit operation instruction, the change instruction of the index and the accumulator, the sleep instruction are added. This section describes:

- MCU programming model (See Fig. 23)
- Addressing modes
- Accumulator and memory manipulation instructions (See Table 8)
- New instructions
- Index register and stack manipulation instructions (See Table 9)
- Jump and branch instructions (See Table 10)
- Condition code register manipulation instructions (See Table 11)
- Op-code map (See Table 12)

• MCU Programming Model

The programming model for the HD6301V0 is shown in Figure 23. The double accumulator is physically the same as the accumulator A concatenated with the accumulator B, so that the contents of A and B is changed with executing operation of an accumulator D.

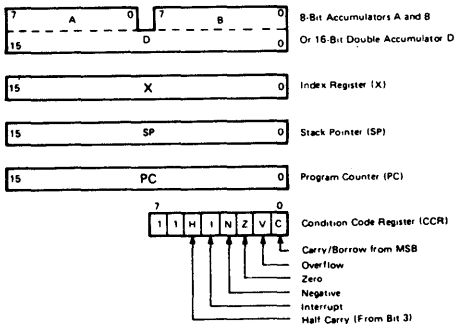


Figure 23 MCU Programming Model

• MCU Addressing Modes

The HD6301V0 has seven address modes which depend on both of the instruction type and the code. The address mode for every instruction is shown along with execution time given in terms of machine cycles (Table 8 to 12). When the clock frequency is 4 MHz, the machine cycle will be microseconds.

Accumulator (ACCX) Addressing

Only the accumulator (A or B) is addressed. Either accumulator A or B is specified by one-byte instructions.

Immediate Addressing

In this mode, the operand is stored in the second byte of the instruction except that the operand in LDS and LDX, etc are stored in the second and the third byte. These are two or three-byte instructions.

Direct Addressing

In this mode, the second byte of instruction indicates the address where the operand is stored. Direct addressing allows the user to directly address the lowest 256 Bytes in the machine ie; locations zero through 255. Enhanced execution times are achieved by storing data in these locations. For system configuration, it is recommended that these locations should be RAM and be utilized preferably for user's data realm. These are two-byte instructions except the AIM, OIM, EIM and TIM each have three.

Extended Addressing

In this mode, the second byte indicates the upper 8 bit addresses where the operand is stored, while the third byte indicates the lower 8 bits. This is an absolute address in memory. These are three-byte instructions.

Indexed Addressing

In this mode, the contents of the second byte is added to the lower 8 bits in the Index Register. For each of AIM, OIM, EIM and TIM instructions, the contents of the third byte are added to the lower 8 bits in the Index Register. In addition, this carry is added to the upper 8 bits in the Index Register. The result is used for addressing memory. Because the modified address is held in the Temporary Address Register, there is no change to the Index Register. These are two-byte instructions but AIM, OIM, EIM, TIM have three.

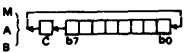
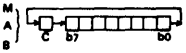
Implied Addressing

In this mode, the instruction itself gives the address; stack pointer, index register, etc. These are 1-byte instructions.

Relative Addressing

In this mode, the contents of the second byte is added to the lower 8 bits in the program counter. The resulting carry or borrow is added to the upper 8 bits. This helps the user to address the data within a range of −126 to +129 bytes of the current execution instruction. These are two-byte instructions.

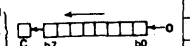
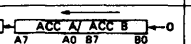
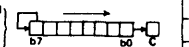
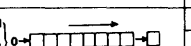
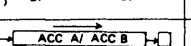
Table 8 Accumulator, Memory Manipulation Instructions

Operations	Mnemonic	Addressing Modes															Boolean/ Arithmetic Operation	Condition Code Register					
		IMMED			DIRECT			INDEX			EXTEND			IMPLIED				5	4	3	2	1	0
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#		H	I	N	Z	V	C
Add	ADDA	8B	2	2	9B	3	2	AB	4	2	BB	4	3				A + M→ A	↕	•	↕	↕	↕	↕
	ADDB	CB	2	2	DB	3	2	EB	4	2	FB	4	3				B + M→ B	↕	•	↕	↕	↕	↕
Add Double	ADDD	C3	3	3	D3	4	2	E3	5	2	F3	5	3				A : B + M : M + 1 → A : B	•	•	↕	↕	↕	↕
Add Accumulators	ABA													1B	1	1	A + B→ A	↕	•	↕	↕	↕	↕
Add With Carry	ADCA	89	2	2	99	3	2	A9	4	2	B9	4	3				A + M + C→ A	↕	•	↕	↕	↕	↕
	ADCB	C9	2	2	D9	3	2	E9	4	2	F9	4	3				B + M + C→ B	↕	•	↕	↕	↕	↕
AND	ANDA	84	2	2	94	3	2	A4	4	2	B4	4	3				A·M→ A	•	•	↕	↕	R	•
	ANDB	C4	2	2	D4	3	2	E4	4	2	F4	4	3				B·M→ B	•	•	↕	↕	R	•
Bit Test	BIT A	85	2	2	95	3	2	A5	4	2	B5	4	3				A·M	•	•	↕	↕	R	•
	BIT B	C5	2	2	D5	3	2	E5	4	2	F5	4	3				B·M	•	•	↕	↕	R	•
Clear	CLR							6F	5	2	7F	5	3				00 → M	•	•	R	S	R	R
	CLRA													4F	1	1	00 → A	•	•	R	S	R	R
	CLRB													5F	1	1	00 → B	•	•	R	S	R	R
Compare	CMPA	81	2	2	91	3	2	A1	4	2	B1	4	3				A - M	•	•	↕	↕	↕	↕
	CMPB	C1	2	2	D1	3	2	E1	4	2	F1	4	3				B - M	•	•	↕	↕	↕	↕
Compare Accumulators	CBA													11	1	1	A - B	•	•	↕	↕	↕	↕
Complement, 1's	COM							63	6	2	73	6	3				$\overline{M}$ → M	•	•	↕	↕	R	S
	COMA													43	1	1	$\overline{A}$ → A	•	•	↕	↕	R	S
	COMB													53	1	1	$\overline{B}$ → B	•	•	↕	↕	R	S
Complement, 2's	NEG							60	6	2	70	6	3				00 - M → M	•	•	↕	↕	①	②
(Negate)	NEGA													40	1	1	00 - A → A	•	•	↕	↕	①	②
	NEGB													50	1	1	00 - B → B	•	•	↕	↕	①	②
Decimal Adjust, A	DAA													19	1	1	Converts binary add of BCD characters into BCD format	•	•	↕	↕	↕	③
Decrement	DEC							6A	6	2	7A	6	3				M - 1 → M	•	•	↕	↕	④	•
	DECA													4A	1	1	A - 1 → A	•	•	↕	↕	④	•
	DECB													5A	1	1	B - 1 → B	•	•	↕	↕	④	•
Exclusive OR	EORA	88	2	2	98	3	2	A8	4	2	B8	4	3				A ⊕ M→ A	•	•	↕	↕	R	•
	EORB	C8	2	2	D8	3	2	E8	4	2	F8	4	3				B ⊕ M→ B	•	•	↕	↕	R	•
Increment	INC							6C	6	2	7C	6	3				M + 1 → M	•	•	↕	↕	⑤	•
	INCA													4C	1	1	A + 1 → A	•	•	↕	↕	⑤	•
	INCB													5C	1	1	B + 1 → B	•	•	↕	↕	⑤	•
Load Accumulator	LDAA	86	2	2	96	3	2	A6	4	2	B6	4	3				M → A	•	•	↕	↕	R	•
	LDAB	C6	2	2	D6	3	2	E6	4	2	F6	4	3				M → B	•	•	↕	↕	R	•
Load Double Accumulator	LDD	CC	3	3	DC	4	2	EC	5	2	FC	5	3				M + 1 → B, M → A	•	•	↕	↕	R	•
Multiply Unsigned	MUL													3D	7	1	A x B → A : B	•	•	•	•	•	⑪
OR, Inclusive	ORAA	8A	2	2	9A	3	2	AA	4	2	BA	4	3				A + M→ A	•	•	↕	↕	R	•
	ORAB	CA	2	2	DA	3	2	EA	4	2	FA	4	3				B + M→ B	•	•	↕	↕	R	•
Push Data	PSHA													36	4	1	A → Msp, SP - 1 → SP	•	•	•	•	•	•
	PSHB													37	4	1	B → Msp, SP - 1 → SP	•	•	•	•	•	•
Pull Data	PULA													32	3	1	SP + 1 → SP, Msp → A	•	•	•	•	•	•
	PULB													33	3	1	SP + 1 → SP, Msp → B	•	•	•	•	•	•
Rotate Left	ROL							69	6	2	79	6	3				M, A, B 	•	•	↕	↕	⑥	↕
	ROLA													49	1	1		•	•	↕	↕	⑥	↕
	ROLB													59	1	1		•	•	↕	↕	⑥	↕
Rotate Right	ROR							66	6	2	76	6	3				M, A, B 	•	•	↕	↕	⑥	↕
	RORA													46	1	1		•	•	↕	↕	⑥	↕
	RORB													56	1	1		•	•	↕	↕	⑥	↕

Note) Condition Code Register will be explained in Note of Table 11.

(to be continued)

Table 8 Accumulator, Memory Manipulation Instructions

Operations	Mnemonic	Addressing Modes															Boolean/ Arithmetic Operation	Condition Code Register					
		IMMED			DIRECT			INDEX			EXTEND			IMPLIED				5	4	3	2	1	0
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#		H	I	N	Z	V	C
Shift Left Arithmetic	ASL							68	6	2	78	6	3				M, A, B 	•	•	↕	↕	⑥	↕
	ASLA													48	1	1		•	•	↕	↕	⑥	↕
	ASLB													58	1	1		•	•	↕	↕	⑥	↕
Double Shift Left, Arithmetic	ASLD													05	1	1		•	•	↕	↕	⑥	↕
Shift Right Arithmetic	ASR							67	6	2	77	6	3				M, A, B 	•	•	↕	↕	⑥	↕
	ASRA													47	1	1		•	•	↕	↕	⑥	↕
	ASRB													57	1	1		•	•	↕	↕	⑥	↕
Shift Right Logical	LSR							64	6	2	74	6	3				M, A, B 	•	•	R	↕	⑥	↕
	LSRA													44	1	1		•	•	R	↕	⑥	↕
	LSRB													54	1	1		•	•	R	↕	⑥	↕
Double Shift Right Logical	LSRD													04	1	1		•	•	R	↕	⑥	↕
Store Accumulator	STAA				97	3	2	A7	4	2	B7	4	3				A → M	•	•	↕	↕	R	•
	STAB				D7	3	2	E7	4	2	F7	4	3				B → M	•	•	↕	↕	R	•
Store Double Accumulator	STD				DD	4	2	ED	5	2	FD	5	3				A → M B → M + 1	•	•	↕	↕	R	•
Subtract	SUBA	80	2	2	90	3	2	A0	4	2	B0	4	3				A - M → A	•	•	↕	↕	↕	↕
	SUBB	C0	2	2	D0	3	2	E0	4	2	F0	4	3				B - M → B	•	•	↕	↕	↕	↕
Double Subtract	SUBD	83	3	3	93	4	2	A3	5	2	B3	5	3				A : B - M : M + 1 → A : B	•	•	↕	↕	↕	↕
Subtract Accumulators	SBA													10	1	1	A - B → A	•	•	↕	↕	↕	↕
Subtract With Carry	SBCA	82	2	2	92	3	2	A2	4	2	B2	4	3				A - M - C → A	•	•	↕	↕	↕	↕
	SBCB	C2	2	2	D2	3	2	E2	4	2	F2	4	3				B - M - C → B	•	•	↕	↕	↕	↕
Transfer Accumulators	TAB													16	1	1	A → B	•	•	↕	↕	R	•
	TBA													17	1	1	B → A	•	•	↕	↕	R	•
Test Zero or Minus	TST							6D	4	2	7D	4	3				M - 00	•	•	↕	↕	R	R
	TSTA													4D	1	1	A - 00	•	•	↕	↕	R	R
	TSTB													5D	1	1	B - 00	•	•	↕	↕	R	R
And Immediate	AIM				71	6	3	61	7	3							M·IMM→M	•	•	↕	↕	R	•
OR Immediate	OIM				72	6	3	62	7	3							M+IMM→M	•	•	↕	↕	R	•
EOR Immediate	EIM				75	6	3	65	7	3							M⊕IMM→M	•	•	↕	↕	R	•
Test Immediate	TIM				7B	4	3	6B	5	3							M·IMM	•	•	↕	↕	R	•

Note) Condition Code Register will be explained in Note of Table 11.

- **New Instructions**

In addition to the HD6801 Instruction Set, the HD6301V0 has the following new instructions:

AIM----(M) · (IMM) → (M)
Evaluates the AND of the immediate data and the memory, places the result in the memory.

OIM----(M) + (IMM) → (M)
Evaluates the OR of the immediate data and the memory, places the result in the memory.

EIM----(M) ⊕ (IMM) → (M)
Evaluates the EOR of the immediate data and the contents of memory, places the result in memory.

TIM----(M) · (IMM)
Evaluates the AND of the immediate data and the memory, changes the flag of associated condition code register

Each instruction has three bytes; the first is op-code, the second is immediate data, the third is address modifier.

XGDX--(ACCD) ↔ (IX)
Exchanges the contents of accumulator and the index register.

SLP----The MPU is brought to the sleep mode. For sleep mode, see the "sleep mode" section.

Table 9 Index Register, Stack Manipulation Instructions

Pointer Operations	Mnemonic	Addressing Modes														Boolean/ Arithmetic Operation	Condition Code Register					
		IMMED			DIRECT			INDEX			EXTEND			IMPLIED				5	4	3	2	1	0
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#		H	I	N	Z	V	C
Compare Index Reg	CPX	8C	3	3	9C	4	2	AC	5	2	BC	5	3				X−M:M+1	•	•	↕	↕	↕	↕
Decrement Index Reg	DEX													09	1	1	X − 1 → X	•	•	•	↕	•	•
Decrement Stack Pntr	DES													34	1	1	SP − 1 → SP	•	•	•	•	•	•
Increment Index Reg	INX													08	1	1	X + 1 → X	•	•	•	↕	•	•
Increment Stack Pntr	INS													31	1	1	SP + 1 → SP	•	•	•	•	•	•
Load Index Reg	LDX	CE	3	3	DE	4	2	EE	5	2	FE	5	3				M → X_H, (M + 1) → X_L	•	•	⑦	↕	R	•
Load Stack Pntr	LDS	8E	3	3	9E	4	2	AE	5	2	BE	5	3				M → SP_H, (M+1) → SP_L	•	•	⑦	↕	R	•
Store Index Reg	STX				DF	4	2	EF	5	2	FF	5	3				X_H → M, X_L → (M + 1)	•	•	⑦	↕	R	•
Store Stack Pntr	STS				9F	4	2	AF	5	2	BF	5	3				SP_H → M, SP_L → (M + 1)	•	•	⑦	↕	R	•
Index Reg → Stack Pntr	TXS													35	1	1	X − 1 → SP	•	•	•	•	•	•
Stack Pntr → Index Reg	TSX													30	1	1	SP + 1 → X	•	•	•	•	•	•
Add	ABX													3A	1	1	B + X → X	•	•	•	•	•	•
Push Data	PSHX													3C	5	1	X_L → M_{sp}, SP − 1 → SP X_H → M_{sp}, SP − 1 → SP	•	•	•	•	•	•
Pull Data	PULX													38	4	1	SP + 1 → SP, M_{sp} → X_H SP + 1 → SP, M_{sp} → X_L	•	•	•	•	•	•
Exchange	XGDX													18	2	1	ACCD↔IX	•	•	•	•	•	•

Note) Condition Code Register will be explained in Note of Table 11.

Table 10 Jump, Branch Instruction

Operations	Mnemonic	Addressing Modes															Branch Test	Condition Code Register					
		RELATIVE			DIRECT			INDEX			EXTEND			IMPLIED				5	4	3	2	1	0
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#		H	I	N	Z	V	C
Branch Always	BRA	20	3	2													None	•	•	•	•	•	•
Branch Never	BRN	21	3	2													None	•	•	•	•	•	•
Branch If Carry Clear	BCC	24	3	2													C = 0	•	•	•	•	•	•
Branch If Carry Set	BCS	25	3	2													C = 1	•	•	•	•	•	•
Branch If = Zero	BEQ	27	3	2													Z = 1	•	•	•	•	•	•
Branch If ≥ Zero	BGE	2C	3	2													N ⊕ V = 0	•	•	•	•	•	•
Branch If > Zero	BGT	2E	3	2													Z + (N ⊕ V) = 0	•	•	•	•	•	•
Branch If Higher	BHI	22	3	2													C + Z = 0	•	•	•	•	•	•
Branch If ≤ Zero	BLE	2F	3	2													Z + (N ⊕ V) = 1	•	•	•	•	•	•
Branch If Lower Or Same	BLS	23	3	2													C + Z = 1	•	•	•	•	•	•
Branch If < Zero	BLT	2D	3	2													N ⊕ V = 1	•	•	•	•	•	•
Branch If Minus	BMI	2B	3	2													N = 1	•	•	•	•	•	•
Branch If Not Equal Zero	BNE	26	3	2													Z = 0	•	•	•	•	•	•
Branch If Overflow Clear	BVC	28	3	2													V = 0	•	•	•	•	•	•
Branch If Overflow Set	BVS	29	3	2													V = 1	•	•	•	•	•	•
Branch If Plus	BPL	2A	3	2													N = 0	•	•	•	•	•	•
Branch To Subroutine	BSR	8D	5	2													See Special Operations	•	•	•	•	•	•
Jump	JMP							6E	3	2	7E	3	3					•	•	•	•	•	•
Jump To Subroutine	JSR				9D	5	2	AD	5	2	BD	6	3					•	•	•	•	•	•
No Operation	NOP													01	1	1	Advances Prog. Cntr. Only	•	•	•	•	•	•
Return From Interrupt	RTI													3B	10	1	See Special Operations	——— ⑧ ———					
Return From Subroutine	RTS													39	5	1		•	•	•	•	•	•
Software Interrupt	SWI													3F	12	1		•	S	•	•	•	•
Wait for Interrupt*	WAI													3E	9	1		•	⑨	•	•	•	•
Sleep	SLP													1A	4	1		•	•	•	•	•	•

Note) *WAI puts R/$\overline{W}$ high; Address Bus goes to FFFF; Data Bus goes to the three state level.
Condition Register will be explained in Note of Table 11.

Table 11 Condition Code Register Manipulation Instructions

Operations	Mnemonic	Addressing Modes IMPLIED OP	~	#	Boolean Operation	Condition Code Register 5 H	4 I	3 N	2 Z	1 V	0 C
Clear Carry	CLC	0C	1	1	0 → C	•	•	•	•	•	R
Clear Interrupt Mask	CLI	0E	1	1	0 → I	•	R	•	•	•	•
Clear Overflow	CLV	0A	1	1	0 → V	•	•	•	•	R	•
Set Carry	SEC	0D	1	1	1 → C	•	•	•	•	•	S
Set Interrupt Mask	SEI	0F	1	1	1 → I	•	S	•	•	•	•
Set Overflow	SEV	0B	1	1	1 → V	•	•	•	•	S	•
Accumulator A → CCR	TAP	06	1	1	A → CCR	⑩					
CCR → Accumulator A	TPA	07	1	1	CCR → A	•	•	•	•	•	•

[NOTE] Condition Code Register Notes: (Bit set if test is true and cleared otherwise)

① (Bit V) Test: Result = 10000000?
② (Bit C) Test: Result ≠ 00000000?
③ (Bit C) Test: BCD Character of high-order byte greater than 10? (Not cleared if previously set)
④ (Bit V) Test: Operand = 10000000 prior to execution?
⑤ (Bit V) Test: Operand = 01111111 prior to execution?
⑥ (Bit N) Test: Set equal to N⊕C=1 after the execution of instructions
⑦ (Bit N) Test: Result less than zero? (Bit 15=1)
⑧ (All) Load Condition Code Register from Stack.
⑨ (Bit I) Set when interrupt occurs. If previously set, a Non-Maskable Interrupt is required to exist the wait state.
⑩ (All Bit) Set according to the contents of Accumulator A.
⑪ (Bit C) Result of Multiplication Bit 7=1? (ACCB)

Table 12 OP-Code Map

OP CODE HI	LO	0000 0	0001 1	0010 2	0011 3	ACC A 0100 4	ACC B 0101 5	IND 0110 6	EXT/DIR* 0111 7	ACCA or SP IMM 1000 8	DIR 1001 9	IND 1010 A	EXT 1011 B	ACCB or X IMM 1100 C	DIR 1101 D	IND 1110 E	EXT 1111 F	
0000	0		SBA	BRA	TSX	NEG				SUB								0
0001	1	NOP	CBA	BRN	INS			AIM		CMP								1
0010	2			BHI	PULA			OIM		SBC								2
0011	3			BLS	PULB	COM				SUBD			ADDD					3
0100	4	LSRD		BCC	DES	LSR				AND								4
0101	5	ASLD		BCS	TXS			EIM		BIT								5
0110	6	TAP	TAB	BNE	PSHA	ROR				LDA								6
0111	7	TPA	TBA	BEQ	PSHB	ASR					STA				STA			7
1000	8	INX	XGDX	BVC	PULX	ASL				EOR								8
1001	9	DEX	DAA	BVS	RTS	ROL				ADC								9
1010	A	CLV	SLP	BPL	ABX	DEC				ORA								A
1011	B	SEV	ABA	BMI	RTI			TIM		ADD								B
1100	C	CLC		BGE	PSHX	INC				CPX				LDD				C
1101	D	SEC		BLT	MUL	TST				BSR	JSR				STD			D
1110	E	CLI		BGT	WAI			JMP		LDS				LDX				E
1111	F	SEI		BLE	SWI	CLR					STS				STX			F
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	

UNDEFINED OP CODE

* Only each instructions of AIM, OIM, EIM, TIM

■ LOW POWER CONSUMPTION MODE

The HD6301V0 has two low power consumption modes; sleep and standby mode

• Sleep Mode

On execution of SLP instruction, the MCU is brought to the sleep mode. In the sleep mode, the MPU sleeps (the MPU clock becomes inactive), but the contents of the register in the MPU is secured. In this mode, the peripherals of MPU will remain operational. So the operations such as transmit and receive of the SCI data and counter may keep on functioning. In this mode, the power consumption is reduced to about 1/10 the value of a normal operation.

The escape from this mode can be done by interrupt, $\overline{RES}$, $\overline{STBY}$. The $\overline{RES}$ resets the MCU and the $\overline{STBY}$ brings it into the standby mode (This will be mentioned later). When interrupt is requested to the MPU and accepted, the sleep mode escapes, then the MPU is brought to the operation mode and vectors to the interrupt routine. When the MPU has masked the interrupt, after releasing from the sleep mode, the next instruction of

sleep starts to execute. However, in such a case that the timer interrupt is inhibited on the timer side, the sleep mode cannot be released due to the absence of the interrupt request to the MPU.

This sleep mode is available to reduce an average power consumption in the applications of the HD6301V0 which may not always drive.

- **Standby Mode**

Bringing $\overline{STBY}$ "Low", the MPU becomes reset with all clocks of the HD6301V0 inactive and goes into the standby mode. This mode remarkably reduces the power consumptions of the HD6301V0.

In the standby mode, the HD6301V0 is continuously supplied with power so the contents of RAM is retained. The standby mode should escape by the reset start. The following is the typical application of this mode.

First, $\overline{NMI}$ routine stacks the MCU's internal information and the contents of SP in RAM, disables RAME bit of RAM control register, sets the STBY bit, and then goes into the standby mode. If the STBY bit keeps set on reset start, it means that the power supply and the contents of RAM is normally guaranteed. The system recovery may be possible by returning SP and bringing into the condition before the standby mode has started. The timing relation for each line in this application is shown in Figure 24.

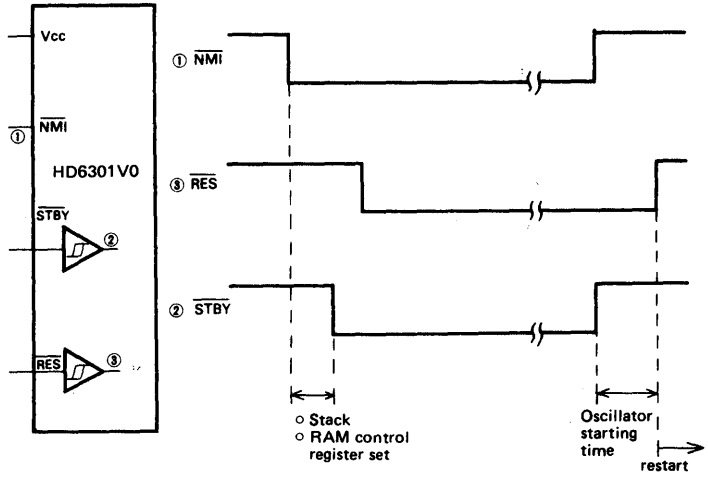


Figure 24 Standby Mode Timing

■ ERROR PROCESSING

When the HD6301V0 fetches an undefined instruction or fetches an instruction from nonresident memory area, it generates the most precedent internal interrupt, that may protect the system from system burst due to noise or a program error.

- **Op-Code Error**

Fetching an undefined op-code, the HD6301V0 will stack the MPU register as in the case of a normal interrupt and vector to the TRAP ($FFEE, $FFEF), that has a second highest priority ($\overline{RES}$ is the highest).

- **Address Error**

When an instruction is fetched from other than a resident ROM, RAM, or an external memory area, the MPU starts the same interrupt as op-code error. In case where the instruction is fetched from external memory area of non-resident memory, it cannot function.

The addresses which cause address error in particular mode are as shown in Table 13.

This feature is applicable only to the instruction fetch, not to normal read/write of data accessing.

Table 13 Address Error

Mode	0	1	4	5	6	7
Address	$0000 ↓ $001F	$0000 ↓ $001F	$0000 ↓ $001F	$0000 ↓ $007F $0200 ↓ $EFFF	$0000 ↓ $001F	$0000 ↓ $007F $0100 ↓ $EFFF

Transitions among the active mode, sleep mode, standby mode and reset are shown in Fig. 25. Figures 26, 27, 28 and 29 shows a system configuration.

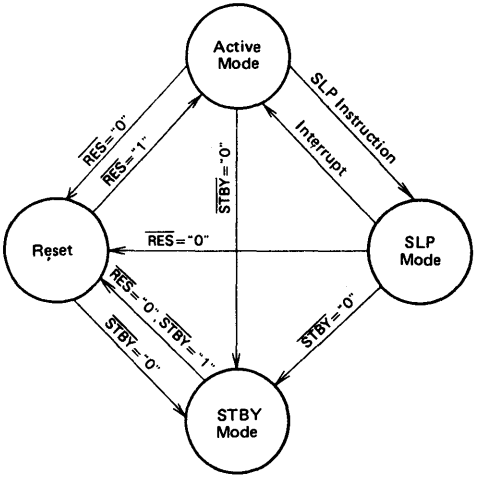


Figure 25 Transitions among Active Mode, Standby Mode Sleep Mode, and Reset

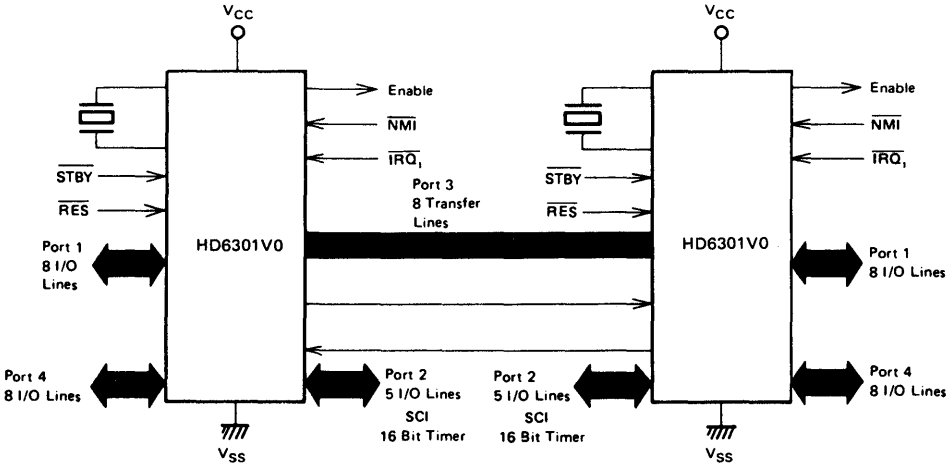


Figure 26 HD6301V0 MCU Single-Chip Dual Processor Configuration

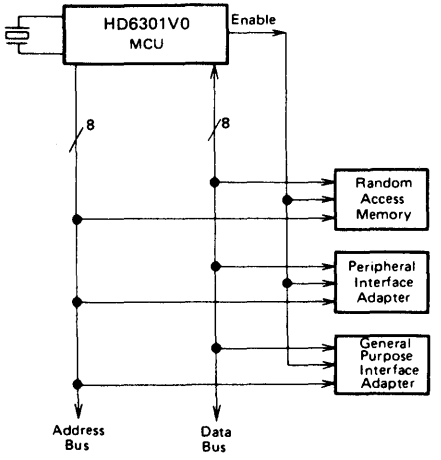


Figure 27 HD6301V0 MCU Expanded Non-Multiplexed Mode (Mode 5)

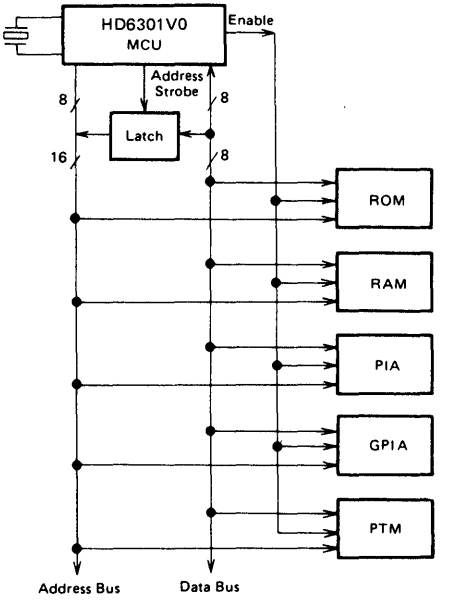


Figure 28 HD6301V0 MCU Expanded Multiplexed Mode

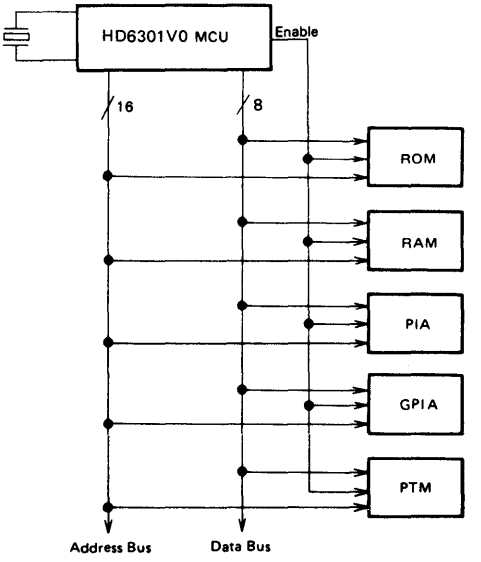


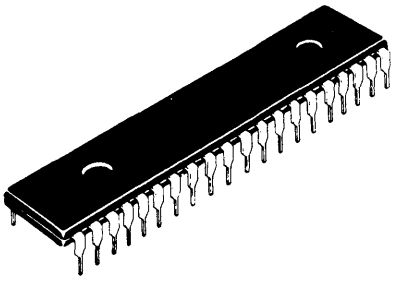
Figure 29 HD6301V0 MCU Expanded Non-Multiplexed Mode (Mode 1)

HD6303,HD63A03,HD63B03 CMOS MPU (Micro Processing Unit)

—ADVANCE INFORMATION—

The HD6303 is an 8-bit CMOS micro processing unit which has the completely compatible instruction set with the HD6301V0. 128 bytes RAM, Serial Communication Interface (SCI), parallel I/O terminals as well as three functions of timer on-chip are incorporated in the HD6303. It is bus compatible with HMCS6800 and can be expanded up to 65k words. Like the HMCS6800 family, I/O level is TTL compatible with +5.0V single power supply. As the HD6303 is CMOS MPU, power dissipation is extremely low. And also Sleep Mode and Stand-By Mode which the HD6303 has for low power dissipation make lower power application possible.

HD6303P
HD63A03P
HD63B03P

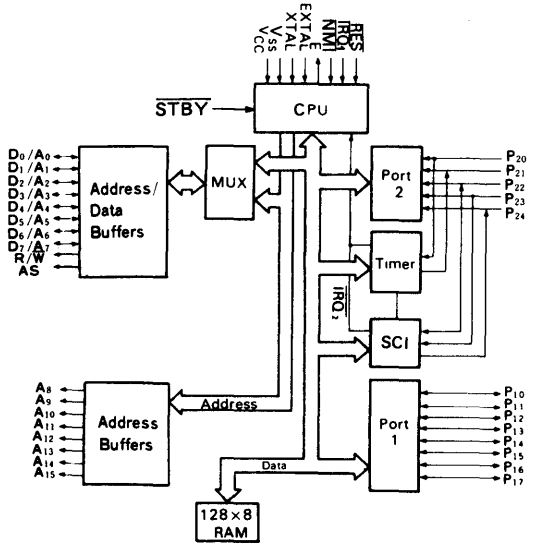


(DP-40)

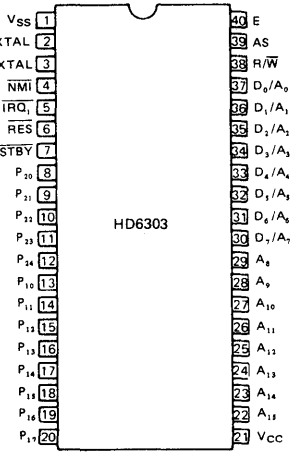
■ FEATURES

- Object Code Upward Compatible with the HD6800, HD6802, HD6801
- Multiplexed Bus (D_0~D_7/A_0~A_7)
- Abundant On-Chip Functions Compatible with the HD6301V0; 128 Bytes RAM, 13 Parallel I/O Lines (including Timer, SCI I/O Terminals), 16-bit Timer, Serial Communication Interface (SCI)
- Low Power Consumption Mode; Sleep Mode, Stand-By Mode
- Minimum Instruction Cycle Time
  1μs (f=1MHz), 0.67μs (f=1.5MHz), 0.5μs (f=2.0MHz)
- Bit Manipulation, Bit Test Instruction
- Error Detecting Function; Address Trap, Op Code Trap
- Up to 65k Words Address Space

■ BLOCK DIAGRAM



■ PIN ARRANGEMENT

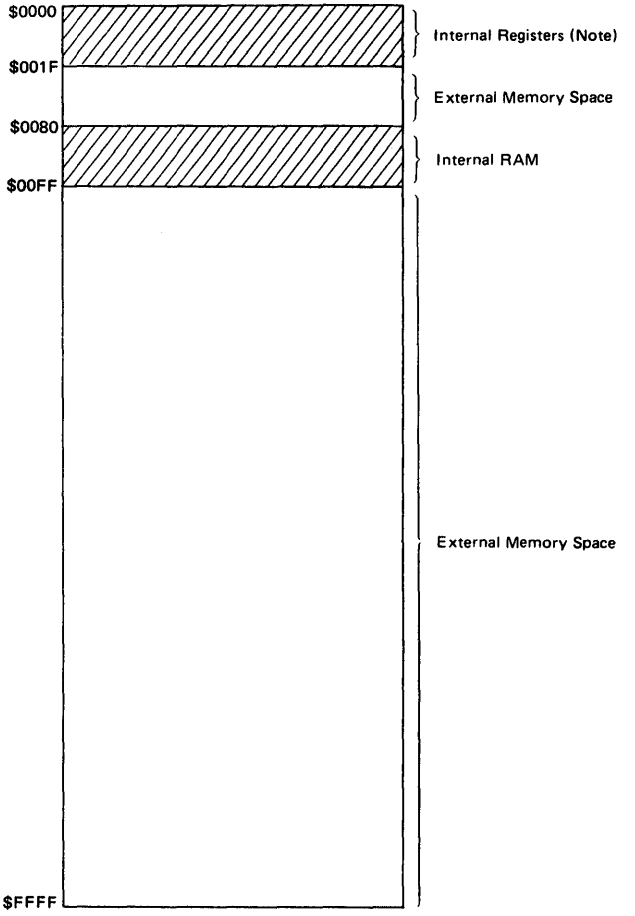


(Top View)

■ TYPE OF PRODUCTS

Type No.	Bus Timing
HD6303	1.0 MHz
HD63A03	1.5 MHz
HD63B03	2.0 MHz

■ MEMORY MAP



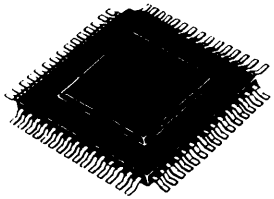
(Note) $04, $05, $06, $07, $0F are external address.

HD63L05
CMOS MCU (Microcomputer Unit)

—PRELIMINARY—

The HD63L05 is a CMOS single-chip microcomputer suitable for low-voltage and low-current operation. Having CPU functions similar to those of the HMCS6800 family, the HD63L05 is equipped with a 4k bytes ROM, 96 bytes RAM, I/O, timer, 8 bits A/D, and LCD (6 x 7 segments) drivers, all on one chip.

HD63L05



(FP-60A)

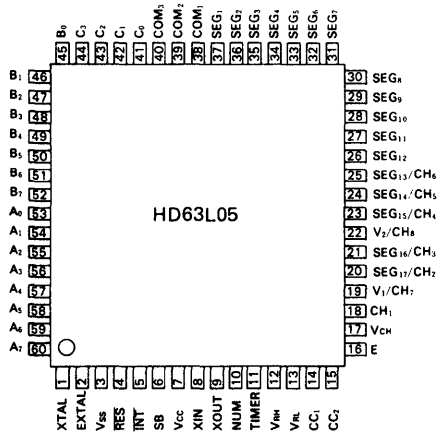
■ HARDWARE FEATURES

- 3V Power Supply
- 8-Bit Architecture
- Built-in 4k Bytes ROM (Mask ROM)
- Built-in 96 Bytes RAM
- 20 Parallel I/O Ports
- Built-in 6 x 7 Segments LCD Driver Capability
- Built-in 8-Bit Timer
- Built-in 8-Bit A/D Converter
- Program Halt Function for Low Power Dissipation
- Stand-by Input Terminal for Data Holding

■ SOFTWARE FEATURES

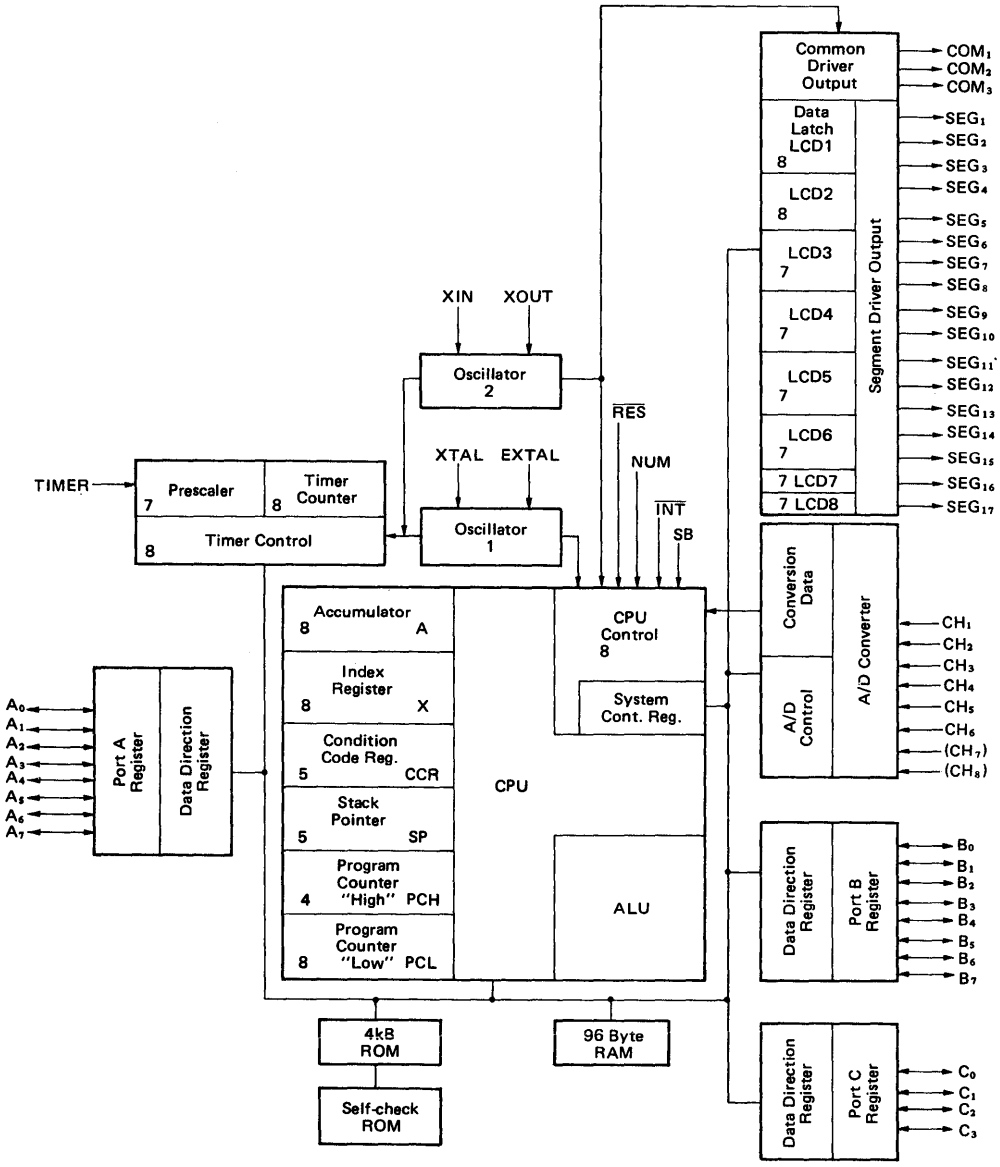
- An Instruction Set Similar to That of The HMCS6800 Family (Compatible with The HD6805S)
- HMCS6800 Family Software Development System Is Applicable

■ PIN ARRANGEMENT



(Top View)

■ BLOCK DIAGRAM



■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC}	-0.3 ~ +5.5	V
Input Voltage	V_{in}	-0.3 ~ V_{CC} +0.3	V
Output Voltage	V_{out}	-0.3 ~ V_{CC} +0.3	V
Operating Temperature	T_{opr}	-20 ~ +75	°C
Storage Temperature	T_{stg}	-55 ~ +125	°C

(NOTE) If LSI's are used at rating exceeding the absolute maximum rating, they can be permanently destroyed.

■ ELECTRICAL CHARACTERISTICS

• DC CHARACTERISTICS (V_{CC} = 3.0V ±0.8V, V_{SS} = 0V, Ta = -20 ~ +75°C, typ means typical value at 3V unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Input "High" Voltage	$\overline{RES}$	V_{IH}		–	2.4	–	V
	$\overline{INT}$			–	2.4	–	V
	Others			–	2.2	–	V
	Timer Mode			–	2.2	–	V
	Self-check Mode			–	1.5	–	V
Input "Low" Voltage	$\overline{RES}$	V_{IL}		–	0.6	–	V
	$\overline{INT}$			–	0.6	–	V
	Others			–	0.8	–	V
Current Dissipation	During System Operation	I_{CC}		–	100	–	μA
	At Halt			–	40	–	μA
	At Stand-By			–	2	–	μA
	During A/D Operation			–	300	–	μA
Input Leakage Current	TIMER	I_{IL}	V_{in} = 0V ~ V_{CC}	–	0.1	–	μA
	$\overline{INT}$			–	0.1	–	μA

• AC CHARACTERISTICS (V_{CC} = 3.0V ±0.8V, V_{SS} = 0V, Ta = -20 ~ +75°C, typ means typical value at 3V unless otherwise noted.)

Item		Symbol	Test Conditions	min	typ	max	Unit
Operating Clock Frequency		f_{cl}		–	400	–	kHz
Cycle Time		t_{cyc}		–	10	–	μs
$\overline{INT}$ Pulse Width		t_{IWL}		–	t_{cyc} +1	–	μs
$\overline{RES}$ Pulse Width		t_{RWL}		–	t_{cyc} +1	–	μs
TIMER Pulse Width		t_{TWL}		–	t_{cyc} +1	–	μs
Oscillation Start Time (Crystal Option)		t_{oscf}	C_L = 10 pF ±20%	–	100	–	ms
Oscillation Start Time (32kHz)		t_{oscl}	C_G = 10 pF ±20%	–	1.0	–	s
Reset Delay Time		t_{RHL}	Ext. Capacitance = 2.2 μF	–	400	–	ms
Oscillation Frequency (Resistor Option)		f_{EXT}	R = 90 kΩ ±1%	–	400	–	kHz
Input Capacitance	EXTAL	C_{in}		–	10	–	pF
	XOUT			–	10	–	pF
	Others			–	5	–	pF

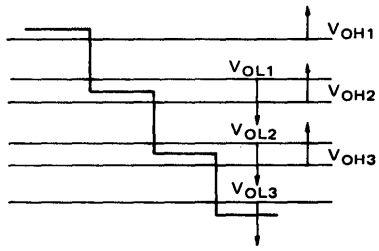
- **PORT CHARACTERISTICS (V_{CC} = 3.0V ±0.8V, V_{SS} = 0V, Ta = -20 ~ +75°C, typ means typical value at 3V unless otherwise noted.)**

Item		Symbol	Test Condition	min	typ	max	Unit
Output "High" Voltage	Port A, B, C	V_{OH}	I_{OH} = -100 μA	–	2.7	–	V
Output "Low" Voltage	Port A, B, C	V_{OL}	I_{OL} = 100 μA	–	0.3	–	V
Input "High" Voltage	Port A, B, C	V_{IH}		–	2.2	–	V
Input "Low" Voltage	Port A, B, C	V_{IL}		–	0.8	–	V
Input Leakage Current	Port A, B, C	I_{IH}	V_{in} = 3.0V	–	0.1	–	μA
Input Leakage Current	Port A, B, C	I_{IL}	V_{in} = 0V	–	0.1	–	μA
Input Leakage Current (Resistor Option)	Port A, B, C	I_{ILR}	V_{in} = 0V	–	20	–	μA

- **LCD DRIVER OUTPUT CHARACTERISTICS (V_{CC} = 3.0V, V_{SS} = 0V, Ta = -20 ~ +75°C, unless otherwise noted.)**

Item		Symbol	Test Condition	min	typ	max	Unit
Output "High" Voltage	Segment	V_{OH1}	I_{OH} = -1 μA	2.8	–	–	V
		V_{OH2}	I_{OH} = -1 μA	1.8	–	–	V
		V_{OH3}	I_{OH} = -1 μA	0.8	–	–	V
Output "Low" Voltage	Segment	V_{OL1}	I_{OL} = 1 μA	–	–	2.2	V
		V_{OL2}	I_{OL} = 1 μA	–	–	1.2	V
		V_{OL3}	I_{OL} = 1 μA	–	–	0.2	V
Output "High" Voltage	Common	V_{OH1}	I_{OH} = -5 μA	2.8	–	–	V
		V_{OH2}	I_{OH} = -5 μA	1.8	–	–	V
		V_{OH3}	I_{OH} = -5 μA	0.8	–	–	V
Output "Low" Voltage	Common	V_{OL1}	I_{OL} = 5 μA	–	–	2.2	V
		V_{OL2}	I_{OL} = 5 μA	–	–	1.2	V
		V_{OL3}	I_{OL} = 5 μA	–	–	0.2	V

(NOTE) V_{OH1} and V_{OL3} characteristics apply to the output obtained when segment terminals are used as output ports (I_{OH} = -30 μA, I_{OL} = 30 μA).



- **A/D CONVERTER CHARACTERISTICS (V_{CC} = 3.0V, V_{SS} = 0V, Ta = -20°C ~ +75°C, C = 300 pF, typ means typical value at 3V unless otherwise noted.)**

Item		Symbol	Test Condition	min	typ	max	Unit
Accuracy	Resolution			–	8	–	Bit
	Non-Linear Error			–	±1.5	–	LSB
Reference Voltage	"High" Side	V_{RH}		–	2.2	–	V
	"Low" Side	V_{RL}		–	0.2	–	V
	$V_{RH} - V_{RL}$	ΔV_{REF}		–	2.0	–	V
Input Voltage Range		V_{IN}		V_{RL}	–	V_{RH}	V
Conversion Time		t_{CNV}		2	–	4	ms

■ SIGNALS

The input and output signals of the MCU are described in the following:

- **V_{CC}, V_{SS}**
  Power is applied to the MCU at these two terminals. V_{CC} is a positive power input port and V_{SS} is grounded.

- **$\overline{INT}$**
  This terminal is used to envoke an external interruption to the MCU. For details, see the information given under the title, "Interruptions".

- **XTAL, EXTAL**
  These are control input ports to the built-in clock circuit. A crystal or resistor is connected to each of them depending on the degree of stability of the internal oscillation. For the method of using the input terminals, see the information, "Internal Oscillator Option".

- **XIN, XOUT**
  Connected to these terminals are crystals for the oscillator on the time base. A clock operation is possible by using a 32.768kHz crystal. For details, see "Internal Oscillator Option".

- **TIMER**
  An external input terminal at which the internal timer is counted down. For details, see the information, "Timer".

- **$\overline{RES}$**
  Used to reset the MCU. For details, see "Reset".

- **STANDBY**
  An external input terminal used halt all MCU operations and hold data. For details, see "Internal Oscillator Option".

- **A/D Input Terminals ($CH_1 \sim CH_8$)**
  Input terminals for analog voltages needed for A/D conversion. These may also be used as level check inputs under program control. For details, see the information, "A/D Converter".

- **V_{RH}, V_{RL}**
  Reference voltages for A/D conversion are applied to these two terminals. For details, see "A/D Converter".

- **CC_1, CC_2**
  Connected to CC_1 and CC_2 are A/D converter offset compensating capacitors. For details, see "A/D Converter".

- **NUM**
  This is not intended for user applications. Connect it to V_{CC}.

- **Input/Output Terminals ($A_0 \sim A_7$, $B_0 \sim B_7$, $C_0 \sim C_3$)**
  Each of these 20 terminals consists of two 8 bits ports and one 4 bits ports. It may be used as an input or output under program control of the data direction register. For details, see "Input/Output".

- **Liquid Crystal Driver Terminals ($COM_1 \sim COM_3$, $SEG_1 \sim SEG_{17}$)**
  These are 6 x 7 segments LCD terminals. $COM_1 \sim COM_3$ are for driving common electrodes, while $SEG_1 \sim SEG_{17}$ are for driving segments. They can be used as outputs only under program control. For details, see "LCD Circuit".

- **V_{CH}**
  Output Terminal from Internal Voltage Regulator (A capacitor is connected between V_{CH} and V_{CC}).

- **E**
  System Clock Output (Cycle clock).

- **V_1, V_2**
  These are terminals for LCD driver. Capacitors are connected between V_1, V_2 and V_{CC}.

■ MEMORY

The memory map of the MCU is shown in Figure 1. During processing of an interruption, the contents of the MCU registers are saved into the stack in the order shown in Figure 2. During saving, the stack pointer is decremented and the lower byte (PCL) of the program counter is the first to be stacked. Then the upper 4 bits (PCH) are stacked. For pulling, the saved contents are pulled in order while the stack pointer is being incremented. In the case of a subroutine call, the contents of only the program counters (PCH, PCL) are saved into the stack.

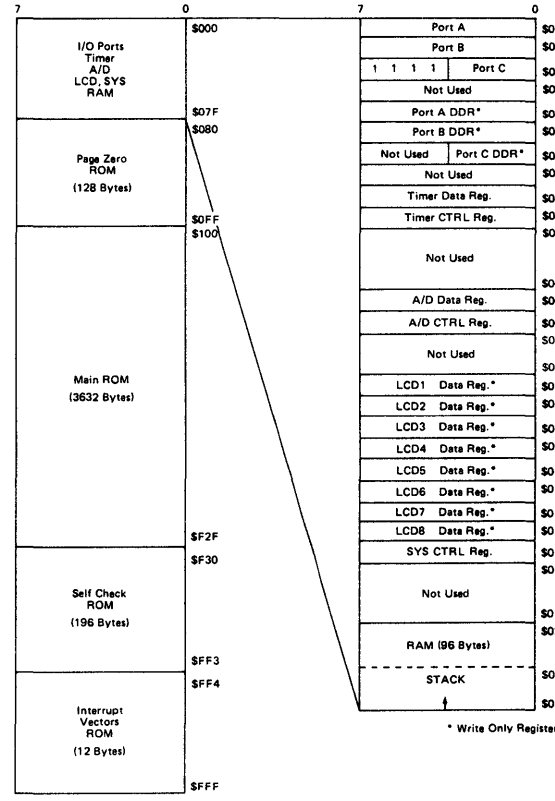


Figure 1 MCU Memory Map

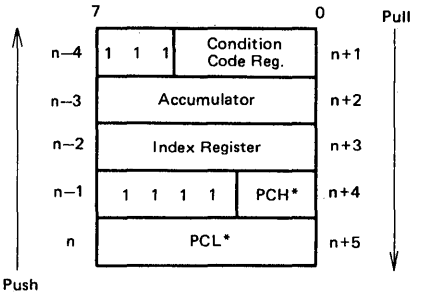


Figure 2 Interruption Stack Sequence

■ REGISTER

The CPU has five registers that can be operated by the programmer. They are shown in Figure 3.

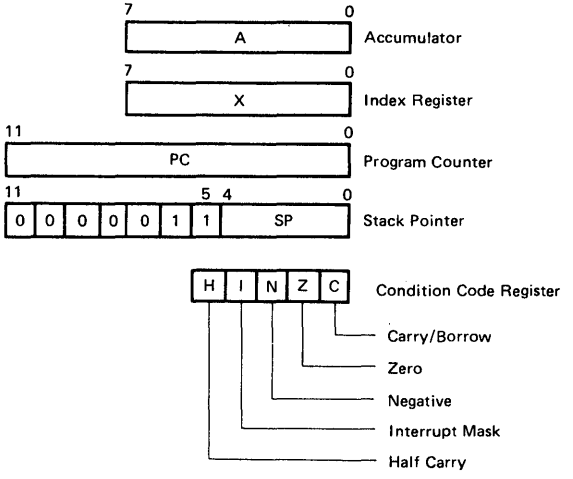


Figure 3 Programming Model

• Accumulator (A)

This accumulator is an ordinary 8 bits register. It is used to accumulate operands and the results of arithmetic operations or data processing.

• Index Register (X)

Being an 8 bits register, this index register is used for index addressing mode. The address contained in the register is composed of 8 bits. An execution address can be obtained by adding the 8 bits to an offset value.

The index register X may also be used for processing a limited range of data at a Read/Modify/Write instruction. When the register is not referenced to by the instruction being executed, it can be used as a temporary storage area.

• Program Counter (PC)

The program counter, a 12 bits register, contains the address of the next instruction to be executed.

• Stack Pointer (SP)

The stack pointer is a 12 bits register that indicates the address the next save space on the stack. At the beginning, the stack pointer is set at address $07F. It is decremented each time data is saved, and incremented each time data is reset. The upper 7 bits of the stack pointer are fixed to 0000011.

During MCU resetting or a reset stack pointer (RSP) instruction, the stack pointer is set to address $07F. Since a subroutine or interruption can use addresses up to $061 for saving, it is possible to call subroutines up to level 15.

• Condition Code Register (CC)

The condition code register is a 5 bits register, each bit showing the result of an instruction having just been executed. All bits can be tested by conditional Branch instructions. The five bits of the condition code register are used as follows:

Half Carry (H)

Used to indicate that a carry occurred between bits 3 and 4 during an arithmetic operation (ADD, ADC).

Interrupt (I)

When this bit is set, all of the timer, external ($\overline{INT}$), A/D and time base interruptions are masked. If an interruption occurs with this bit (I) set, the interruption information is held. It is processed immediately after the interruption mask bit (I) is reset.

Negative (N)

Used to indicate that the result of the most recent arithmetic operation, logical operation or data processing is negative (i.e., bit 7 being at logical "1").

Zero (Z)

Used to indicate that the result of the most recent arithmetic operation, logical operation or data processing is a zero.

Carry/Borrow (C)

Indicates the carry/borrow that occurred in the most recent arithmetic operation. This bit is affected by the Bit Test and Branch instruction, Shift instruction and Rotate instructions.

■ SYSTEM CONTROL REGISTER

Apart from the registers for program operation explained above, there is a register that controls system operation. Its configuration is shown in Figure 4.

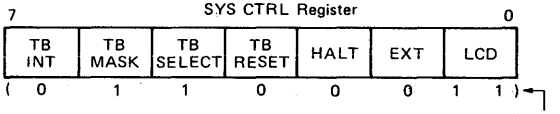


Figure 4 System Control Register Configuration

• Time Base Interruption Request Flag (TB INT)

Stores an interruption request flag from the time base which is selected by the TB select bit. If the TB mask or I (Interrupt Mask Bit in the CCR) is set, the Interruption Request Flag is not acknowledged.

• Time Base Interruption Mask (TB MASK)

If this bit is set, any interrupt request from the time base is not acknowledged.

- **Time Base Select Bit (TB SELECT)**

This bit selects the time base. In logical "1", an interruption from the 1-second cycle time base is acknowledged. In logical "0", 1/16-second cycle time base is acknowledged.

- **Time Base Reset Bit (TB RESET)**

This bit resets the frequency divider behind the 32kHz oscillator. When this bit is set, one shot reset pulse is generated by the hardware. Then, it resets the frequency divider and after that, the frequency divider restarts. As this bit has not a register, it indicates logical "0" to the CPU.

The frequency divider provides the system clocks to the A/D converter and LCD driver. So, it is need to pay an attention when "TB RESET" is used.

- **Halt (HALT)**

Used to halt the CPU, when this bit is set, the registers are saved into the stack in the same sequence as in interruption processing. After all registers have been saved, the CPU halts and is wait-for-interrupt state.

If the bit is reset by an external interruption or time base interruption, the CPU restarts operating. A combined use of the Halt and Time Base Interruption functions permits the CPU to operate intermittently.

- **EXT**

Used to switch the hardware configuration for expanded LCD capabilities. Normally, it is reset.

- **Duty Select Bit (DUTY)**

The LCD drive signal is based on 1/3 bias – 1/3 duty. However, there are switching circuits built in for expanded LCD capabilities and output only ports. For details, see the information given in "LCD Circuit".

■ TIMER

Figure 5 shows a block diagram of the MCU timer. This 8 bits counters is loaded under program control. It starts counting-down immediately after clock inputs are applied. When the timer count comes to zero, the timer interruption request bit (bit 7) in the timer control register is set. In response to the interruption request, the MCU saves its contents into the stack.

Then it fetches a timer interruption vector and executes an interruption routine. Any timer interruption can be masked by setting the timer mask bit (bit 6) within the timer control register. The interruption mask bit (I) within the condition code register also inhibits a timer interruption.

Clock inputs to the timer may be the input signal that is applied from an external source to the timer input terminal, or the clock signal within the MCU. If the internal clock signal is used as the source, the clock input is gated by the input applied to the timer input terminal; this permits easy measurement of its pulse width. Also, there are two types of internal clock signals within the MCU to allow timer operation when the CPU is halted. These clock signals are under program control.

A 7 bits prescaler is provided to increase the timer's timing interval. The number of bits of the prescaler can be program controlled by the lower 3 bits within the timer control register. If the count comes to below zero, the timer continues counting; the count below zero can be monitored anytime by reading the timer data register, without disturbing the contents of the counter.

At the time of resetting, the prescaler and the counter are all initialized to logical "1". Then the timer interruption request bit is cleared and the timer interruption request mask bit is set.

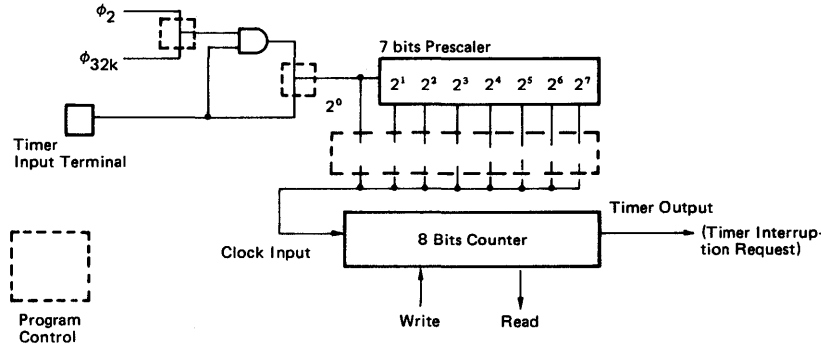


Figure 5 Timer Block Diagram

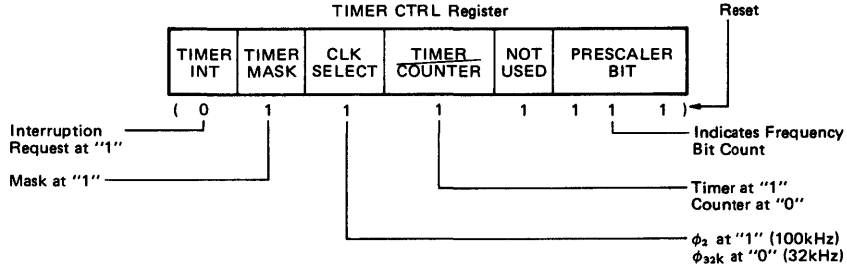


Figure 6 Timer Control Register Configuration

■ RESET

The MCU can be reset either by an external reset input ($\overline{RES}$) or by applying power. In the latter case, the reset input must be "Low" for a sufficient length of time to have the internal oscillator stabilized. A sufficient time of delay is generated by connecting a capacitor to the $\overline{RES}$ input as shown.

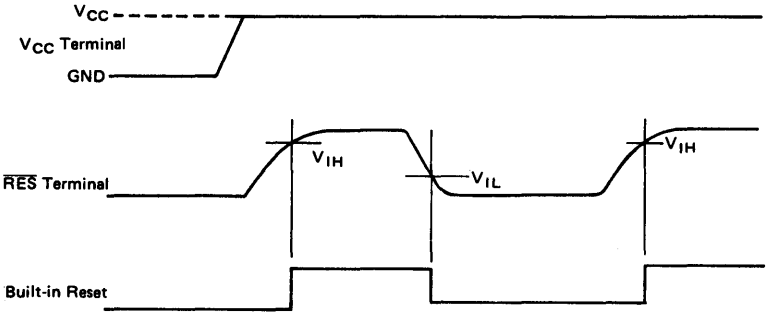


Figure 7 Application of Power and Reset Timing

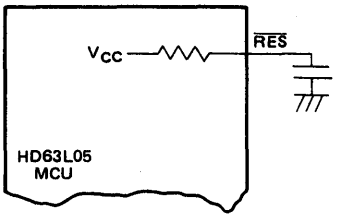


Figure 8 Input Reset Delay Circuit

■ INTERNAL OSCILLATOR OPTION

The MCU incorporates two oscillators: oscillator 1 for system clock supply and oscillator 2 for time base interruption, LCD driving and clock supply.

• Oscillator 1 (XTAL, EXTAL)

The internal oscillator circuit can be driven by an external crystal or resistor depending on the stability. Which to select, crystals or resistors, is determined by the mask option at the time of LSI production. The oscillator 1 can stop when power is applied in either Halt or Standby status. Figure 9 shows the connection.

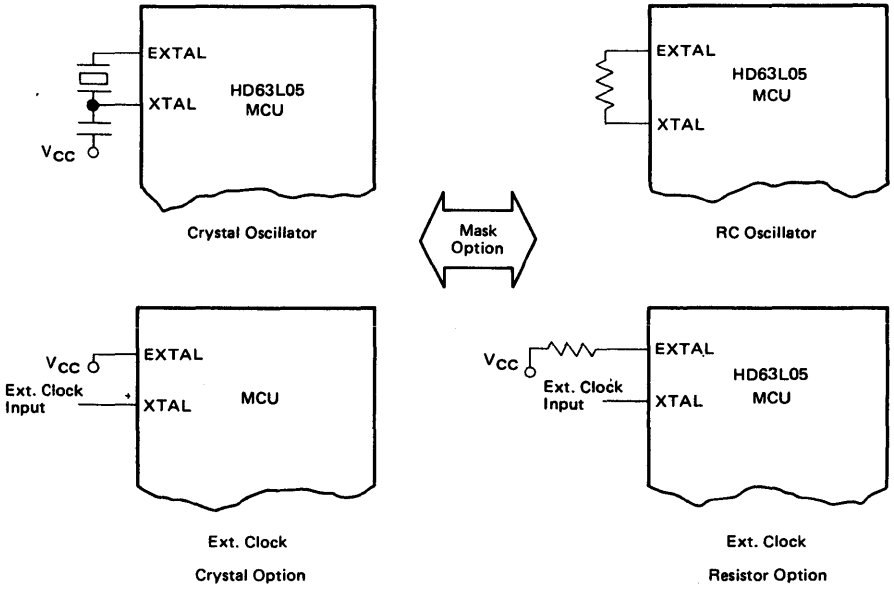


Figure 9 Mask Option for Oscillator 1

- **Oscillator 2 (XIN, XOUT)**

Clocks for time base interruption and LCD driving can be supplied by connecting a 32.768kHz crystal. In Halt status, oscillator 2 operates and this permits low power dissipation, as well as steady LCD driving and clock operation. In Standby status, this oscillator stops when power is applied. Figure 10 shows the connection; the relation between oscillators 1 and 2 is shown in Figures 11 and 12.

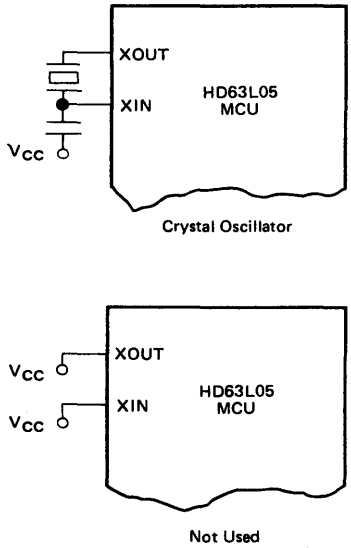


Figure 10 Connection of Oscillator 2

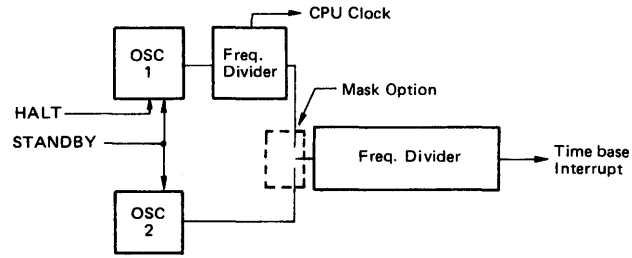


Figure 11 Relation between Oscillators 1 and 2

Mask Option	When OSC1 is X-TAL						When OSC1 is RC					
	OSC2 Not Available			OSC2 Available			OSC2 Not Available			OSC2 Available		
System / State	OSC1	CPU	Peripheral	OSC1	CPU	Peripheral	OSC1	CPU	Peripheral	OSC1	CPU	Peripheral
During System Operation	O	O	O	O	O	O	O	O	O	O	O	O
At Halt	O	X	O	O	X	O	O	X	O	X	X	O
At Standby	X	X	X	X	X	X	X	X	X	X	X	X

(NOTE) O . . . . . run X . . . . . stop

Figure 12 Oscillator 2 Mask Option and System Operation

NOTE IN OSCILLATOR SELECTION

When OSC2 is not available, the clocks for the A/D converter and LCD drivers are provided by the OSC1 through the frequency divider. When OSC1 is crystal, OSC1 is not allowend to stop at HALT. Because the response of the oscillator is not so fast. The accuracy of the time base is kept when OSC2 is 32.768 kHz crystal oscillator.

■ INTERRUPTION

There are six different interruptions to the MCU: external interruption ($\overline{INT}$), interruption envoked via an input terminal, internal timer interruption, interruption by termination of A/D conversion, time base interruption (2 types), and interruption by an instruction (SWI).

When an interruption is envoked, the job in progress is suspended and the state of the MCU is saved into the stack. Also, the interruption mask bit (I) of the condition code register is set and the start address of the interruption routine is obtained from the specified vector address. Then, the routine is executed. The RTI instruction is used where control is returned to the program to which the interruption was envoked, after the interruption service routine has been completed.

Table 1 shows the relation between interruptions, priority and vector addresses. Figure 13 shows the system operation flow, in which the portion surrounded with dot-dash lines contains the interruption execution sequence.

Table 1 Interruption Priority

Interruption	Priority	Vector Address
$\overline{RES}$	1	$FFFE, $FFFF
SWI	2	$FFFC, $FFFD
$\overline{INT}$	3	$FFFA, $FFFB
TIMER	4	$FFF8, $FFF9
A/D	5	$FFF6, $FFF7
TIME BASE	6	$FFF4, $FFF5

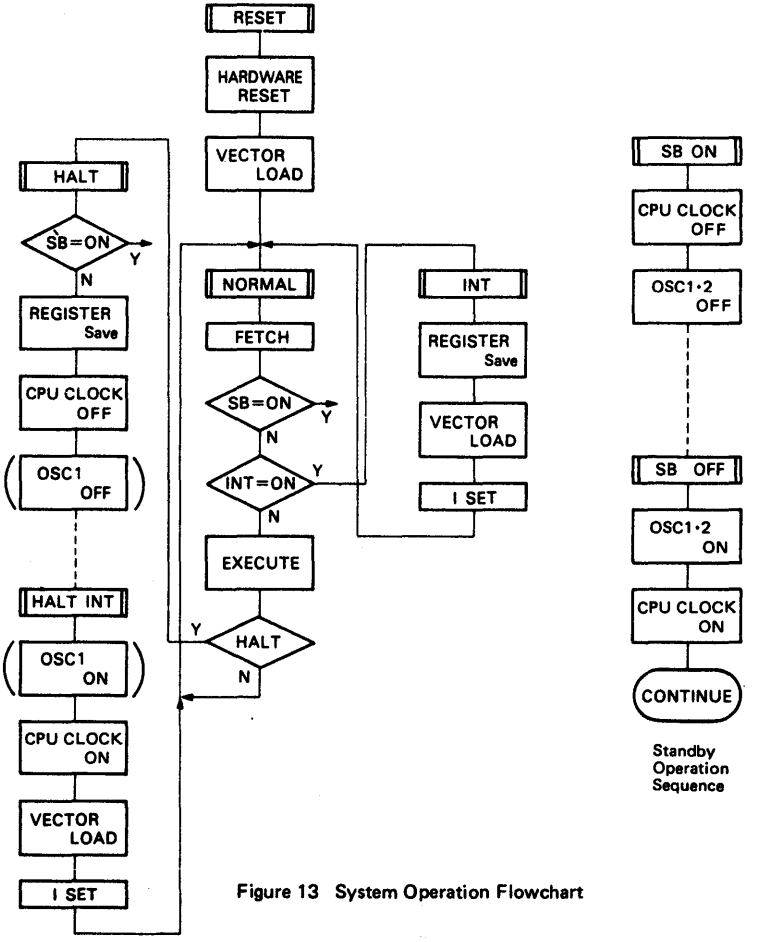


Figure 13 System Operation Flowchart

- **Acknowledging an INT in HALT Status**

In HALT status, the CPU is not operating but the peripherals are operating. When an interruption is acknowledged, the CPU is activated and executes interruption service matching the interruption condition by means of vectoring.

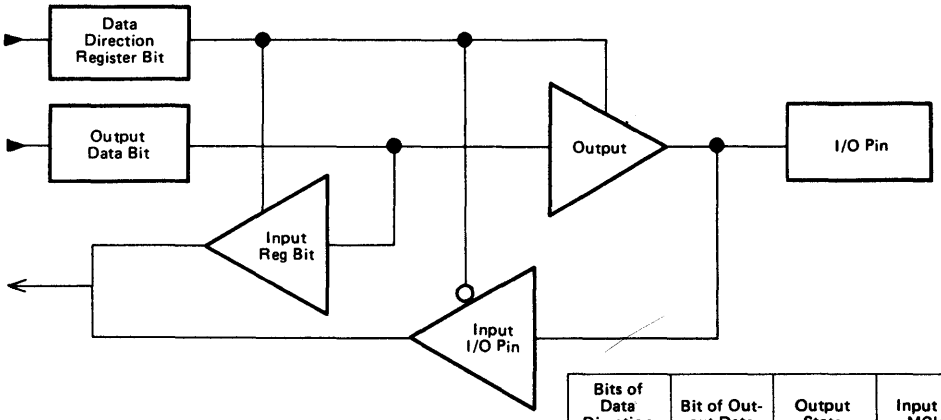
- **Acknowledging an INT in Standby Status**

In Standby status, the system is not operating with power supplied to it, therefore, any interruption request (including $\overline{RES}$) is not acknowledged.

■ INPUT/OUTPUT

There are 20 input/output terminals, which are program controlled by data direction registers for use as either input or output. If an I/O port has been programmed as an output and is read, then the latched logical level data is read even though the output level changes due to the output load.

If a port is to be used as an input terminal, the user must specify whether or not it will be equipped with a pull-up PMOS. Figure 14 shows the port I/O circuit.



Bits of Data Direction Register	Bit of Output Data	Output State	Input to MCU
1	0	0	0
1	1	1	1
0	x	3-State	Pin

Figure 14 Port I/O Circuit

• Configuration of Port

Figure 15 shows the configuration of I/O ports. As the output is on/off controlled by a data direction register, an I/O port may directly be applied as an input terminal. No problem is involved with the input if both "High" and "Low" levels are applied. For only one level, the user must specify the use of a pull-up PMOS for "Open/Low" input application.

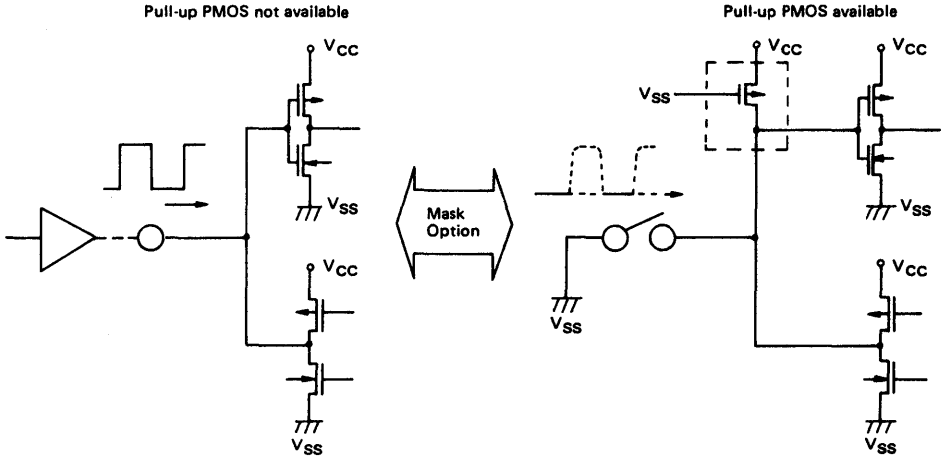
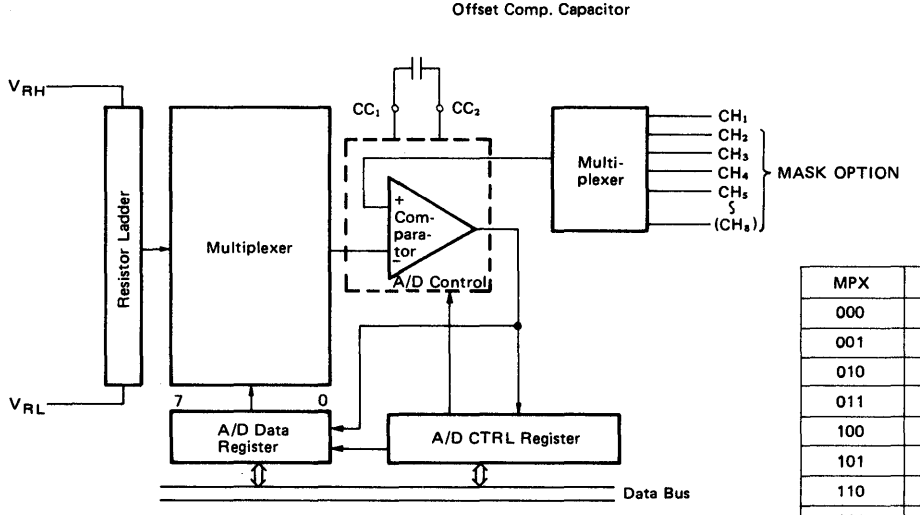


Figure 15 Selection of Input Configuration for I/O Port

■ A/D CONVERTER

The MCU incorporates an 8 bits A/D converter based on the resistor ladder system. Figure 16 shows its block diagram.



MPX	Channel
000	CH_1
001	CH_2
010	CH_3
011	CH_4
100	CH_5
101	CH_6
110	(CH_7)
111	(CH_8)

Figure 16 8 Bits A/D Converter Block Diagram

The "High" signal of reference voltage is applied V_{RH}, while the "Low" signal is applied to V_{RL}. The reference voltage is divided by resistors into voltages matching each bit, which then is compared with analog input voltage for A/D conversion.

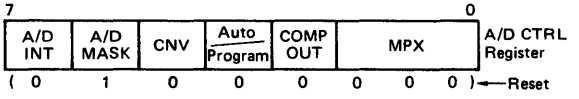


Figure 17 A/D Control Register Configuration

This voltage comparison system achieves high input impedance. Offset are compensated for by external capacitors. Figure 17 shows the configuration of the A/D control register.

• A/D INT

Used to request an interruption after completion of A/D conversion (Request at "1").

• A/D MASK

Used to mask interruptions after completion of A/D conversion (Masking at "1").

• CNV

To start A/D conversion, set this bit to "1". During conversion, "1" is held. The bit is automatically reset to "0" when the A/D conversion ends.

In A/D conversion, supply voltage is applied to the comparator only when CNV = "1". The digital data obtained by the A/D conversion is held in the A/D data register. This data is reset when the CNV is set to "1" again.

• Auto/Program

Used to select either auto-run 8 bits A/D conversion or 8 bits programmed comparator operation (Auto 8 bits A/D conversion at "0").

• COMP OUT

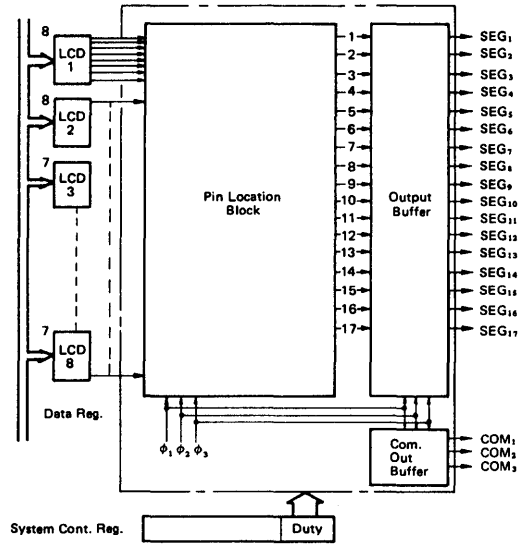
The result of comparator operation under program control can be read from this bit (At "1", input > reference voltage).

• MPX

Used to select 8-channel analog inputs. The multiplexer is an analog switch based on CMOS.

■ LCD CIRCUIT

The system configuration of the LCD circuits is shown in Figure 18. Segment data for display are stored in data registers LCD1 to LCD8. Since the circuits are connected to the output terminals via a pin location block, the user may specify a combination of data to be multiplexed to the segment output terminals.



System Cont. Duty	Contents of $SEG_1 \sim SEG_{17}$
00	OUTPUT PORT
01	———
10	———
11	1/3 Bias 1/3 Duty LCD

Figure 18 LCD Circuit System Configuration

■ LIQUID CRYSTAL DRIVER WAVEFORMS

The LCD circuit is based on 1/3 bias – 1/3 duty driving. Figure 19 shows the common electrode output signal waveforms (COM1, COM2, COM3), segment signal waveforms (SEG_1 through SEG_{17}), and LCD bias waveforms (COM-SEGMENT).

The segment output terminal may be used as an output-only terminal if the duty of the system control register is so specified. Assignment of segment terminals to the bits of the LCD data register, including the case where they are used as output-only terminals, is to be specified by the user when he orders masks.

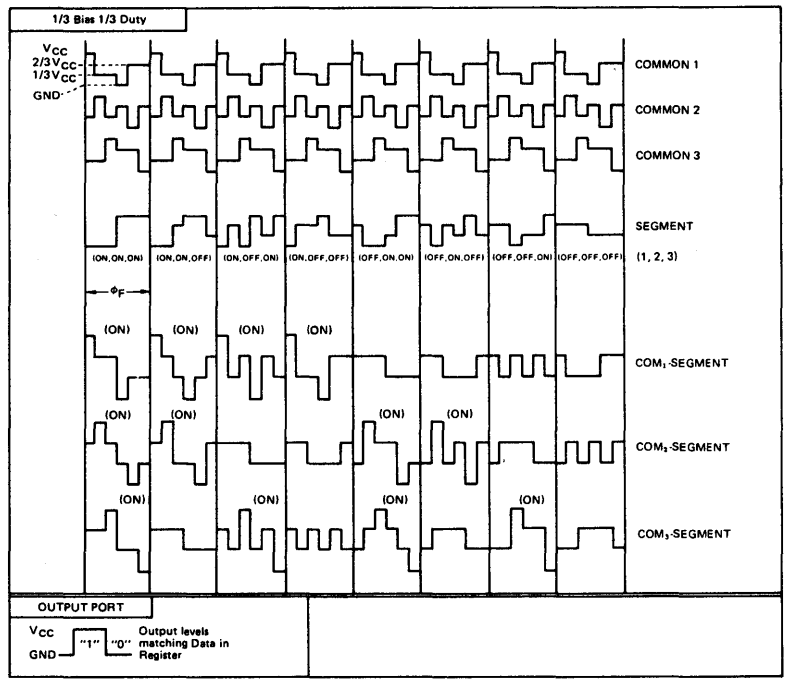


Figure 19 LCD Waveforms

■ BIT PROCESSING

This MCU can use one instruction (BSET, BCLR) to set or clear one bit of the RAM or I/O port (except for the data direction register). All bits of I/O or memory on page 0 are tested by the BRSET and BRCLR instructions. Depending on the result of the test, the program can be branched. Since the bits within the RAM, ROM or I/O can be processed by the MCU, the user can easily use a bit in the RAM as a flag or utilize a single I/O bit as an independent control terminal.

■ ADDRESSING MODE

There are 10 addressing modes available to the MCU for programming. Familiarize yourself with these modes by reading the information and referring to the diagrams that follow.

• Immediate

See Figure 20. In immediate addressing mode, constants that will not change during execution of a program are accessed. The instruction used for that purpose has a length of 2 bytes. The effective address (EA) is PC. The operand is fetched from the byte that follows the OP code.

• Direct

See Figure 21. In direct addressing mode, the address of the operand is contained in the second byte of the instruction. The user can gain direct access to the LSB 256 of memory. All RAM bytes, I/O registers, and 128 bytes of ROM are located on page 0 in order to utilize this useful addressing mode.

• Extended

See Figure 22. The extended addressing mode is used for referencing to all addresses of memory. The EA consists of the contents of the two bytes that follow the OP code. The instruction used for extended addressing has a length of 3 bytes.

• Relative

See Figure 23. Only Branch instructions are used in relative addressing mode. When a branching takes place, the contents of the byte next to the OP code are added to the program counter. EA = (PC) + 2 + Rel., where Rel. indicates signed 8 bits data at the address following the OP code. When no branching takes place, Rel. = 0. When a branching occurs, the program jumps to any byte of +129 to -127 of the current instruction. The length of the Branch instruction is 2 bytes.

• Indexed (without Offset)

See Figure 24. In this addressing mode, the lower 256 bytes of memory are accessed. The length of the instruction used for this mode is one byte. The EA consists of the contents of the index register.

• Indexed (8 Bits Offset)

See Figure 25. The EA consists of the contents of the byte following the OP code, and the contents of the index register. In this mode, the lower addresses of memory up to 511 can be accessed. Two bytes are required for the instruction.

• Indexed (16 Bits Offset)

See Figure 26. The EA consists of the contents of the two bytes following the OP code, and the contents of the index register. In this mode, the whole of the memory can be accessed. The instruction using this addressing mode has a length of 3 bytes.

• Bit Set/Clear

See Figure 27. This addressing mode can be applied to any instruction that permits any bit on page 0 to be set or cleared. The byte following the OP code indicates an address within page 0.

• Bit Test, Branch

See Figure 28. This addressing mode can be applied to instructions that test bits at the first 256 addresses ($00 to $FF) and are branched by relative qualification. The byte to be tested is addressed by the contents of the address next to the OP code. The individual bits of the byte to be tested are designated by the lower 3 bits of the OP code. The third byte indicates a relative value that is to be added to the program counter when a branch condition is satisfied. The instruction has a length of 3 bytes. The value of the bit that has been tested is written at the carry bit of the condition code register.

• Implied

See Figure 29. There is no EA for this mode. All information needed for execution of instructions is contained in the OP code. Operations that are carried out directly on the accumulator and index register are included in the implied addressing mode. In addition, the SWI and RTI instructions are also included in the group of this operation. The instruction using this addressing has a length of one byte.

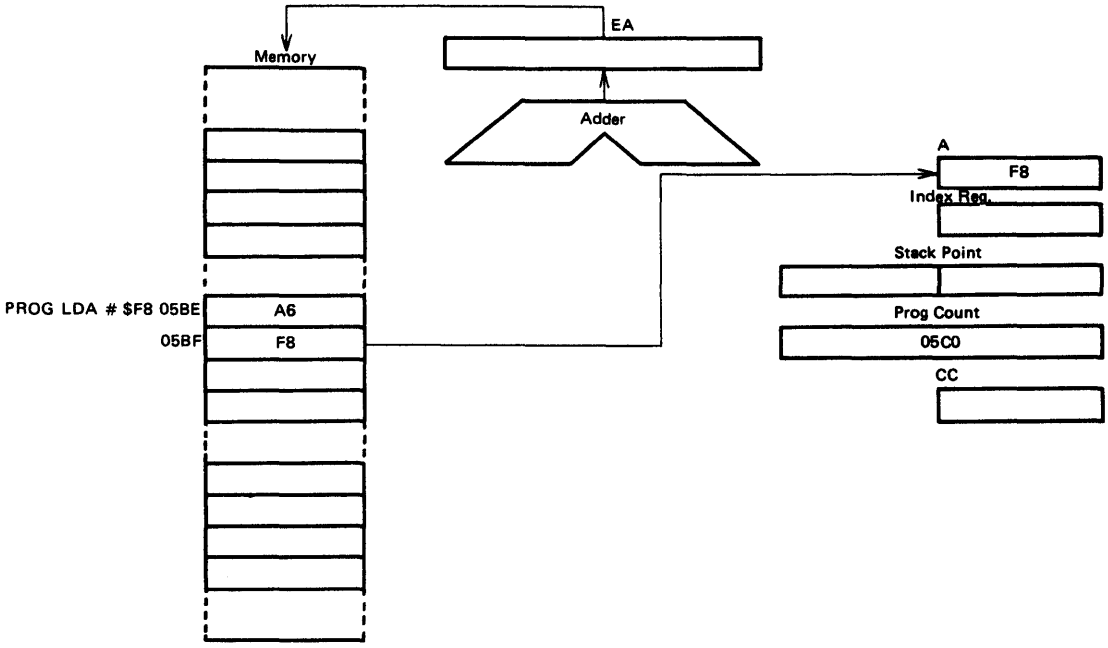


Figure 20 Example of Immediate Addressing

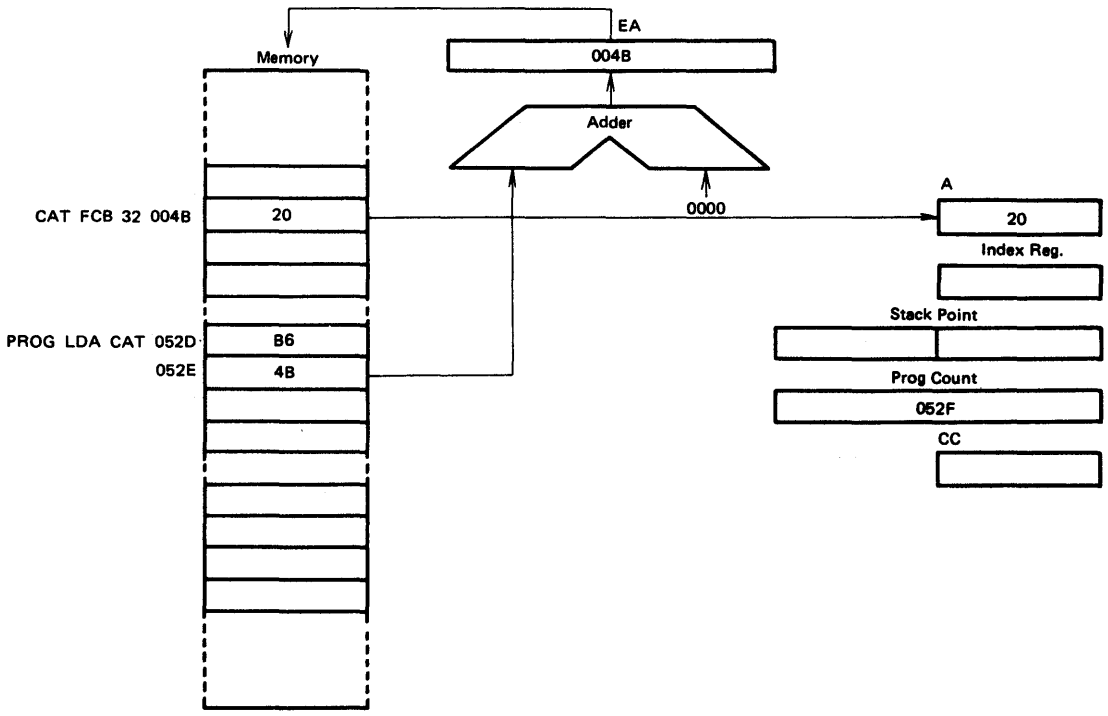


Figure 21 Example of Direct Addressing

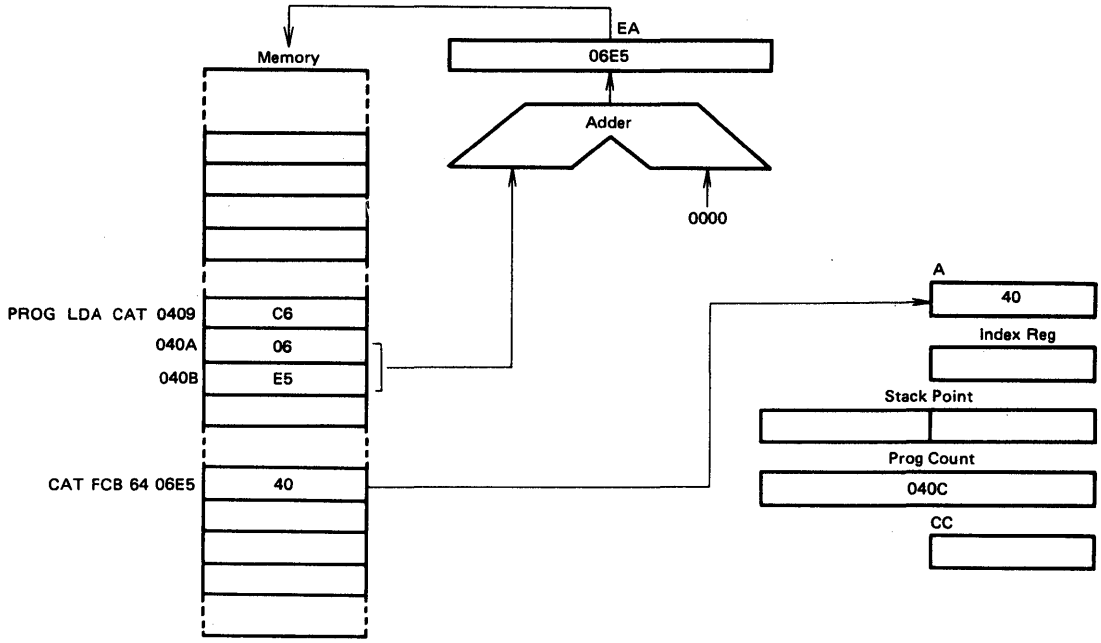


Figure 22 Example of Extended Addressing

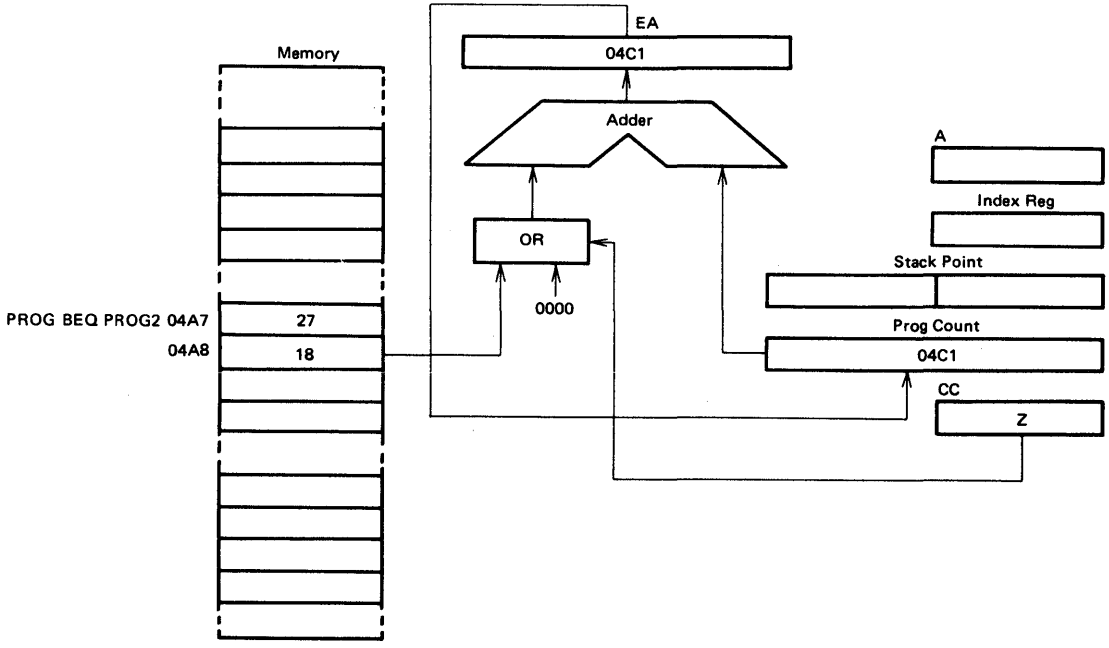


Figure 23 Example of Relative Addressing

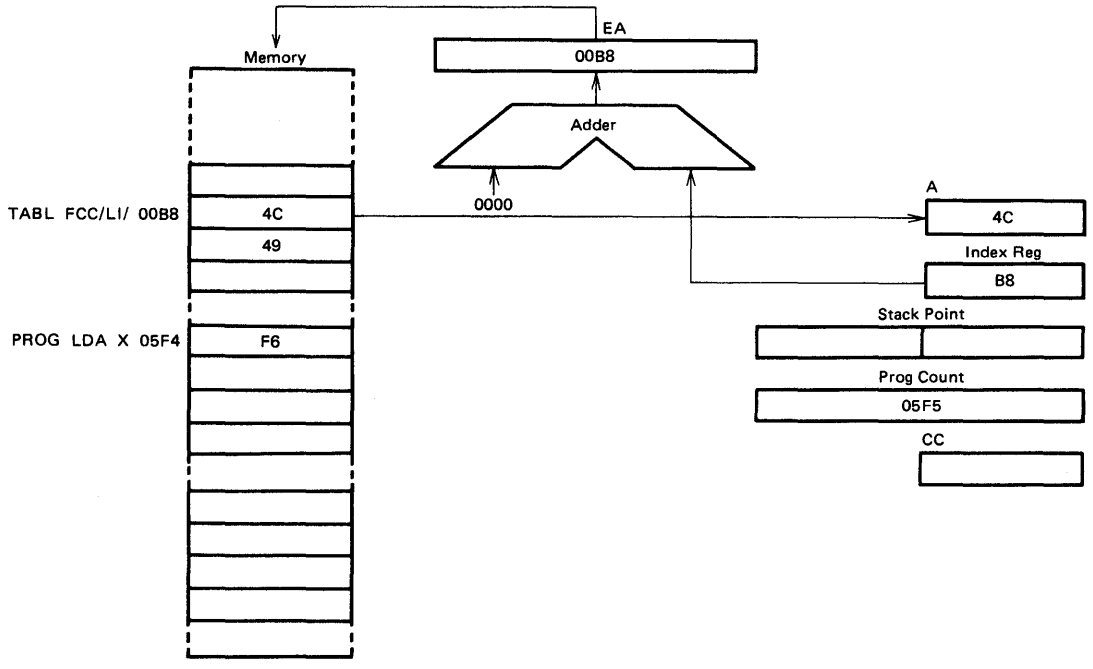


Figure 24 Example of Indexed (without Offset) Addressing

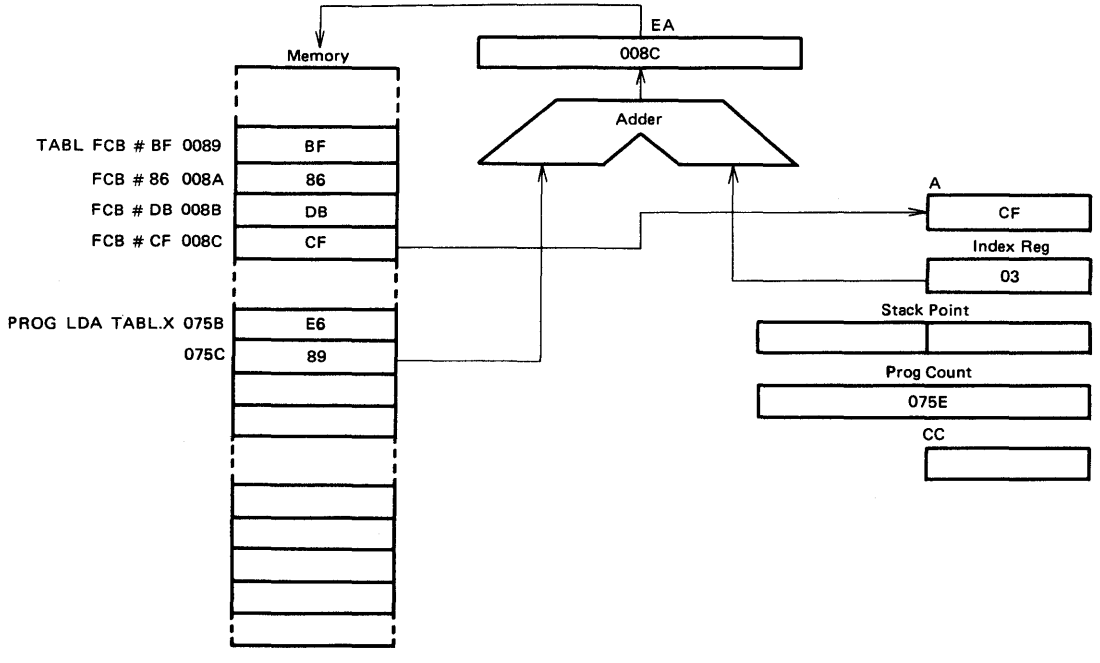


Figure 25 Example of Indexed (8 Bits Offset) Addressing

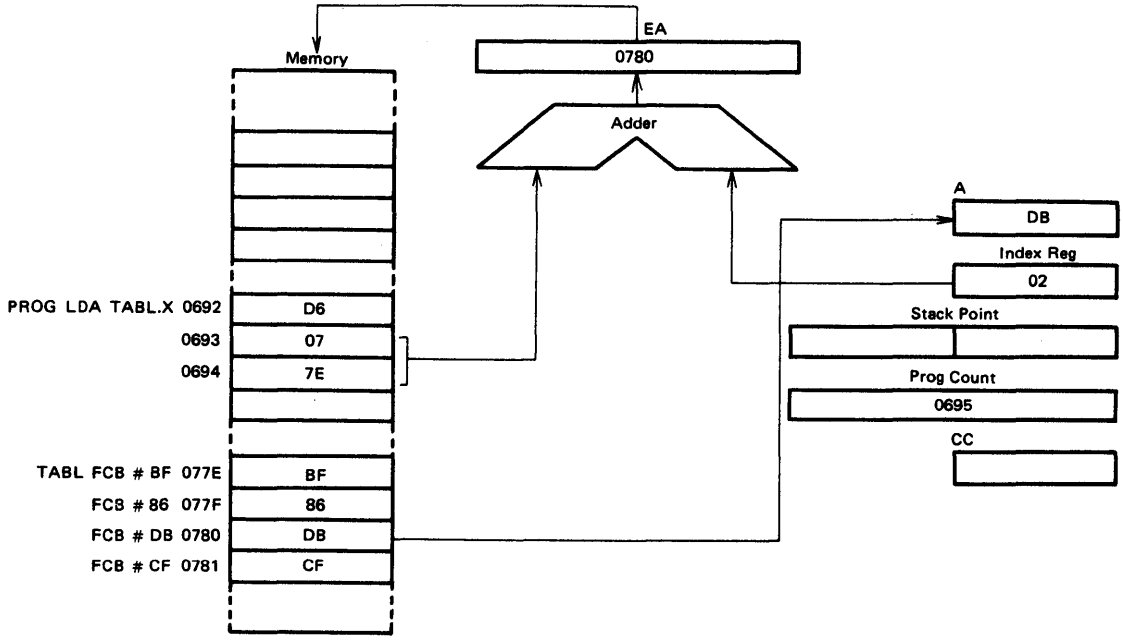


Figure 26 Example of Indexed (16 Bits Offset) Addressing

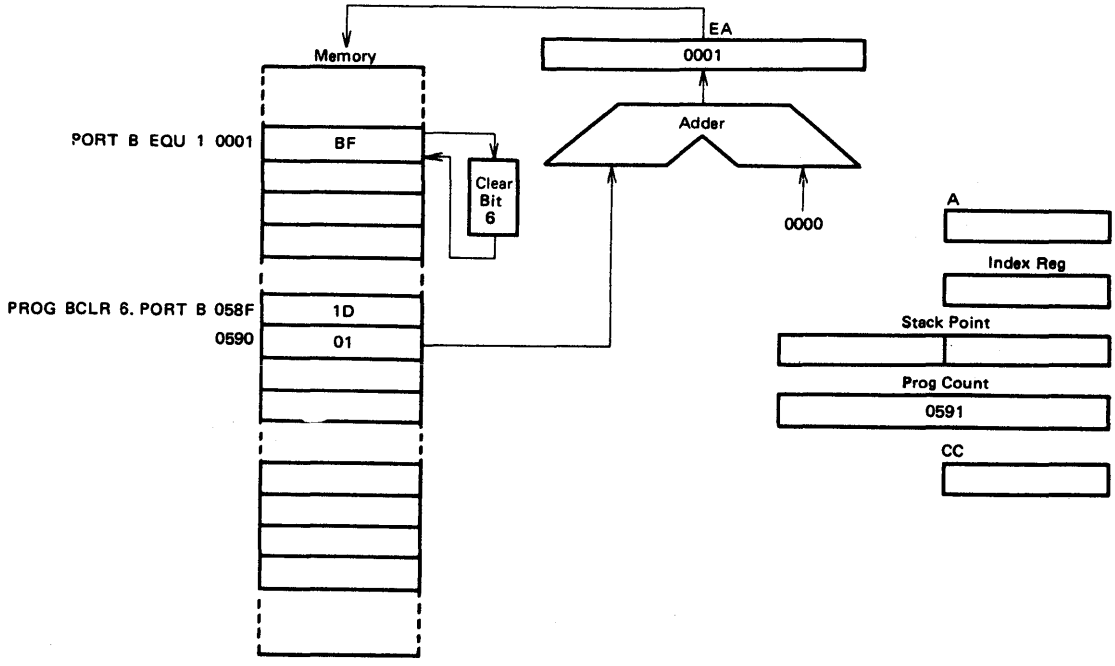


Figure 27 Example of Bit Set/Clear Addressing

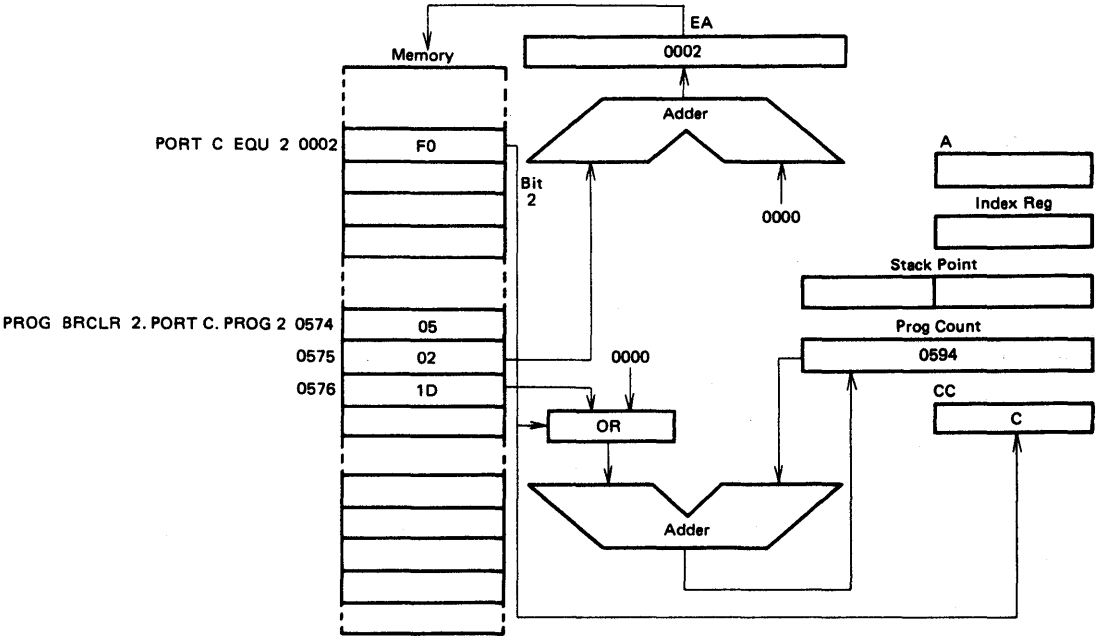


Figure 28 Example of Bit Test and Branch Addressing

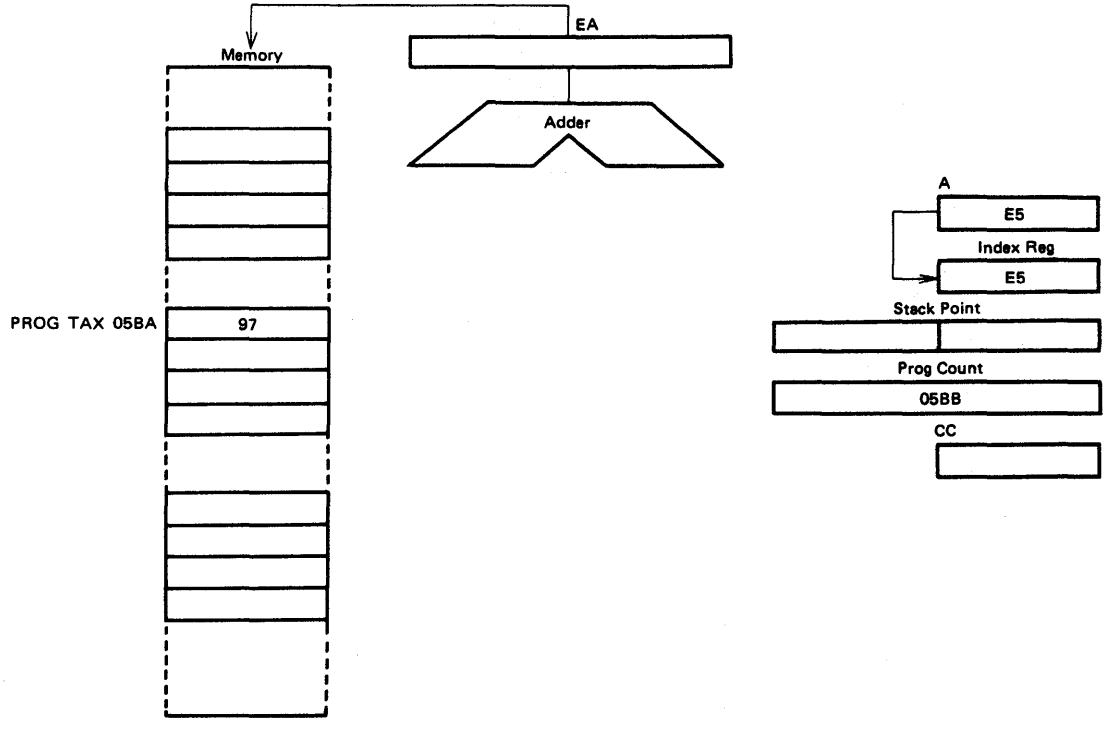


Figure 29 Example of Implied Addressing

■ INSTRUCTION SET

There are 59 instructions available to the MCU. They can be divided into five groups: Register/Memory, Read/Modify/Write, Branch, Bit Processing, and Control. All of these instructions are explained below according to the groups, and are summarized in individual tables.

• Register/Memory

Most of these instructions use two operands. One operand is either the accumulator or index register, while the other is acquired from memory using one of the addressing modes. No operand of register is available in the unconditional Jump (JMP) and Subroutine Jump (JSR) instructions. See Table 2.

• Read/Modify/Write

These instructions read a memory address or register, modify or test its contents, and writes a new value into the memory or register. Negative or Zero instructions (TST) do not provide writing, and are exceptions for the Read/Modify/Write. See Table 3.

• Branch

A Branch instruction will branch from the program sequence in progress if the specific branch condition is satisfied. See Table 4.

• Bit Processing

This instruction can be used for any bit of the first 256 bytes of memory. One group is used for setting or clearing, while the other is used for bit testing and branching. See Table 5.

• Control

The Control instruction controls the operation of the MCU for which a program is being executed. See Table 6.

• A List of Instructions Arranged in Alphabetical Order

All instructions are listed in Table 7 in the alphabetical order.

• OP Code Map

Table 8 shows an OP code map of the instructions used with the MCU.

Table 2 Register/Memory Instructions

		Addressing Mode																	
		Immediate			Direct			Extended			Indexed (No Offset)			Indexed (8-Bit Offset)			Indexed (16-Bit Offset)		
Operation	Mnemonic	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Load A from Memory	LDA	A6	2	2	B6	2	3	C6	3	4	F6	1	2	E6	2	4	D6	3	5
Load X from Memory	LDX	AE	2	2	BE	2	3	CE	3	4	FE	1	2	EE	2	4	DE	3	5
Store A in Memory	STA	–	–	–	B7	2	4	C7	3	5	F7	1	3	E7	2	5	D7	3	6
Store X in Memory	STX	–	–	–	BF	2	4	CF	3	5	FF	1	3	EF	2	5	DF	3	6
Add Memory to A	ADD	AB	2	2	BB	2	3	CB	3	4	FB	1	2	EB	2	4	DB	3	5
Add Memory and Carry to A	ADC	A9	2	2	B9	2	3	C9	3	4	F9	1	2	E9	2	4	D9	3	5
Subtract Memory	SUB	A0	2	2	B0	2	3	C0	3	4	F0	1	2	E0	2	4	D0	3	5
Subtract Memory from A with Borrow	SBC	A2	2	2	B2	2	3	C2	3	4	F2	1	2	E2	2	4	D2	3	5
AND Memory to A	AND	A4	2	2	B4	2	3	C4	3	4	F4	1	2	E4	2	4	D4	3	5
OR Memory with A	ORA	AA	2	2	BA	2	3	CA	3	4	FA	1	2	EA	2	4	DA	3	5
Exclusive OR Memory with A	EOR	A8	2	2	B8	2	3	C8	3	4	F8	1	2	E8	2	4	D8	3	5
Arithmetic Compare A with Memory	CMP	A1	2	2	B1	2	3	C1	3	4	F1	1	2	E1	2	4	D1	3	5
Arithmetic Compare X with Memory	CPX	A3	2	2	B3	2	3	C3	3	4	F3	1	2	E3	2	4	D3	3	5
Bit Test Memory with A (Logical Compare)	BIT	A5	2	2	B5	2	3	C5	3	4	F5	1	2	E5	2	4	D5	3	5
Jump Unconditional	JMP	–	–	–	BC	2	2	CC	3	3	FC	1	2	EC	2	3	DC	3	4
Jump to Subroutine	JSR	–	–	–	BD	2	4	CD	3	5	FD	1	3	ED	2	4	DD	3	5

Symbols: Op = Operation # = Instruction

Table 3 Read/Modify/Write Instructions

		Addressing Mode														
		Implied (A)			Implied (X)			Direct			Indexed (No Offset)			Indexed (8-Bit Offset)		
Operation	Mnemonic	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Increment	INC	4C	1	1	5C	1	1	3C	2	4	7C	1	3	6C	2	5
Decrement	DEC	4A	1	1	5A	1	1	3A	2	4	7A	1	3	6A	2	5
Clear	CLR	4F	1	1	5F	1	1	3F	2	4	7F	1	3	6F	2	5
Complement	COM	43	1	1	53	1	1	33	2	4	73	1	3	63	2	5
Negate (2's Complement)	NEG	40	1	1	50	1	1	30	2	4	70	1	3	60	2	5
Rotate Left Thru Carry	ROL	49	1	1	59	1	1	39	2	4	79	1	3	69	2	5
Rotate Right Thru Carry	ROR	46	1	1	56	1	1	36	2	4	76	1	3	66	2	5
Logical Shift Left	LSL	48	1	1	58	1	1	38	2	4	78	1	3	68	2	5
Logical Shift Right	LSR	44	1	1	54	1	1	34	2	4	74	1	3	64	2	5
Arithmetic Shift Right	ASR	47	1	1	57	1	1	37	2	4	77	1	3	67	2	5
Arithmetic Shift Left	ASL	48	1	1	58	1	1	38	2	4	78	1	3	68	2	5
Test for Negative or Zero	TST	4D	1	1	5D	1	1	3D	2	4	7D	1	3	6D	2	5

Symbols: Op = Operation # = Instruction

Table 4 Branch Instructions

Operation	Mnemonic	Relative Addressing Mode		
		Op Code	# Bytes	# Cycles
Branch Always	BRA	20	2	3
Branch Never	BRN	21	2	2 or 3 *
Branch IF Higher	BHI	22	2	2 or 3 *
Branch IF Lower or Same	BLS	23	2	2 or 3 *
Branch IF Carry Clear	BCC	24	2	2 or 3 *
(Branch IF Higher or Same)	(BHS)	24	2	2 or 3 *
Branch IF Carry Set	BCS	25	2	2 or 3 *
(Branch IF Lower)	(BLO)	25	2	2 or 3 *
Branch IF Not Equal	BNE	26	2	2 or 3 *
Branch IF Equal	BEQ	27	2	2 or 3 *
Branch IF Half Carry Clear	BHCC	28	2	2 or 3 *
Branch IF Half Carry Set	BHCS	29	2	2 or 3 *
Branch IF Plus	BPL	2A	2	2 or 3 *
Branch IF Minus	BMI	2B	2	2 or 3 *
Branch IF Interrupt Mask Bit is Clear	BMC	2C	2	2 or 3 *
Branch IF Interrupt Mask Bit is Set	BMS	2D	2	2 or 3 *
Branch IF Interrupt Line is Low	BIL	2E	2	2 or 3 *
Branch IF Interrupt Line is High	BIH	2F	2	2 or 3 *
Branch to Subroutine	BSR	AD	2	4

Symbol: Op = Operation # = Instruction
* If branched, each instruction will be a 3-cycle instruction.

Table 5 Bit Processing Instructions

Operations	Mnemonic	Addressing Mode					
		Bit Set/Clear			Bit Test and Branch		
		Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Branch IF Bit n is Set	BRSET n (n = 0.....7)	–	–	–	2 • n	3	4 or 5 *
Branch IF Bit n is Clear	BRCLR n (n = 0.....7)	–	–	–	01 + 2 • n	3	4 or 5 *
Set Bit n	BSET n (n = 0.....7)	10 + 2 • n	2	4	–	–	–
Clear Bit n	BCLR n (n = 0.....7)	11 + 2 • n	2	4	–	–	–

Symbol: Op = Operation # = Instruction
* If Branched, each instruction will be a 5-cycle instruction.

Table 6 Control Instructions

Operation	Mnemonic	Implied		
		Op Code	# Bytes	# Cycles
Transfer A to X	TAX	97	1	1
Transfer X to A	TXA	9F	1	1
Set Carry Bit	SEC	99	1	1
Clear Carry Bit	CLC	98	1	1
Set Interrupt Mask Bit	SEI	9B	1	1
Clear Interrupt Mask Bit	CLI	9A	1	1
Software Interrupt	SWI	83	1	9
Return from Subroutine	RTS	81	1	4
Return from Interrupt	RTI	80	1	7
Reset Stack Pointer	RSP	9C	1	1
No-Operation	NOP	9D	1	1

Symbol: Op = Operation # = Instruction

Table 7 Instruction Set

Mnemonic	Addressing Modes										Condition Code				
	Implied	Immediate	Direct	Extended	Relative	Indexed (No Offset)	Indexed (8 Bits)	Indexed (16 Bits)	Bit Set/ Clear	Bit Test & Branch	H	I	N	Z	C
ADC		x	x	x		x	x	x			∧	•	∧	∧	∧
ADD		x	x	x		x	x	x			∧	•	∧	∧	∧
AND		x	x	x		x	x	x			•	•	∧	∧	•
ASL	x		x			x	x				•	•	∧	∧	∧
ASR	x		x			x	x				•	•	∧	∧	∧
BCC					x						•	•	•	•	•
BCLR									x		•	•	•	•	•
BCS					x						•	•	•	•	•
BEQ					x						•	•	•	•	•
BHCC					x						•	•	•	•	•
BHCS					x						•	•	•	•	•
BHI					x						•	•	•	•	•
BHS					x						•	•	•	•	•
BIH					x						•	•	•	•	•
BIL					x						•	•	•	•	•
BIT		x	x	x		x	x	x			•	•	∧	∧	•
BLO					x						•	•	•	•	•
BLS					x						•	•	•	•	•
BMC					x						•	•	•	•	•
BMI					x						•	•	•	•	•
BMS					x						•	•	•	•	•
BNE					x						•	•	•	•	•
BPL					x						•	•	•	•	•
BRA					x						•	•	•	•	•

(Continued)

Symbols for condition code:
- H Half Carry (From Bit 3)
- I Interrupt Mask
- N Negative (Sign Bit)
- Z Zero
- C Carry/Borrow
- ∧ Test and Set if True, Cleared Otherwise
- • Not Affected

Table 7 Instruction Set (Continued)

Mnemonic	Addressing Modes										Condition Code				
	Implied	Immediate	Direct	Extended	Relative	Indexed (No Offset)	Indexed (8 Bits)	Indexed (16 Bits)	Bit Set/Clear	Bit Test & Branch	H	I	N	Z	C
BRN					x						•	•	•	•	•
BRCLR										x	•	•	•	•	∧
BRSET										x	•	•	•	•	∧
BSET									x		•	•	•	•	•
BSR					x						•	•	•	•	•
CLC	x										•	•	•	•	0
CLI	x										•	0	•	•	•
CLR	x		x			x	x				•	•	0	1	•
CMP		x	x	x		x	x	x			•	•	∧	∧	∧
COM	x		x			x	x				•	•	∧	∧	1
CPX		x	x	x		x	x	x			•	•	∧	∧	∧
DEC	x		x			x	x				•	•	∧	∧	•
EOR		x	x	x		x	x	x			•	•	∧	∧	•
INC	x		x			x	x				•	•	∧	∧	•
JMP			x	x		x	x	x			•	•	•	•	•
JSR			x	x		x	x	x			•	•	•	•	•
LDA		x	x	x		x	x	x			•	•	∧	∧	•
LDX		x	x	x		x	x	x			•	•	∧	∧	•
LSL	x		x			x	x				•	•	∧	∧	∧
LSR	x		x			x	x				•	•	0	∧	∧
NEG	x		x			x	x				•	•	∧	∧	∧
NOP	x										•	•	•	•	•
ORA		x	x	x		x	x	x			•	•	∧	∧	•
ROL	x		x			x	x				•	•	∧	∧	∧
ROR	x		x			x	x				•	•	∧	∧	∧
RSP	x										•	•	•	•	•
RTI	x										?	?	?	?	?
RTS	x										•	•	•	•	•
SBC		x	x	x		x	x	x			•	•	∧	∧	∧
SEC	x										•	•	•	•	1
SEI	x										•	1	•	•	•
STA			x	x		x	x	x			•	•	∧	∧	•
STX			x	x		x	x	x			•	•	∧	∧	•
SUB		x	x	x		x	x	x			•	•	∧	∧	∧
SWI	x										•	1	•	•	•
TAX	x										•	•	•	•	•
TST	x		x			x	x				•	•	∧	∧	•
TXA	x										•	•	•	•	•

Symbols for condition code:

H Half Carry (From Bit 3)
I Interrupt Mask
N Negative (Sign Bit)
Z Zero
C Carry/Borrow
∧ Test and Set if True, Cleared Otherwise
• Not Affected
? Load CC Register From Stack

Table 8 OP Code Map

	Bit Manipulation		Branch	Read/Modify/Write					Control		Register/Memory						
	Test & Branch	Set/ Clear	Rel	DIR	A	X	,X1	,X0	IMP	IMP	IMM	DIR	EXT	,X2	,X1	,X0	
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	←HIGH
0	BRSET0	BSET0	BRA	NEG					RTI*	–	SUB						0
1	BRCLR0	BCLR0	BRN	–					RTS*	–	CMP						1
2	BRSET1	BSET1	BHI	–					–	–	SBC						2
3	BRCLR1	BCLR1	BLS	COM					SWI*	–	CPX						3 L
4	BRSET2	BSET2	BCC	LSR					–	–	AND						4 O
5	BRCLR2	BCLR2	BCS	–					–	–	BIT						5 W
6	BRSET3	BSET3	BNE	ROR					–	–	LDA						6
7	BRCLR3	BCLR3	BEQ	ASR					–	TAX	–	STA (+1)					7
8	BRSET4	BSET4	BHCC	LSL/ASL					–	CLC	EOR						8
9	BRCLR4	BCLR4	BHCS	ROL					–	SEC	ADC						9
A	BRSET5	BSET5	BPL	DEC					–	CLI	ORA						A
B	BRCLR5	BCLR5	BMI	–					–	SEI	ADD						B
C	BRSET6	BSET6	BMC	INC					–	RSP	–	JMP(–1)					C
D	BRCLR6	BCLR6	BMS	TST					–	NOP	BSR*	JSR(+1)		JSR		JSR(+1)	D
E	BRSET7	BSET7	BIL	–					–	–	LDX						E
F	BRCLR7	BCLR7	BIH	CLR					–	TXA	–	STX(+1)					F
	3/4 or 5	2/4	2/2 or 3	2/4	1/1	1/1	2/5	1/3	1/*	1/1	2/2	2/3	3/4	3/5	2/4	1/2	

(NOTES) 1. "–" is an undefined operation code.
2. The figure in the lowest row of each column gives the number of bytes and the cycles needed for the instruction.
The number of cycles for the asterisked (*) mnemonics is a follows:

RTI	7
RTS	4
SWI	9
BSR	4

3. The parenthesized figure must be added to the cycle count of the associated instruction.
4. If the instruction is branched, the cycle count is the larger figure.

■ HD63L05 LCD PIN LOCATION COMPOSITION TABLE

Register \ Bit		Multiplexed Timing			Segment Output Terminal																
		COM1	COM2	COM3	SEG1	SEG2	SEG3	SEG4	SEG5	SEG6	SEG7	SEG8	SEG9	SEG10	SEG11	SEG12	SEG13	SEG14	SEG15	SEG16	SEG17
LCD1	0																				
	1																				
	2																				
	3																				
	4																				
	5																				
	6																				
	7																				
LCD2	0																				
	1																				
	2																				
	3																				
	4																				
	5																				
	6																				
	7																				
LCD3	0																				
	1																				
	2																				
	3																				
	4																				
	5																				
	6																				
LCD4	0																				
	1																				
	2																				
	3																				
	4																				
	5																				
	6																				
LCD5	0																				
	1																				
	2																				
	3																				
	4																				
	5																				
	6																				
LCD6	0																				
	1																				
	2																				
	3																				
	4																				
	5																				
	6																				
LCD7	0																				
	1																				
	2																				
	3																				
	4																				
	5																				
	6																				
LCD8	0																				
	1																				
	2																				
	3																				
	4																				
	5																				
	6																				
ϕWRITE		○																			

(NOTE) Mark a selected Multiplexed Timing and Segment Output Terminal with a circle (0).
In the case of Output or Static LCD driver, Multiplexed Timing is fixed at COM_1.
ϕ_{WRITE} is a write clock for the external option (EXT = "1").
It is generated when LCD1 is rewritten by the CPU.

■ **HD63L05 I/O COMPOSITION TABLE**

Pin Name	I/O	I/O Option A	I/O Option B	I/O Option C	I/O Option D	Remarks
A_0	I/O					
A_1	I/O					
A_2	I/O					
A_3	I/O					
A_4	I/O					
A_5	I/O					
A_6	I/O					
A_7	I/O					
B_0	I/O					
B_1	I/O					
B_2	I/O					
B_3	I/O					
B_4	I/O					
B_5	I/O					
B_6	I/O					
B_7	I/O					
C_0	I/O					
C_1	I/O					
C_2	I/O					
C_3	I/O					
$\overline{INT}$	I					

Pin Name	I/O	I/O Option E	I/O Option F	I/O Option G	I/O Option H	Remarks
SEG_1	O					
SEG_2	O					
SEG_3	O					
SEG_4	O					
SEG_5	O					
SEG_6	O					
SEG_7	O					
SEG_8	O					
SEG_9	O					
SEG_{10}	O					
SEG_{11}	O					
SEG_{12}	O					
SEG_{13}/CH_6	O,I					
SEG_{14}/CH_5	O,I					
SEG_{15}/CH_4	O,I					
SEG_{16}/CH_3	O,I					
SEG_{17}/CH_2	O,I					
V_1/CH_7	O,I					
V_2/CH_8	O,I					

(NOTE) Mark a selected composition with a circle (0).
A. No pull up MOS
B. With pull up MOS
C. CMOS Output
D. Open Drain Output
E. A/D Input
F. Segment Output
G. Output Port
H. LCD Power Supply

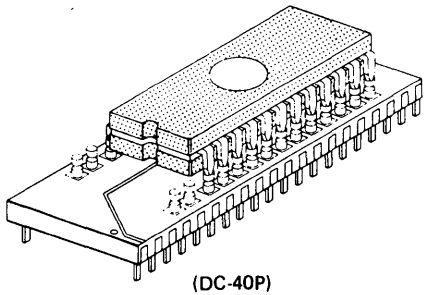
HD68P01S0, HD68P01V05, HD68P01V07

MCU (Microcomputer Unit)

—PRELIMINARY—

The HD68P01 is an 8-bit single chip microcomputer unit (MCU) which significantly enhances the capabilities of the HMCS6800 family of parts. It can be used in production systems to allow for easy firmware changes with minimum delay or it can be used to emulate the HD6801 for software development. It includes 128 bytes of RAM, Serial Communications Interface (SCI), parallel I/O and a three function Programmable Timer on chip, and 2048 bytes, 4096 bytes or 8192 bytes of EPROM on package. It includes an upgrade HD6800 microprocessing unit (MPU) while retaining upward source and object code compatibility. Execution times of key instructions have been improved and several new instructions have been added including an unsigned 8 by 8 multiply with 16-bit result. The HD68P01 can function as a monolithic microcomputer or can be expanded to a 65k byte address space. It is TTL compatible and requires one +5 volt power supply. A summary of HD68P01 features includes:

HD68P01S0, HD68P01V05, HD68P01V07



(DC-40P)

■ FEATURES

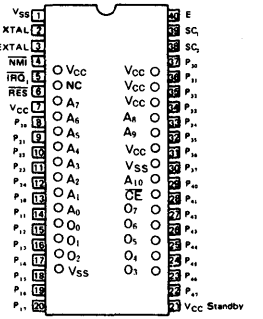
- Expanded HMCS6800 Instruction Set
- 8 x 8 Multiply Instruction
- Serial Communications Interface (SCI)
- Upward Source and Object Code Compatible with HD6800
- 16-bit Three-function Programmable Timer
- Applicable to All Type of EPROM
  - 2048 bytes; HN462716
  - 4096 bytes; HN462732 or HN462532
  - 8192 bytes; HN482764
- 128 Bytes of RAM (64 bytes Retainable on Powerdown)
- 29 Parallel I/O and Two Handshake Control Line
- Internal Clock Generator with Divide-by-Four Output
- Full TTL Compatibility
- Full Interrupt Capability
- Single-Chip or Expandable to 65k Bytes Address Space
- Bus compatible with HMCS6800 Family

■ TYPE OF PRODUCTS

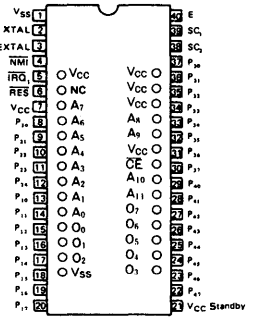
Type No.	Bus Timing	EPROM Type No.
HD68P01S0	1 MHz	HN462716
HD68P01V05	1 MHz	HN462532
HD68P01V07	1 MHz	HN462732

■ PIN ARRANGEMENT (Top View)

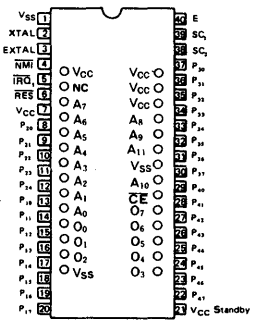
HD68P01S0



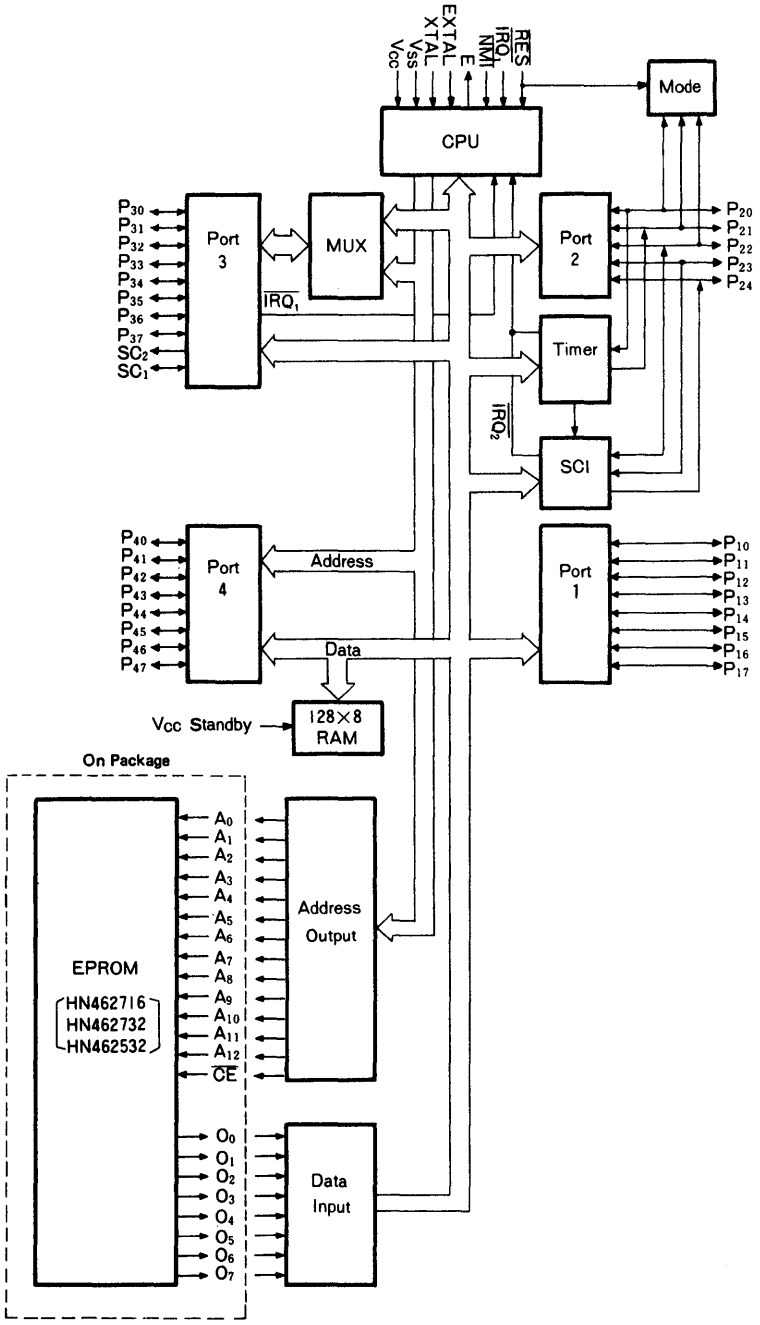
HD68P01V05



HD68P01V07



■ BLOCK DIAGRAM



■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC} *	-0.3 ~ +7.0	V
Input Voltage	V_{in} *	-0.3 ~ +7.0	V
Operating Temperature	T_{opr}	0 ~ +70	°C
Storage Temperature	T_{stg}	-55 ~ +150	°C

* With respect to V_{SS} (SYSTEM GND)

[NOTE] Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ ELECTRICAL CHARACTERISTICS

• DC CHARACTERISTICS (V_{CC} =5.0V±5%, V_{SS} = 0V, Ta = 0 ~ +70°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Input "High" Voltage	$\overline{RES}$	V_{IH}		4.0	–	V_{CC}	V
	Other Inputs*			2.0	–	V_{CC}	
Input "Low" Voltage	All Inputs*	V_{IL}		-0.3	–	0.8	V
Input Load Current	P_{40} ~ P_{47}	$\|I_{in}\|$	V_{in} = 0 ~ 2.4V	–	–	0.5	mA
	SC_1			–	–	0.8	
	EXTAL		V_{in} = 0 ~ V_{CC}	–	–	0.8	
Input Leakage Current	$\overline{NMI}$, $\overline{IRQ_1}$, $\overline{RES}$	$\|I_{in}\|$	V_{in} = 0 ~ 5.25V	–	–	2.5	μA
Three State (Offset) Leakage Current	P_{10} ~ P_{17}, P_{30} ~ P_{37}	$\|I_{TSI}\|$	V_{in} = 0.5 ~ 2.4V	–	–	10	μA
	P_{20} ~ P_{24}			–	–	100	
Output "High" Voltage	P_{30} ~ P_{37}	V_{OH}	I_{LOAD} = -205 μA	2.4	–	–	V
	P_{40} ~ P_{47}, E, SC_1, SC_2		I_{LOAD} = -145 μA	2.4	–	–	
	Other Outputs		I_{LOAD} = -100 μA	2.4	–	–	
Output "Low" Voltage	All Outputs	V_{OL}	I_{LOAD} = 1.6 mA	–	–	0.5	V
Darlington Drive Current	P_{10} ~ P_{17}	$-I_{OH}$	V_{out} = 1.5V	1.0	–	10.0	mA
Power Dissipation		P_D		–	–	1200	mW
Input Capacitance	P_{30} ~ P_{37}, P_{40} ~ P_{47}, SC_1	C_{in}	V_{in} = 0V, Ta = 25°C, f = 1.0 MHz	–	–	12.5	pF
	Other Inputs			–	–	10.0	
V_{CC} Standby	Powerdown	V_{SBB}		4.0	–	5.25	V
	Operating	V_{SB}		4.75	–	5.25	
Standby Current	Powerdown	I_{SBB}	V_{SBB} = 4.0V	–	–	8.0	mA
Frequency of Operation	Extarnal Clock	$4f_O$		2.0	–	4.0	MHz
	Crystal	f_{XTAL}		3.579	–	4.0	

*Except Mode Programming Levels: See Figure 8.

• AC CHARACTERISTICS

BUS TIMING (V_{CC} = 5.0V±5%, V_{SS} = 0V, Ta = 0 ~ +70°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Cycle Time		t_{cyc}		1	–	10	μs
Address Strobe Pulse Width "High"		PW_{ASH}		200	–	–	ns
Address Strobe Rise Time		t_{ASr}		5	–	50	ns
Address Strobe Fall Time		t_{ASf}		5	–	50	ns
Address Strobe Delay Time		t_{ASD}		60	–	–	ns
Enable Rise Time		t_{Er}		5	–	50	ns
Enable Fall Time		t_{Ef}		5	–	50	ns
Enable Pulse Width "High" Time		PW_{EH}		450	–	–	ns
Enable Pulse Width "Low" Time		PW_{EL}		450	–	–	ns
Address Strobe to Enable Delay Time		t_{ASED}		60	–	–	ns
Address Delay Time		t_{AD}	Fig. 1	–	–	260	ns
Address Delay Time for Latch (f=1.0MHz)		t_{ADL}	Fig. 2	–	–	270	ns
Data Set-up Write Time		t_{DSW}		225	–	–	ns
Data Set-up Read Time		t_{DSR}		80	–	–	ns
Data Hold Time	Read	t_{HR}		10	–	–	ns
	Write	t_{HW}		20	–	–	
Address Set-up Time for Latch		t_{ASL}		60	–	–	ns
Address Hold Time for Latch		t_{AHL}		20	–	–	ns
Address Hold Time		t_{AH}		20	–	–	ns
Peripheral Read Access Time	Non-Multiplexed Bus	(t_{ACCN})		–	–	(610)	ns
	Multiplexed Bus	(t_{ACCM})		–	–	(600)	
Oscillator stabilization Time		t_{RC}	Fig. 11	100	–	–	ms
Processor Control Set-up Time		t_{PCS}	Fig. 12	200	–	–	ns

PERIPHERAL PORT TIMING (V_{CC} = 5.0V ±5%, V_{SS} = 0V, Ta = 0 ~ +70°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Peripheral Data Setup Time	Port 1, 2, 3, 4	t_{PDSU}	Fig. 3	200	–	–	ns
Peripheral Data Hold Time	Port 1, 2, 3, 4	t_{PDH}	Fig. 3	200	–	–	ns
Delay Time, Enable Positive Transition to $\overline{OS3}$ Negative Transition		t_{OSD1}	Fig. 5	–	–	350	ns
Delay Time, Enable Positive Transition to $\overline{OS3}$ Positive Transition		t_{OSD2}	Fig. 5	–	–	350	ns
Delay Time, Enable Negative Transition to Peripheral Data Valid	Port 1, 2*, 3, 4	t_{PWD}	Fig. 4	–	–	400	ns
Delay Time, Enable Negative Transition to Peripheral CMOS Data Valid	Port 2**, 4	t_{CMOS}	Fig. 4	–	–	2.0	μs
Input Strobe Pulse Width		t_{PWIS}	Fig. 6	200	–	–	ns
Input Data Hold Time	port 3	t_{IH}	Fig. 6	50	–	–	ns
Input Data Set-up Time	Port 3	t_{IS}	Fig. 6	20	–	–	ns

*Except P_{21} **10kΩ pull up register required for Port 2

TIMER, SCI TIMING (V_{CC} = 5.0V ±5%, V_{SS} = 0V, Ta = 0 ~ +70°C, unless otherwise noted.)

Item	Symbol	Test Condition	min	typ	max	Unit
Timer Input Pulse Width	t_{PWT}		2 t_{cyc}+200	–	–	ns
Delay Time, Enable Positive Transition to Timer Out	t_{TOD}	Fig. 7	–	–	600	ns
SCI Input Clock Cycle	t_{Scyc}		1	–	–	t_{cyc}
SCI Input Clock Pulse Width	t_{PWSCK}		0.4	–	0.6	t_{Scyc}

MODE PROGRAMMING (V_{CC} = 5.0V ±5%, V_{SS} = 0V, Ta = 0 ~ +70°C, unless otherwise noted.)

Item	Symbol	Test Condition	min	typ	max	Unit
Mode Programming Input "Low" Voltage	V_{MPL}	Fig. 8	–	–	1.7	V
Mode Programming Input "High" Voltage	V_{MPH}		4.0	–	–	V
$\overline{RES}$ "Low" Pulse Width	PW_{RSTL}		3.0	–	–	t_{cyc}
Mode Programming Set-up Time	t_{MPS}		2.0	–	–	t_{cyc}
Mode Programming Hold Time	t_{MPH}		150	–	–	ns

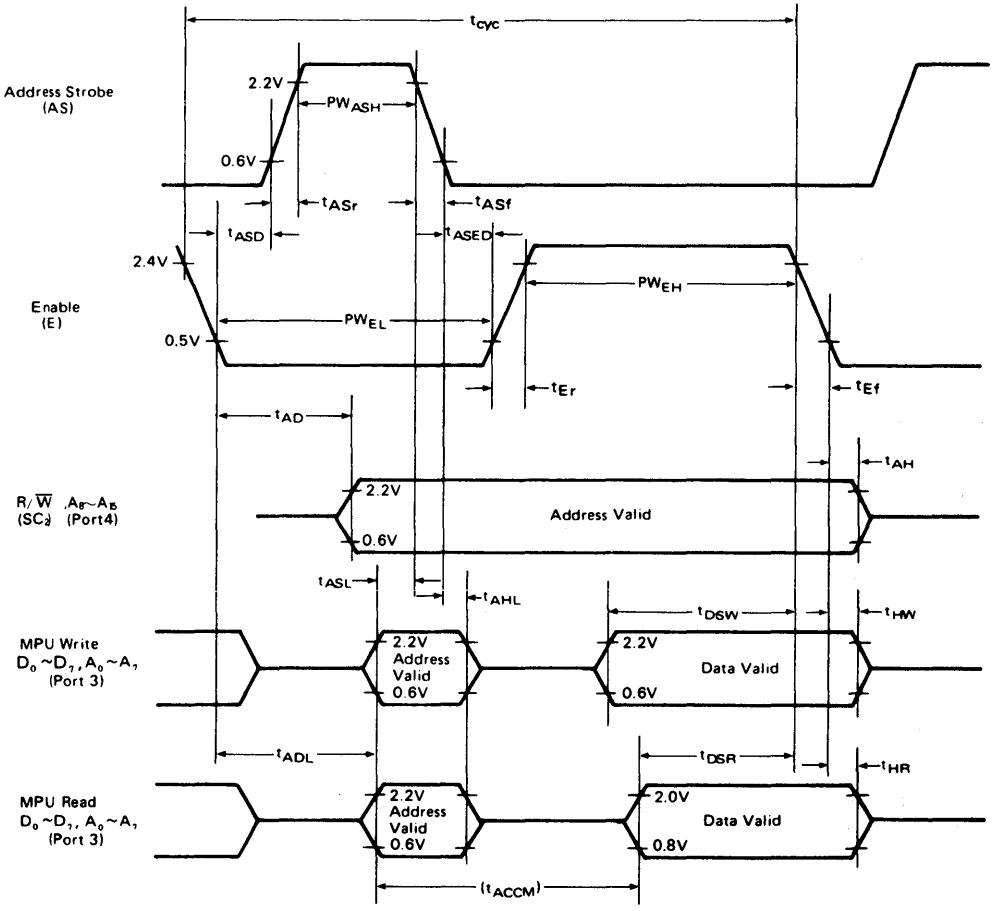


Figure 1 Expanded Multiplexed Bus Timing

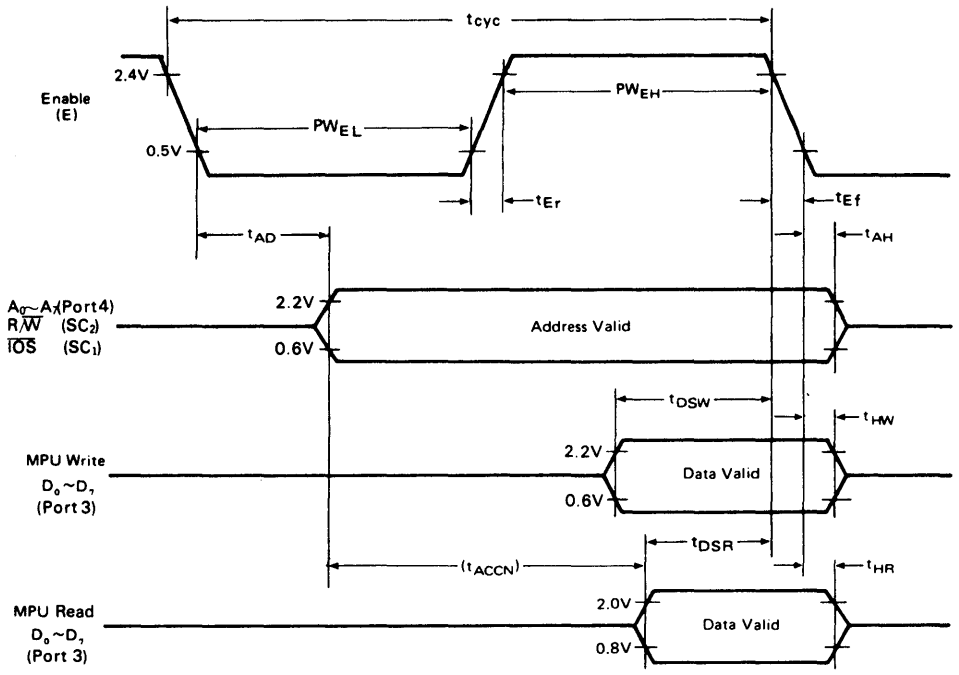
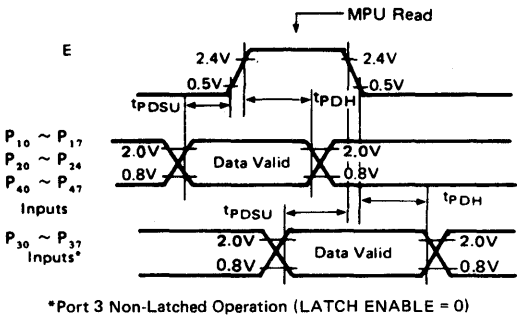
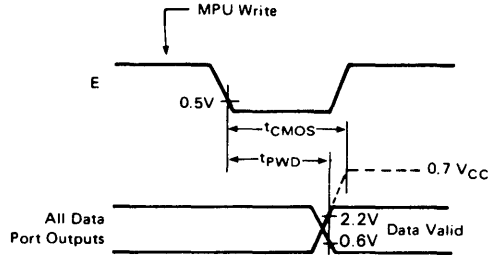


Figure 2 Expanded Non-Multiplexed Bus Timing



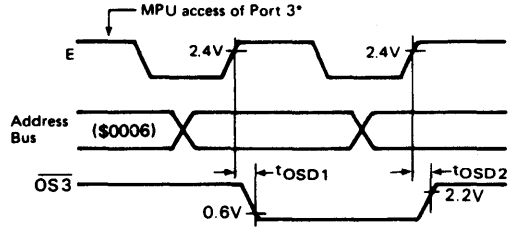
*Port 3 Non-Latched Operation (LATCH ENABLE = 0)

Figure 3 Data Set-up and Hold Times (MPU Read)



(NOTE) 1. 10 kΩ Pullup resistor required for Port 2 to reach 0.7 V_{CC}
2. Not applicable to P_{21}
3. Port 4 cannot be pulled above V_{CC}

Figure 4 Port Data Delay Timing (MPU Write)



*Access matches Output Strobe Select (OSS = 0, a read; OSS = 1, a write)

Figure 5 Port 3 Output Strobe Timing (Single Chip Mode)

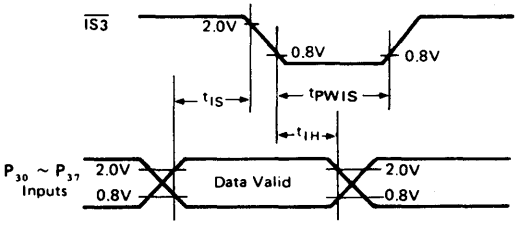


Figure 6 Port 3 Latch Timing (Single Chip Mode)

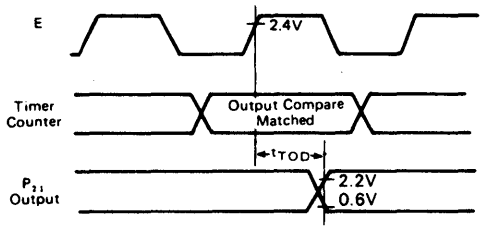


Figure 7 Timer Output Timing

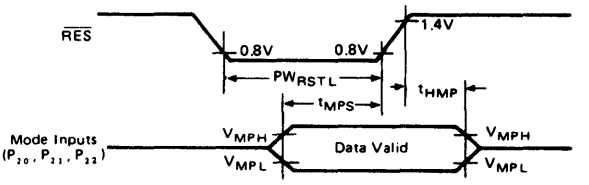
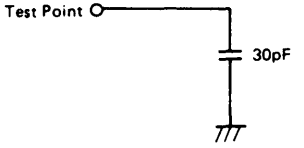
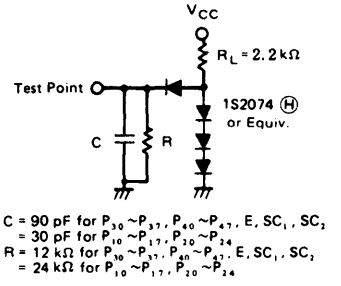


Figure 8 Mode Programming Timing



(a) CMOS Load



(b) TTL Load

Figure 9 Bus Timing Test Loads

■ INTRODUCTION

The HD68P01 is an 8-bit monolithic microcomputer which can be configured to function in a wide variety of applications. The facility which provides this extraordinary flexibility is its ability to be hardware programmed into eight different operating modes. The operating mode controls the configuration of 18 of the MCU's 40 pins, available on-chip resources, memory map, location (internal or external) of interrupt vectors, and type of external bus. The configuration of the remaining 22 pins is not dependent on the operating mode.

Twenty-nine pins are organized as three 8-bit ports and one 5-bit port. Each port consists of at least a Data Register and a write-only Data Direction Register. The Data Direction Register is used to define whether corresponding bits in the Data Register are configured as an input (clear) or output (set).

The term "port", by itself, refers to all of its associated hardware. When the port is used as a "data port" or "I/O port", it is controlled by its Data Direction Register and the programmer has direct access to its pins using the port's Data Register. Port pins are labled as Pij where i identifies one of four ports and j indicates the particular bit.

The Microprocessor Unit (MPU) is an enhanced HD6800 MPU with additional capabilities and greater throughput. It is upward source and object code compatible with the HD6800. The programming model is depicted in Figure 10 where Accumulator D is a concatenation of Accumulators A and B. A list of new operations added to the HMCS6800 instruction set are shown in Table 8.

The basic difference between the HD6801 and the HD68P01 is that the HD6801 has an on-chip ROM while the HD68P01 has an on-package EPROM. The HD68P01 is pin and code compatible with the HD6801 and can be used to emulate the HD6801, allowing easy software development using the on-package EPROM. Software developed using the HD68P01 can then be masked into the HD6801 ROM.

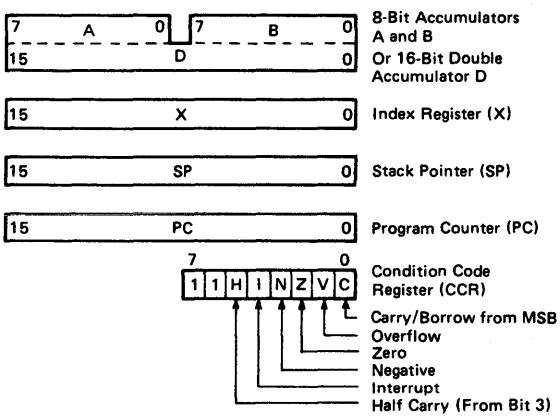


Figure 10 HD68P01 Programming Model

■ INTERRUPTS

The MCU supports two types of interrupt requests: maskable and non-maskable. A Non-Maskable Interrupt ($\overline{NMI}$) is always recognized and acted upon at the completion of the current instruction. Maskable interrupts are controlled by the Condition Code Register's I-bit and by individual enable bits. The I-bit controls all maskable interrupts. Of the maskable interrupts, there are two types: $\overline{IRQ_1}$ and $\overline{IRQ_2}$. The Programmable Timer and Serial Communications Interface use an internal $\overline{IRQ_2}$ interrupt line, as shown in BLOCK DIAGRAM. External devices (and $\overline{IS3}$) use $\overline{IRQ_1}$. An $\overline{IRQ_1}$ interrupt is serviced before $\overline{IRQ_2}$ if both are pending.

All $\overline{IRQ_2}$ interrupts use hardware prioritized vectors. The single SCI interrupt and three timer interrupts are serviced in a prioritized order where each is vectored to a separate location. All MCU interrupt vector locations are shown in Table 1.

The Interrupt flowchart is depicted in Figure 13 and is common to every MCU interrupt excluding Reset. The Program Counter, Index Register, A Accumulator, B Accumulator, and Condition Code Register are pushed to the stack. The I-bit is set to inhibit maskable interrupts and a vector is fetched corresponding to the current highest priority interrupt. The vector is transferred to the Program Counter and instruction execution is resumed. Interrupt and $\overline{RES}$ timing is illustrated in Figure 11 and 12.

Table 1 MCU Interrupt Vector Locations

MSB	LSB	Interrupt
FFFE	FFFF	$\overline{RES}$
FFFC	FFFD	$\overline{NMI}$
FFFA	FFFB	Software Interrupt (SWI)
FFF8	FFF9	$\overline{IRQ_1}$ (or IS3)
FFF6	FFF7	ICF (Input Capture)
FFF4	FFF5	OCF (Output Compare)
FFF2	FFF3	TOF (Timer Overflow)
FFF0	FFF1	SCI (RDRF + ORFE + TDRE)

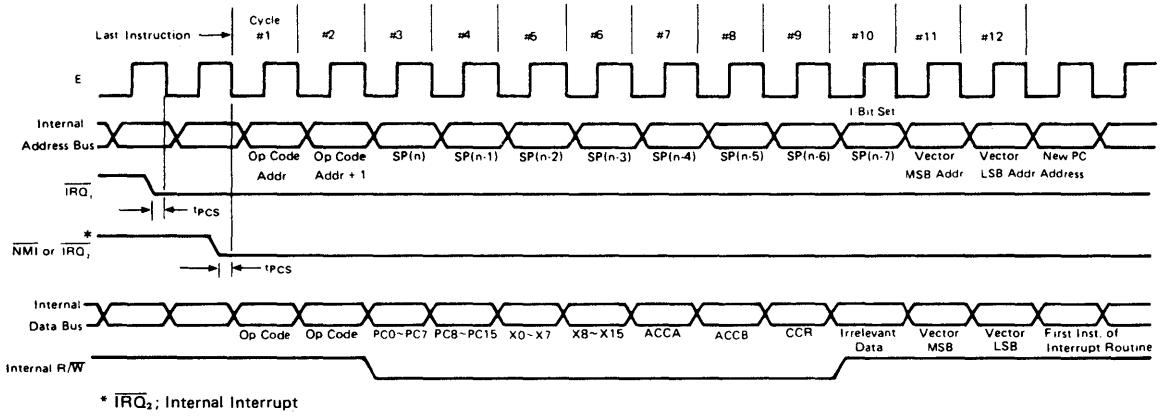


Figure 11 Interrupt Sequence

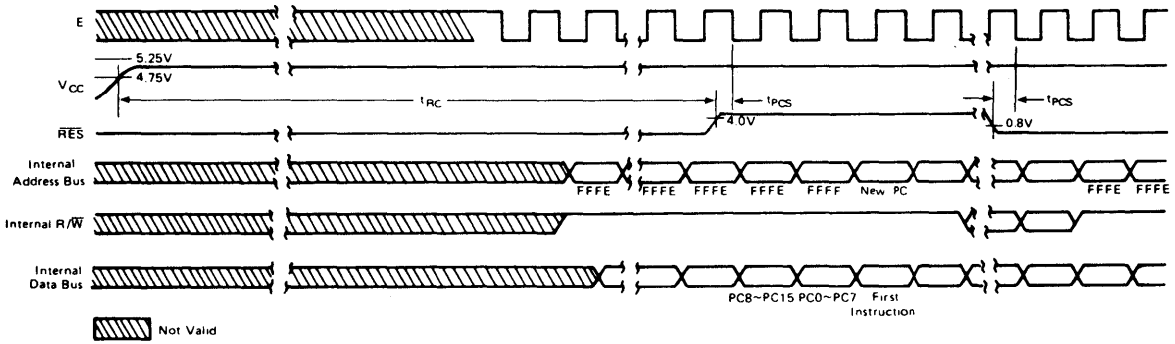


Figure 12 Reset Timing

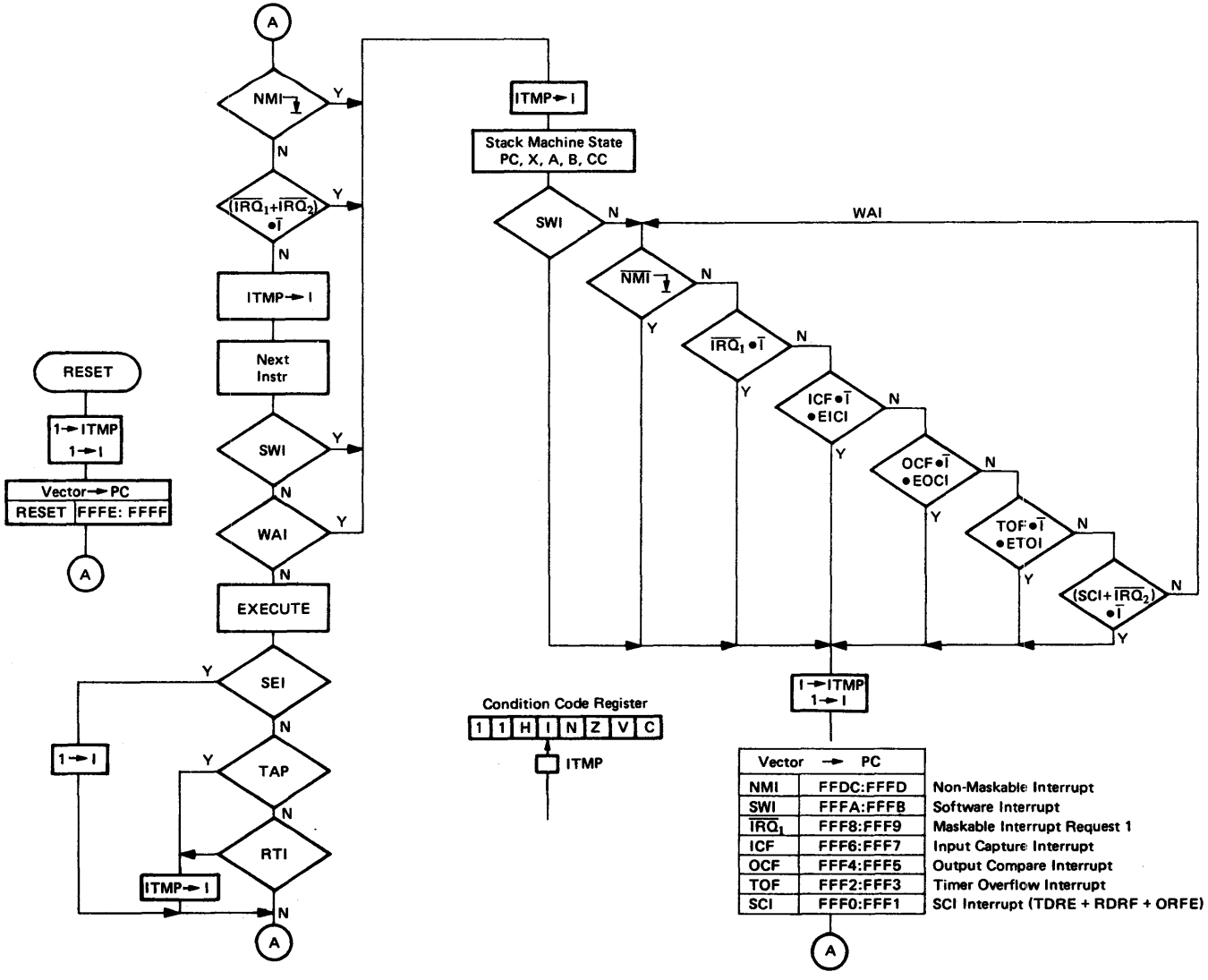


Figure 13 Interrupt Flowchart

■ FUNCTIONAL PIN DESCRIPTIONS

• V_{CC} and V_{SS}

V_{CC} and V_{SS} provide power to a large portion of the MCU. The power supply should provide +5 volts (±5%) to V_{CC}, and V_{SS} should be tied to ground. Total power dissipation (including V_{CC} Standby), will not exceed P_D milliwatts.

• V_{CC} Standby

V_{CC} Standby provides power to the standby portion ($80 through $BF) of the RAM and the STBY PWR and RAME bits of the RAM Control Register. Voltage requirements depend on whether the MCU is in a powerup or powerdown state. In the powerup state, the power supply should provide +5 volts (±5%) and must reach V_{SB} volts before $\overline{RES}$ reaches 4.0 volts. During powerdown, V_{CC} Standby must remain above V_{SBB} (min) to sustain the standby RAM and STBY PWR bit. While in powerdown operation, the standby current will not exceed I_{SBB}.

It is typical to power both V_{CC} and V_{CC} Standby from the same source during normal operation. A diode must be used between them to prevent supplying power to V_{CC} during powerdown operation. V_{CC} Standby should be tied to either ground or V_{CC} in Mode 3.

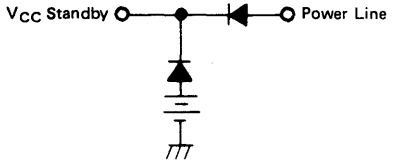


Figure 14 Battery Backup for V_{CC} Standby

• RAM Control Register ($14)

The RAM Control Register includes two bits which can be used to control RAM accesses and determine the adequacy of the standby power source during powerdown operation. It is intended that RAME be cleared and STBY PWR be set as part of a powerdown procedure.

RAM Control Register

7	6	5	4	3	2	1	0
STBY PWR	RAME	X	X	X	X	X	X

Bit 0~5 Not Used

Bit 6 RAME — RAM Enable. This Read/Write bit can be used to remove the entire RAM from the internal memory map. RAME is set (enabled) during Reset provided standby power is available on the positive edge of $\overline{RES}$. If RAME is clear, any access to a RAM address is external. If RAME is set and not in Mode 3, the RAM is included in the internal map.

Bit 7 STBY PWR — Standby Power. This bit is a Read/Write status bit which is cleared whenever V_{CC} Standby decreases below V_{SBB} (min). It can be set only by software and is not affected by $\overline{RES}$.

• XTAL and EXTAL

These two input pins interface either a crystal or TTL compatible clock to the MCU's internal clock generator. Divide-by-four circuitry is included which allows use of the inexpensive 3.58 MHz Color Burst TV crystals. A 22 pF capacitor is required from each crystal pin to ground to ensure reliable startup and operation. Alternatively, EXTAL may be driven with an external TTL compatible clock of $4f_0$ with a duty cycle of 50% (±10%) with XTAL connected to ground.

The internal oscillator is designed to interface with an AT-cut quartz crystal resonator or a ceramic resonator operated in parallel resonance mode in the frequency range specified for f_{XTAL}. The crystal should be mounted as close as possible to the input pins to minimize output distortion and startup stabilization time. The MCU is compatible with most commercially available crystals and ceramic resonators and nominal crystal parameters are shown in Figure 15.

• $\overline{RES}$

This input is used to reset the MCU's internal state and provide an orderly startup procedure. During powerup, $\overline{RES}$ must be held below 0.8 volts: (1) at least t_{RC} after V_{CC} reaches 4.75 volts in order to provide sufficient time for the clock generator to stabilize, and (2) until V_{CC} Standby reaches 4.75 volts. $\overline{RES}$ must be held low at least three E-cycles if asserted during powerup operation.

When a "High" level is detected, the MCU does the following:

1) All the higher order address lines will be forced "High".
2) I/O Port 2 bits, 2, 1, and 0 are latched into programmed control bits PC2, PC1 and PC0.
3) The last two ($FFFE, $FFFF) locations in memory will be used to load the program addressed by the program counter.
4) The interrupt mask bit is set; must be cleared before the MPU can recognize maskable interrupts.

• E (Enable)

This is an output clock used primarily for bus synchronization. It is TTL compatible and is the slightly skewed divide-by-four result of the MCU input frequency. It will drive one Schottky TTL load and 90 pF, and all data given in cycles is referenced to this clock unless otherwise noted.

• $\overline{NMI}$ (Non-Maskable Interrupt)

An $\overline{NMI}$ negative edge request an MCU interrupt sequence, but the current instruction will be completed before it responds to the request. The MCU will then begin an interrupt sequence. Finally, a vector is fetched from $FFFC and $FFFD, transferred to the Program Counter and instruction execution resumes. $\overline{NMI}$ typically requires a 3.3 kΩ (nominal) resistor to V_{CC}. There is no internal $\overline{NMI}$ pullup resistor. $\overline{NMI}$ must be held low for at least one E-cycle to be recognized under all conditions.

• $\overline{IRQ_1}$ (Maskable Interrupt Request 1)

$\overline{IRQ_1}$ is a level-sensitive input which can be used to request an interrupt sequence. The MPU will complete the current instruction before it responds to the request. If the interrupt mask bit (I-bit) in the Condition Code Register is clear, the MCU will begin an interrupt sequence. Finally, a vector is fetched from $FFF8 and $FFF9, transferred to the Program Counter, and instruction execution is resumed.

$\overline{IRQ_1}$ typically requires an external 3.3 kΩ (nominal) resistor to V_{CC} for wire-OR application. $\overline{IRQ_1}$ has no internal pullup resistor.

- **SC_1 and SC_2 (Strobe Control 1 and 2)**

The function of SC_1 and SC_2 depends on the operating mode. SC_1 is configured as an output in all modes except single chip mode, whereas SC_2 is always an output. SC_1 and SC_2 can drive one Schottky load and 90 pF.

SC_1 and SC_2 in Single Chip Mode

In Single Chip Modes, SC_1 and SC_2 are configured as an input and output, respectively, and both function as Port 3 control lines. SC_1 functions as $\overline{IS3}$ and can be used to indicate that Port 3 input data is ready or output data has been accepted. Three options associated with $\overline{IS3}$ are controlled by Port 3's Control and Status Register and are discussed in Port 3's description. If unused, $\overline{IS3}$ can remain unconnected.

SC_2 is configured as $\overline{OS3}$ and can be used to strobe output data or acknowledge input data. It is controlled by Output Strobe Select (OSS) in Port 3's Control and Status Register. The strobe is generated by a read (OSS= 0) or write (OSS = 1) to Port 3's Data Register. $\overline{OS3}$ timing is shown in Figure 5.

SC_1 and SC_2 in Expanded Non-Multiplexed Mode

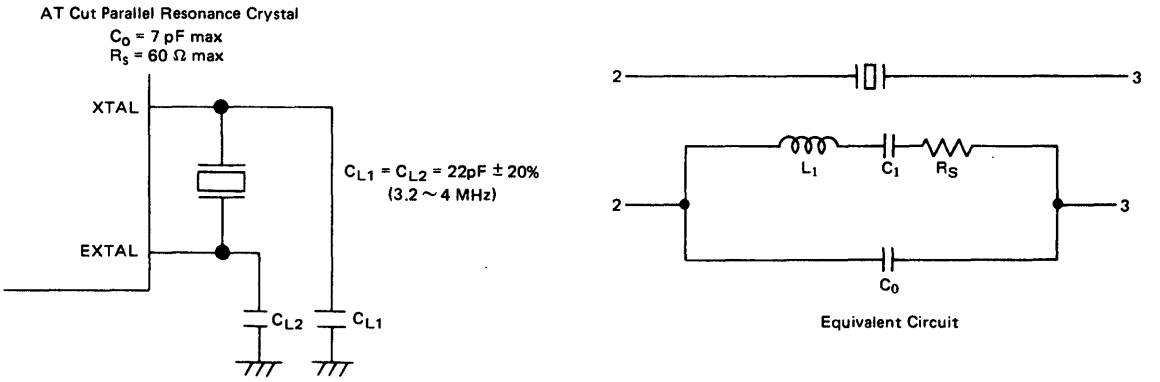
In the Expanded Non-Multiplexed Mode, both SC_1 and SC_2 are configured as outputs. SC_1 functions as Input/Output Select ($\overline{IOS}$) and is asserted only when $0100 through $01FF is sensed on the internal address bus.

SC_2 is configured as Read/Write and is used to control the direction of data bus transfers. An MPU read is enabled when Read/Write and E are high.

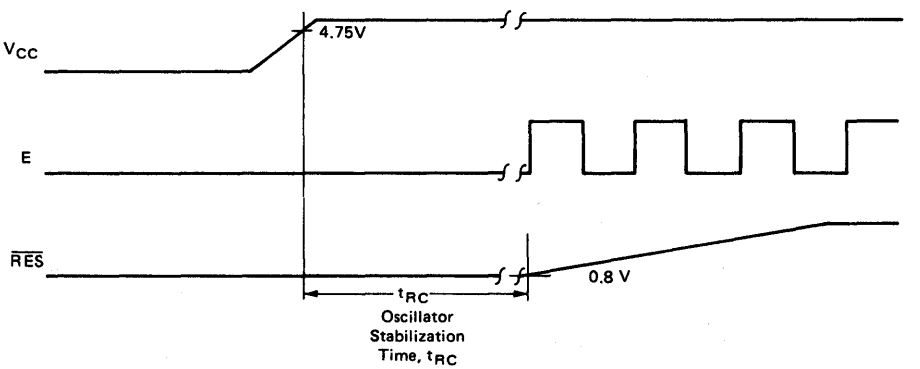
SC_1 and SC_2 in Expanded Multiplexed Mode

In the Expanded Multiplexed Modes, both SC_1 and SC_2 are configured as outputs. SC_1 functions as Address Strobe and can be used to demultiplex the eight least significant addresses and the data bus. A latch controlled by Address Strobe captures address on the negative edge, as shown in Figure 20.

SC_2 is configured as Read/Write and is used to control the direction of data bus transfers. An MPU read is enabled when Read/Write and E are high.



(a) Nominal Recommended Crystal Parameters



(b) Oscillator Stabilization Time (t_{RC})

Figure 15 Oscillator Characteristics

■ PORTS

There are four I/O ports on the MCU; three 8-bit ports and one 5-bit port. There are two control lines associated with one of the 8-bit ports. Each port has an associated write only Data Direction Register which allows each I/O line to be programmed to act as an input or an output. A "1" in the corresponding Data Direction Register bit will cause that I/O line to be an output. A "0" in the corresponding Data Direction Register bit will cause the I/O line to be an input. There are four ports: Port 1, Port 2, Port 3, and Port 4. Their addresses and the addresses of their Data Direction registers are given in Table 2.

Table 2 Port and Data Direction Register Addresses

Ports	Port Address	Data Direction Register Address
I/O Port 1	$0002	$0000
I/O Port 2	$0003	$0001
I/O Port 3	$0006	$0004
I/O Port 4	$0007	$0005

• P_{10}~P_{17} (Port 1)

Port 1 is a mode independent 8-bit I/O port where each line is an input or output as defined by its Data Direction Register. The TTL compatible three-state output buffers can drive one Schottky TTL load and 30 pF, Darlington transistors, or CMOS devices using external pullup resistors. It is configured as a data input port by $\overline{RES}$. Unused lines can remain unconnected.

• P_{20}~P_{24} (Port 2)

Port 2 is a mode independent 5-bit I/O port where each line is configured by its Data Direction Register. During $\overline{RES}$, all lines are configured as inputs. The TTL compatible three-state output buffers can drive one Schottky TTL load and 30 pF or CMOS devices using external pullup resistors. P_{20}, P_{21} and P_{22} must always be connected to provide the operating mode. If lines P_{23} and P_{24} are unused, they can remain unconnected.

P_{20}, P_{21}, and P_{22} provide the operating mode which is latched into the Program Control Register on the positive edge of $\overline{RES}$. The mode may be read from Port 2 Data Register as shown where PC2 is latched from pin 10.

Port 2 also provides an interface for the Serial Communications Interface and Timer. Bit 1, if configured as an output, is dedicated to the timer's Output Compare function and cannot be used to provide output from Port 2 Data Register.

Port 2 Data Register

7	6	5	4	3	2	1	0	
PC2	PC1	PC0	P24	P23	P22	P21	P20	$0003

• P_{30}~P_{37} (Port 3)

Port 3 can be configured as an I/O port, a bidirectional 8-bit data bus, or a multiplexed address/data bus depending on the operating mode. The TTL compatible three-state output buffers can drive one Schottky TTL load and 90 pF. Unused lines can remain unconnected.

Port 3 in Single-Chip Mode

Port 3 is an 8-bit I/O port in Single-Chip Mode where each line is configured by its Data Direction Register. There are also two lines, $\overline{IS3}$ and $\overline{OS3}$, which can be used to control Port 3 data transfers.

Three Port 3 options are controlled by the Port 3 Control and Status Register and available only in Single-Chip Mode: (1) Port 3 input data can be latched using $\overline{IS3}$ as a control signal, (2) $\overline{OS3}$ can be generated by either an MPU read or write to Port 3's Data Register, and (3) an $\overline{IRQ_1}$ interrupt can be enabled by an $\overline{IS3}$ negative edge. Port 3 latch timing is shown in Figure 6.

Port 3 Control and Status Register

7	6	5	4	3	2	1	0	
$\overline{IS3}$ Flag	$\overline{IS3}$ $\overline{IRQ_1}$ Enable	X	OSS	Latch Enable	X	X	X	$000F

Bit 0~2 Not used.

Bit 3 LATCH ENABLE. This bit controls the input latch for Port 3. If set, input data is latched by an $\overline{IS3}$ negative edge. The latch is transparent after a read of Port 3's Data Register. LATCH ENABLE is cleared by $\overline{RES}$.

Bit 4 OSS (Output Strobe Select). This bit determines whether $\overline{OS3}$ will be generated by a read or write of Port 3's Data Register. When clear, the strobe is generated by a read; when set, it is generated by a write. OSS is cleared by $\overline{RES}$.

Bit 5 Not used.

Bit 6 $\overline{IS3}$ $\overline{IRQ_1}$ ENABLE. When set, an $\overline{IRQ_1}$ interrupt will be enabled whenever $\overline{IS3}$ FLAG is set; when clear, the interrupt is inhibited. This bit is cleared by $\overline{RES}$.

Bit 7 $\overline{IS3}$ FLAG. This read-only status bit is set by an $\overline{IS3}$ negative edge. It is cleared by a read of the Port 3 Control and Status Register (with $\overline{IS3}$ FLAG set) followed by a read or write to Port 3's Data Register or by $\overline{RES}$.

Port 3 in Expanded Non-Multiplexed Mode

Port 3 is configured as a bidirectional data bus (D_0~D_7) in the Expanded Non-Multiplexed Mode. The direction of data transfers is controlled by Read/Write (SC_2) and clocked by E (Enable).

Port 3 in Expanded Multiplexed Mode

Port 3 is configured as a time multiplexed address (A_0~A_7) and data bus (D_0~D_7) in Expanded Multiplexed Mode where Address Strobe (AS) can be used to demultiplex the two buses. Port 3 is held in a high impedance state between valid address and data to prevent potential bus conflicts.

• P_{40}~P_{47} (Port 4)

Port 4 is configured as an 8-bit I/O port, address outputs, or data inputs depending on the operating mode. Port 4 can drive one Schottky TTL load and 90 pF and is the only port with internal pullup resistors. Unused lines can remain unconnected.

Port 4 in Single Chip Mode

In Single Chip Mode, Port 4 functions as an 8-bit I/O port where each line is configured by its Data Direction Register.

Internal pullup resistors allow the port to directly interface with CMOS at 5 volt levels. External pullup resistors to more than 5 volts, however, cannot be used.

Port 4 in Expanded Non-Multiplexed Mode

Port 4 is configured from $\overline{RES}$ as an 8-bit input port where its Data Direction Register can be written to provide any or all of address lines, A_0 to A_7. Internal pullup resistors are intended to pull the lines high until its Data Direction Register is configured.

Port 4 in Expanded Multiplexed Mode

In all Expanded Multiplexed modes except Mode 6, Port 4 functions as half of the address bus and provides A_8 to A_{15}. In Mode 6, the port is configured from $\overline{RES}$ as an 8-bit parallel input port where its Data Direction Register can be written to provide any or all of address lines, A_8 to A_{15}. Internal pullup resistors are intended to pull the lines high until its Data Direction Register is configured where bit 0 controls A_8.

■ OPERATING MODES

The MCU provides eight different operating modes which are selectable by hardware programming and referred to as Mode 0 through Mode 7. The operating mode controls the memory map, configuration of Port 3, Port 4, SC_1, SC_2, and the physical location of interrupt vectors.

• Fundamental Modes

The MCU's eight modes can be grouped into three fundamental modes which refer to the type of bus it supports: Single Chip, Expanded Non-Multiplexed, and Expanded Multiplexed. Single chip modes include 4 and 7, Expanded Non-Multiplexed is Mode 5 and the remaining five are Expanded Multiplexed modes. Table 3 summarizes the characteristics of the operating modes.

Single Chip Modes (4, 7)

In Single-Chip Mode, the MCU's four ports are configured as parallel input/output data ports, as shown in Figure 16. The MCU functions as a monolithic microcomputer in these two modes without external address or data buses. A maximum of 29 I/O lines and two Port 3 control lines are provided. In addition to other peripherals, another MCU can be interfaced to Port 3 in a loosely coupled dual processor configuration, as shown in Figure 17.

In Single-Chip Test Mode (4), the RAM responds to $XX80 through $XXFF and the ROM is removed from the internal address map. A test program must first be loaded into the RAM using modes 0, 1, 2, or 6. If the MCU is Reset and then programmed into Mode 4, execution will begin at $XXFE: XXFF. Mode 5 can be irreversibly entered from Mode 4 without going through Reset by setting bit 5 of Port 2's Data Register. This mode is used primarily to test Ports 3 and 4 in the Single-Chip and Non-Multiplexed Modes.

Expanded Non-Multiplexed Mode (5)

A modest amount of external memory space is provided in the Expanded Non-Multiplexed Mode while retaining significant on-chip resources. Port 3 functions as an 8-bit bidirectional data bus and Port 4 is configured as an input data port. Any combination of the eight least-significant address lines may be obtained by writing to Port 4's Data Direction Register. Stated alternatively, any combination of A_0 to A_7 may be provided while retaining the remainder as input data lines. Internal pullup resistors are intended to pull Port 4's lines high until it is configured.

Figure 18 illustrates a typical system configuration in the Expanded Non-Multiplexed Mode. The MCU interfaces directly with HMCS6800 family parts and can access 256 bytes of external address space at $100 through $1FF. $\overline{IOS}$ provides an address decode of external memory ($100–$1FF) and can be used similarly to an address or chip select line.

Table 3 Summary of HD6800 Operating Modes

Common to all Modes:
Reserved Register Area
Port 1
Port 2
Programmable Timer
Serial Communication Interface
Single Chip Mode 7
128 bytes of RAM; 2048 bytes of ROM
Port 3 is a parallel I/O port with two control lines
Port 4 is a parallel I/O port
SC_1 is Input Strobe 3 ($\overline{IS3}$)
SC_2 is Output Strobe 3 ($\overline{OS3}$)
Expanded Non-Multiplexed Mode 5
128 bytes of RAM; 2048 bytes of ROM
256 bytes of external memory space
Port 3 is an 8-bit data bus
Port 4 is an input port/address bus
SC_1 is Input/Output Select ($\overline{IOS}$)
SC_2 is read/write (R/$\overline{W}$)
Expanded Multiplexed Modes 1, 2, 3, 6
Four memory space options (65k address space):
(1) No internal RAM or ROM (Mode 3)
(2) Internal RAM, no ROM (Mode 2)
(3) Internal RAM and ROM (Mode 1)
(4) Internal RAM, ROM with partial address bus (Mode 6)
Port 3 is a multiplexed address/data bus
Port 4 is an address bus (inputs/address in Mode 6)
SC_1 is Address Strobe (AS)
SC_2 is Read/Write (R/$\overline{W}$)
Test Modes 0 and 4
Expanded Multiplexed Test Mode 0
May be used to test RAM and ROM
Single Chip and Non-Multiplexed Test Mode 4
(1) May be changed to Mode 5 without going through Reset
(2) May be used to test Ports 3 and 4 as I/O ports

Expanded-Multiplexed Modes (0, 1, 2, 3, 6)

In the Expanded-Multiplexed Modes, the MCU has the ability to access a 65k bytes memory space. Port 3 functions as a time multiplexed address/data bus with address valid on the negative edge of Address Strobe (AS) and the data bus valid while E is high. In Modes 0 to 3, Port 4 provides address lines A_8 to A_{15}. In Mode 6, however, Port 4 is configured during $\overline{RES}$ as data port inputs and the Data Direction Register can be changed to provide any combination of address lines, A_8 to A_{15}. Stated alternatively, any subset of A_8 to A_{15} can be provided while retaining the remainder as input data lines. Internal pullup resistors are intended to pull Port 4's lines high until software configures the port.

Figure 19 depicts a typical configuration for the Expanded-Multiplexed Modes. Address Strobe can be used to control a transparent D-type latch to capture addresses A_0 to A_7, as shown in Figure 20. This allows Port 3 to function as a Data Bus when E is high.

In Mode 0, the Reset vector is external for the first two E-cycles after the positive edge of $\overline{RES}$ and internal thereafter. In

addition, the internal and external data buses are connected and there must be no memory map overlap to avoid potential bus conflicts. Mode 0 is used primarily to verify the ROM pattern and monitor the internal data bus with the automated test equipment.

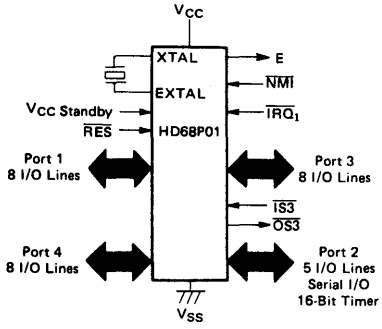


Figure 16 Single Chip Mode

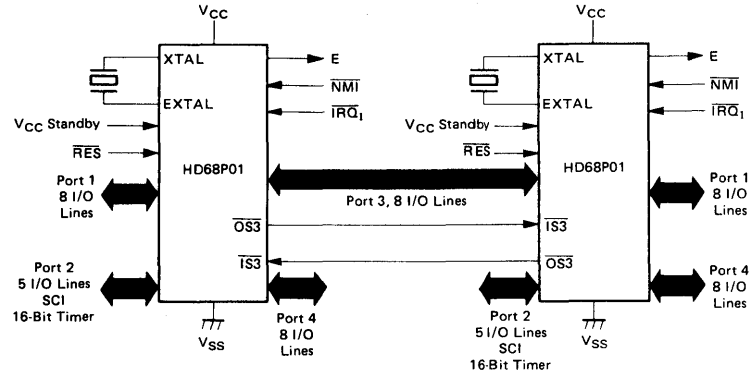
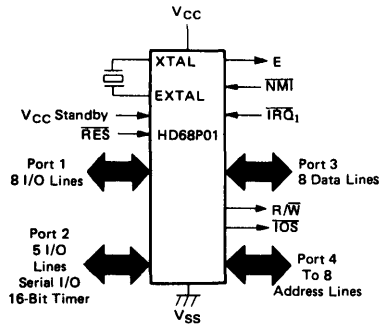


Figure 17 Single Chip Dual Processor Configuration



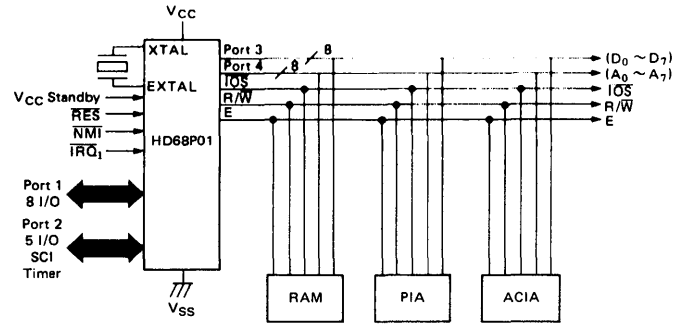
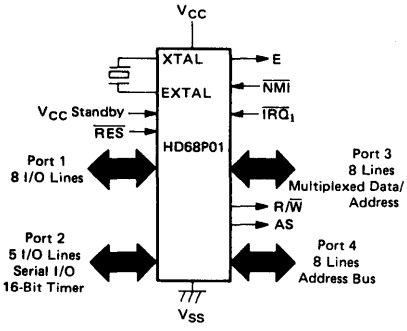


Figure 18 Expanded Non-Multiplexed Configuration



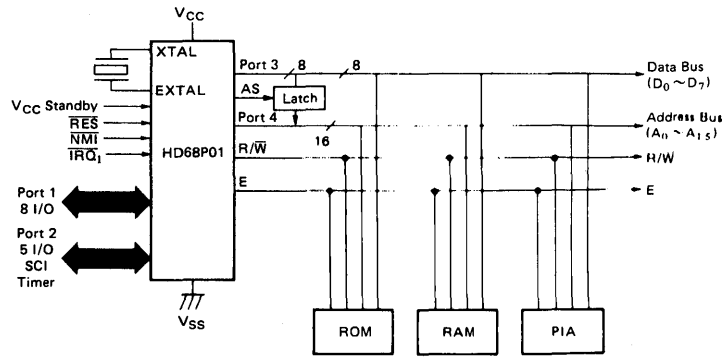
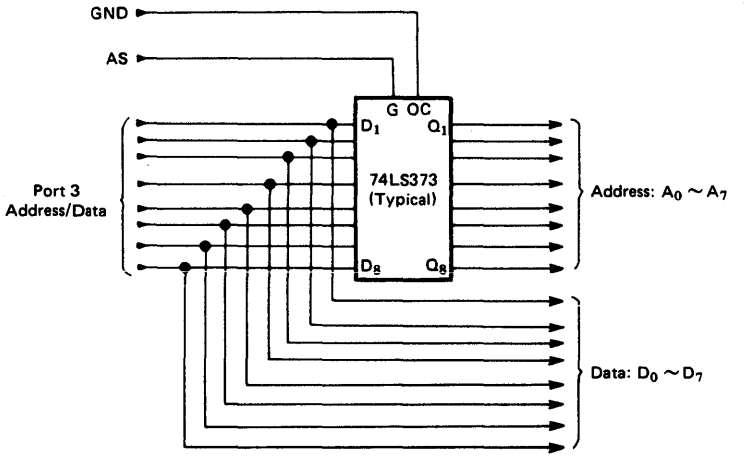


Figure 19 Expanded Multiplexed Configuration



Function Table

Output Control	Enable G	D	Output Q
L	H	H	H
L	H	L	L
L	L	X	Q_0
H	X	X	Z

Figure 20 Typical Latch Arrangement

- **Programming The Mode**

The operating mode is programmed by the levels asserted on P_{22}, P_{21}, and P_{20} which are latched into PC2, PC1, and PC0 of the program control register on the positive edge of $\overline{RES}$. The operating mode may be read from Port 2 Data Register as shown below, and programming levels and timing must be met as shown in Figure 8. A brief outline of the operating modes is shown in Table 4.

Port 2 Data Register

7	6	5	4	3	2	1	0	
PC2	PC1	PC0	P24	P23	P22	P21	P20	$0003

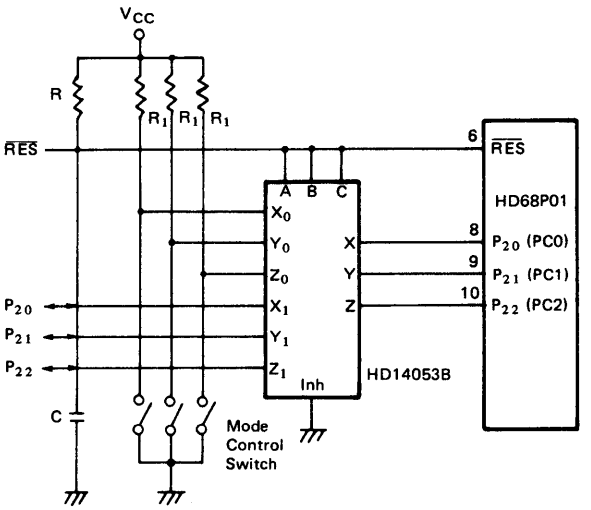
Circuitry to provide the programming levels is dependent primarily on the normal system usage of the three pins. If configured as outputs, the circuit shown in Figure 21 may be used; otherwise, three-state buffers can be used to provide isolation while programming the mode.

Table 4 Mode Selection Summary

Mode	P_{22} (PC2)	P_{21} (PC1)	P_{20} (PC0)	ROM	RAM	Interrupt Vectors	Bus Mode	Operating Mode
7	H	H	H	I	I	I	I	Single Chip
6	H	H	L	I	I	I	MUX(5, 6)	Multiplexed/Partial Decode
5	H	L	H	I	I	I	NMUX(5, 6)	Non-Multiplexed/Partial Decode
4	H	L	L	I(2)	I(1)	I	I	Single Chip Test
3	L	H	H	E	E	E	MUX(4)	Multiplexed /No RAM or ROM
2	L	H	L	E	I	E	MUX(4)	Multiplexed /RAM
1	L	L	H	I	I	E	MUX(4)	Multiplexed/RAM & ROM
0	L	L	L	I	I	I(3)	MUX(4)	Multiplexed Test

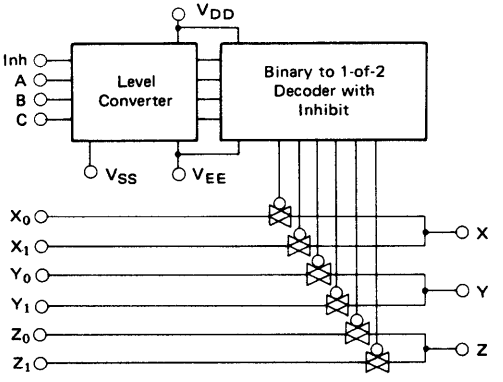
Legend:
I – Internal
E – External
MUX – Multiplexed
NMUX – Non-Multiplexed
L – Logic "0"
H – Logic "1"

Notes:
(1) Internal RAM is addressed at $XX80
(2) Internal ROM is disabled
(3) $\overline{RES}$ vector is external for 2 cycles after $\overline{RES}$ goes high
(4) Addresses associated with Ports 3 and 4 are considered external in Modes 0, 1, 2, and 3
(5) Addresses associated with Port 3 are considered external in Modes 5 and 6
(6) Port 4 default is user data input; address output is optional by writing to Port 4 Data Direction Register



[NOTES] 1) Mode 7 as shown
2) RC ≈ Reset time constant
3) R_1 = 10kΩ

Figure 21 Recommended Circuit for Mode Selection



Truth Table

Control Input				On Switch		
Inhibit	Select					
	C	B	A	HD14053B		
0	0	0	0	Z_0	Y_0	X_0
0	0	0	1	Z_0	Y_0	X_1
0	0	1	0	Z_0	Y_1	X_0
0	0	1	1	Z_0	Y_1	X_1
0	1	0	0	Z_1	Y_0	X_0
0	1	0	1	Z_1	Y_0	X_1
0	1	1	0	Z_1	Y_1	X_0
0	1	1	1	Z_1	Y_1	X_1
1	X	X	X	–		

Figure 22 HD14053B Multiplexers/Demultiplexers

■ MEMORY MAPS

The MCU can provide up to 65k byte address space depending on the operating mode. The HD68P01 provides 8k byte address space for EPROM, but the maps differ in EPROM types as follows.

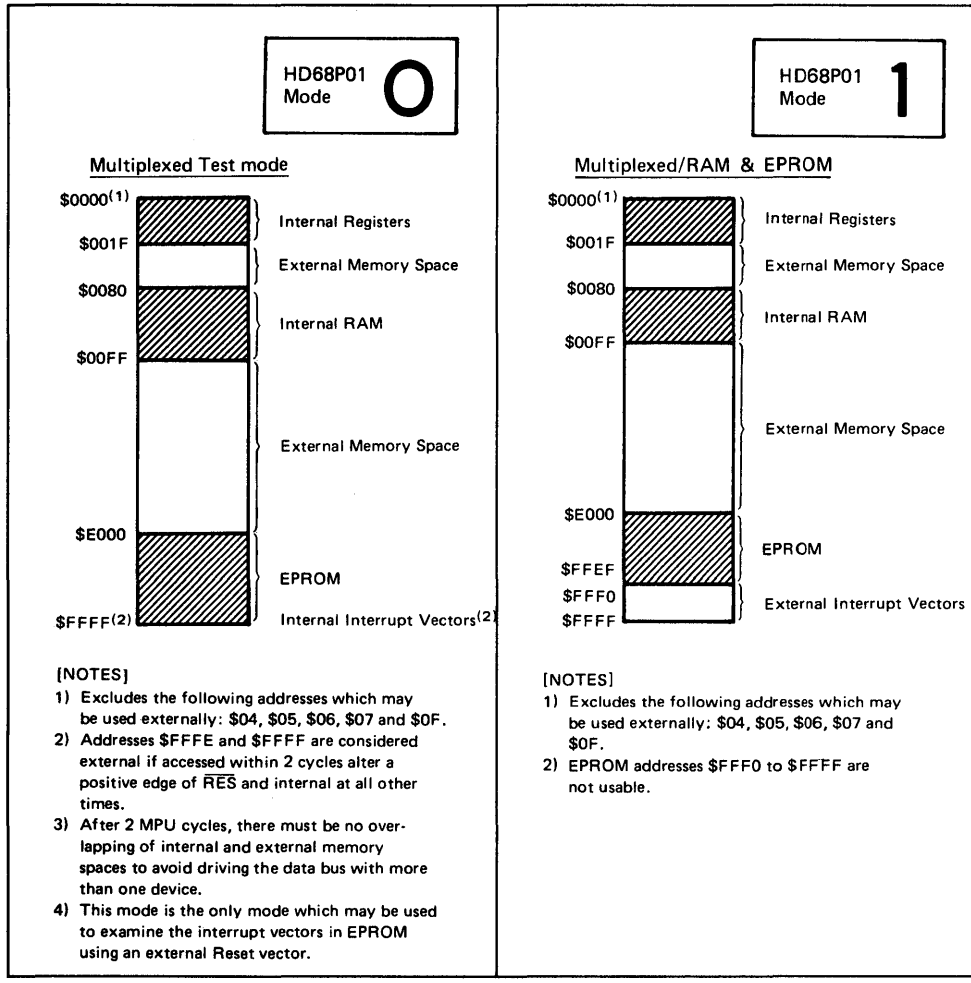
1) HN462716 (a 2k byte ROM)
   In order to support the HD6801S0, EPROM of the HD68P01S0 must be located at $F800-$FFFF.
2) HN462532, HN462732 (a 4k byte ROM)
   In order to support the HD6801V0 EPROM of the HD68P01V05 and the HD68P01V07 must be located at $F000-$FFFF.
3) HN482764 (a 8k byte ROM)
   The HD68P01 can provide up to 8k byte address space using HN482764 instead of HN462732. In this case, EPROM of the HD68P01 is located at $E000-$FFFF.

A memory map for each operating mode is shown in Figure 23. The first 32 locations of each map are reserved for the MCU's internal register area, as shown in Table 5, with exceptions as indicated.

Table 5 Internal Register Area

Register	Address
Port 1 Data Direction Register***	00
Port 2 Data Direction Register***	01
Port 1 Data Register	02
Port 2 Data Register	03
Port 3 Data Direction Register***	04*
Port 4 Data Direction Register***	05**
Port 3 Data Register	06*
Port 4 Data Register	07**
Timer Control and Status Register	08
Counter (High Byte)	09
Counter (Low Byte)	0A
Output Compare Register (High Byte)	0B
Output Compare Register (Low Byte)	0C
Input Capture Register (High Byte)	0D
Input Capture Register (Low Byte)	0E
Port 3 control and Status Register	0F*
Rate and Mode Control Register	10
Transmit/Receive Control and Status Register	11
Receive Data Register	12
Transmit Data Register	13
RAM Control Register	14
Reserved	15-1F

* External address in Modes 0, 1, 2, 3, 5, 6; cannot be accessed in Mode 5 (No $\overline{IOS}$)
** External addresses in Modes 0, 1, 2, 3
*** 1 = Output, 0 = Input



[NOTES]
1) Excludes the following addresses which may be used externally: $04, $05, $06, $07 and $0F.
2) Addresses $FFFE and $FFFF are considered external if accessed within 2 cycles alter a positive edge of $\overline{RES}$ and internal at all other times.
3) After 2 MPU cycles, there must be no overlapping of internal and external memory spaces to avoid driving the data bus with more than one device.
4) This mode is the only mode which may be used to examine the interrupt vectors in EPROM using an external Reset vector.

[NOTES]
1) Excludes the following addresses which may be used externally: $04, $05, $06, $07 and $0F.
2) EPROM addresses $FFF0 to $FFFF are not usable.

Figure 23 HD68P01 Memory Maps

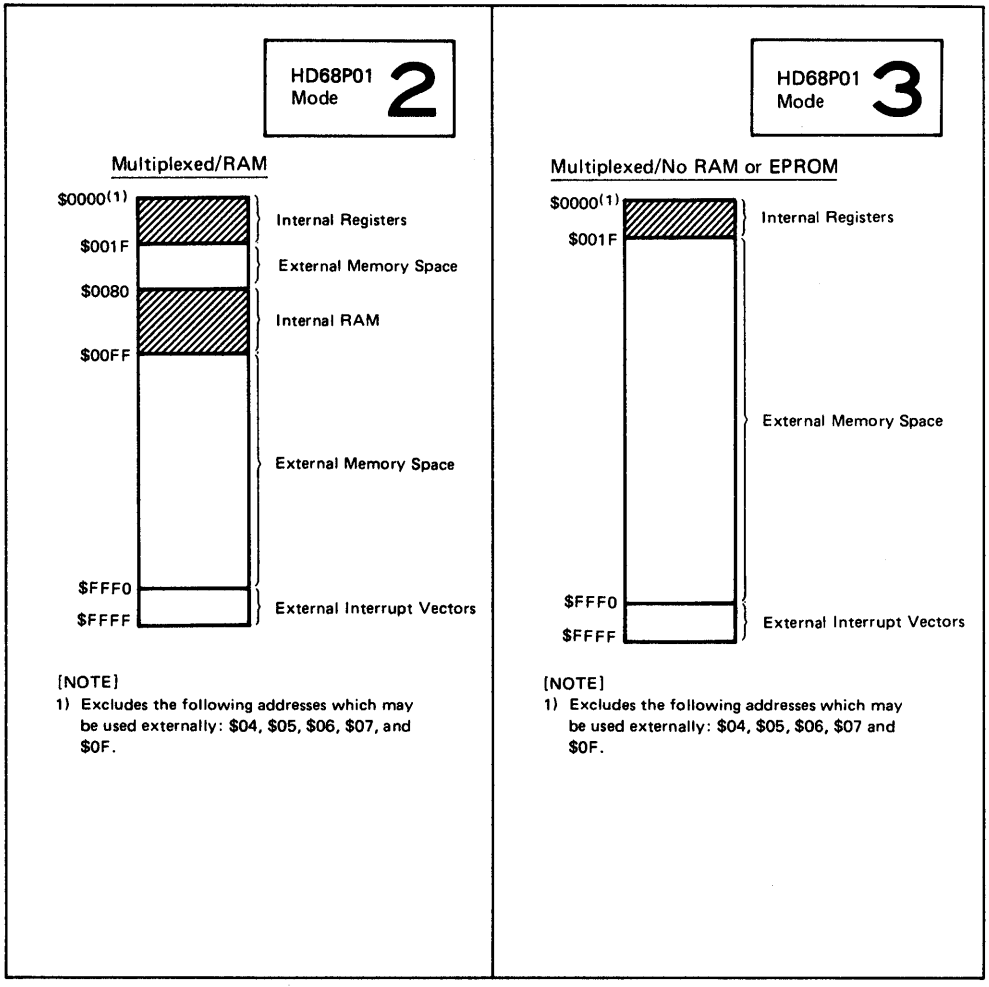


Figure 23 HD68P01 Memory Maps (Continued)

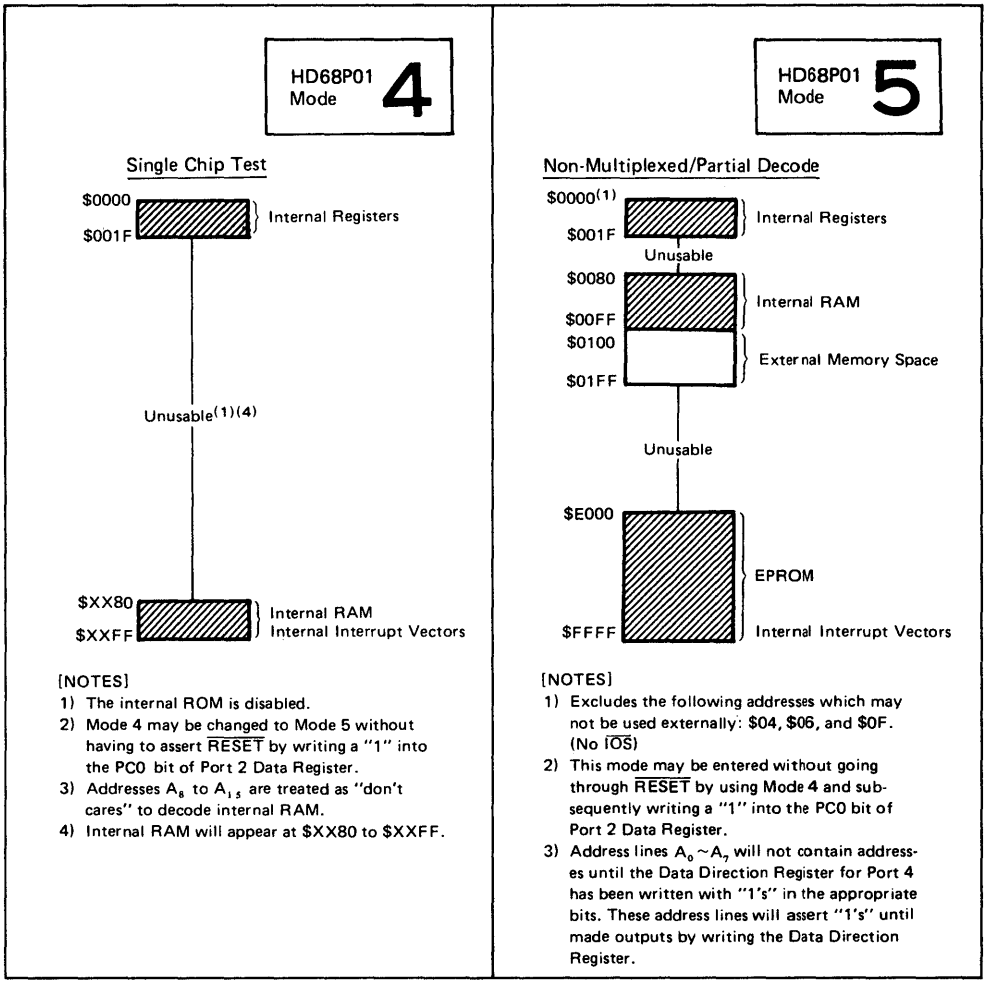
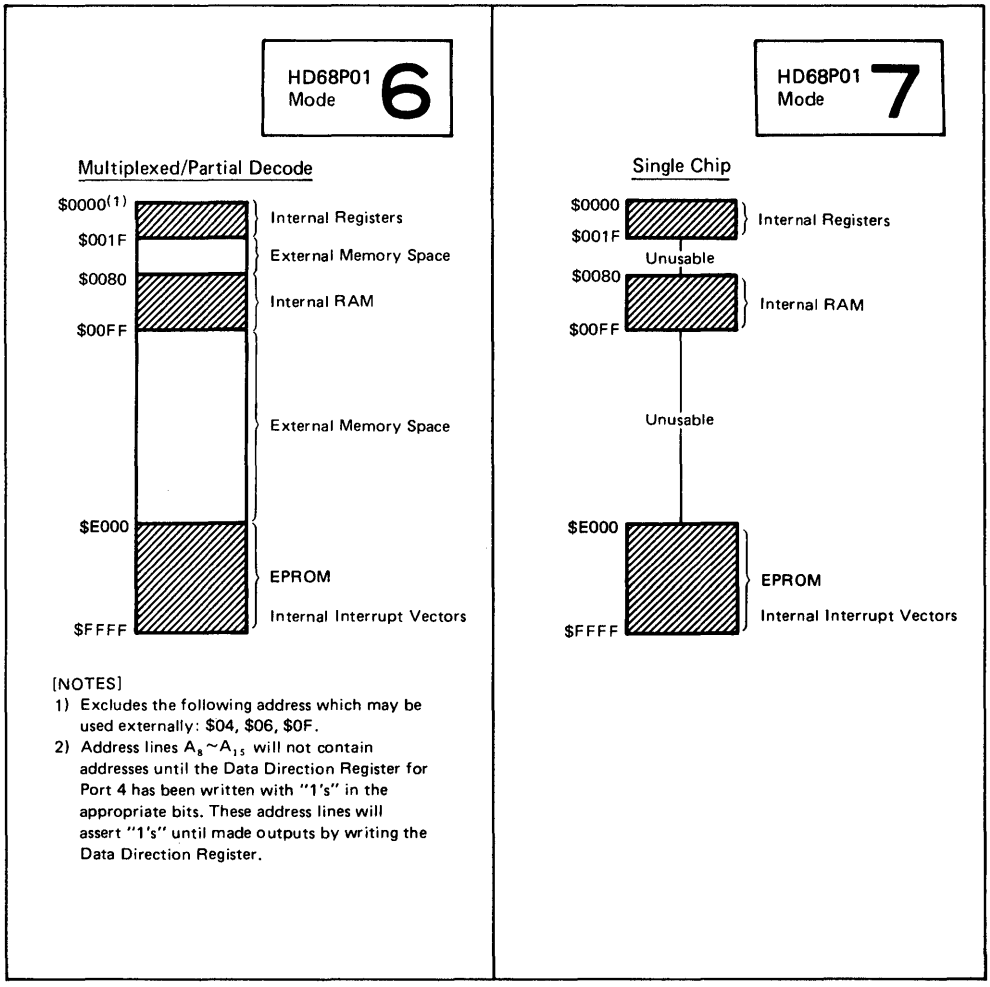


Figure 23 HD68P01 Memory Maps (Continued)



[NOTES]
1) Excludes the following address which may be used externally: $04, $06, $0F.
2) Address lines $A_8 \sim A_{15}$ will not contain addresses until the Data Direction Register for Port 4 has been written with "1's" in the appropriate bits. These address lines will assert "1's" until made outputs by writing the Data Direction Register.

Figure 23 HD68P01 Memory Maps (Continued)

■ PROGRAMMABLE TIME

The Programmable Timer can be used to perform input waveform measurements while independently generating an output waveform. Pulse widths can vary from several microseconds to many seconds. A block diagram of the Timer is shown in Figure 24.

• Counter ($09:0A)

The key timer element is a 16-bit free-running counter which is incremented by E (Enable). It is cleared during $\overline{RES}$ and is read-only with one exception: a write to the counter ($09) will preset it to $FFF8. This feature, intended for testing, can disturb serial operations because the counter provides the SCI's internal bit rate clock. TOF is set whenever the counter contains all 1's.

• Output Compare Register ($0B:0C)

The Output Compare Register is a 16-bit Read/Write register used to control an output waveform or provide an arbitrary timeout flag. It is compared with the free-running counter on each E-cycle. When a match is found, OCF is set and OLVL is clocked to an output level register. If Port 2, bit 1, is configured as an output, OLVL will appear at P_{21} and the Output Compare Register and OLVL can then be changed for the next compare. The function is inhibited for one cycle after a write to its high byte of the Compare Resister ($0B) to ensure a valid compare. The Output Compare Register is set to $FFFF by $\overline{RES}$.

• Input Capture Register ($0D: 0E)

The Input Capture Register is a 16-bit read-only register used to store the free-running counter when a "proper" input transition occurs as defined by IEDG. Port 2, bit 0 should be configured as an input, but the edge detect circuit always senses P_{20} even when configured as an output. An input capture can occur independently of ICF: the register always contains the most current value. Counter transfer is inhibited, however, between accesses of a double byte MPU read. The input pulse width must be at least two E-cycles to ensure an input capture under all conditions.

• Timer Control and Status Register ($08)

The Timer Control and Status Register (TCSR) is an 8-bit register of which all bits are readable while bits 0~4 can be written. The three most significant bits provide the timer's status and indicate if:

- a proper level transition has been dtected,
- a match has been found between the free-running counter and the output compare register, and
- the free-running counter has overflowed.

Each of the three events can generate an $\overline{IRQ_2}$ interrupt and is controlled by an individual enable bit in the TCSR.

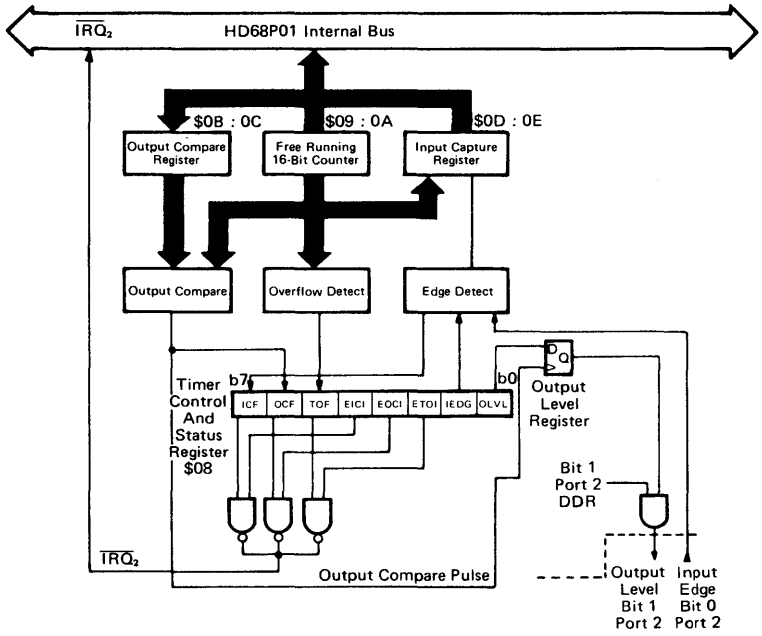


Figure 24 Block Diagram of Programmable Timer

Timer Control and Status Register (TCSR)

7	6	5	4	3	2	1	0	
ICF	OCF	TOF	EICI	EOCI	ETOI	IEDG	OLVL	$0008

Bit 0 OLVL Output level. OLVL is clocked to the output level register by a successful output compare and will appear at P_{21} if Bit 1 of Port 2's Data Direction Register is set. It is cleared by $\overline{RES}$.

Bit 1 IEDG Input Edge. IEDG is cleared by $\overline{RES}$ and controls which level transition will trigger a counter transfer to the Input Capture Register:
IEDG = 0 Transfer on a negative-edge
IEDG = 1 Transfer on a positive-edge.

Bit 2 ETOI Enable Timer Overflow Interrupt. When set, an $\overline{IRQ_2}$ interrupt is enabled for a timer overflow; when clear, the interrupt is inhibited. It is cleared by $\overline{RES}$.

Bit 3 EOCI Enable Output Compare Interrupt. When set, an $\overline{IRQ_2}$ interrupt is enabled for an output compare; when clear, the interrupt is inhibited. It is cleared by $\overline{RES}$.

Bit 4 EICI Enable Input Capture Interrupt. When set, an $\overline{IRQ_2}$ interrupt is enabled for an input capture; when clear, the interrupt is inhibited. It is cleared by $\overline{RES}$.

Bit 5 TOF Timer Overflow Flag. TOF is set when the counter contains all 1's. It is cleared by reading the TCSR (with TOF set) followed by the counter's high byte ($09), or by $\overline{RES}$.

Bit 6 OFC Output Compare Flag. OCF is set when the Output Compare Register matches the free-running counter. It is cleared by reading the TCSR (with OCF set) and then writing to the Output Compare Register ($0B or $0C), or by $\overline{RES}$.

Bit 7 ICF Input Capture Flag. ICF is set to indicate a proper level transition; it is cleared by reading the TCSR (with ICF set) and then the Input Capture Register High Byte ($0D), or by $\overline{RES}$.

■ SERIAL COMMUNICATIONS INTERFACE (SCI)

A full-duplex asynchronous Serial Communications Interface (SCI) is provided with a data format and a variety of rates. The SCI transmitter and receiver are functionally independent, but use the same data format and bit rate. Serial data format is standard mark/space (NRZ) and provides one start bit, eight data bits, and one stop bit. "Baud" and "bit rate" are used synonymously in the following description.

• Wake-Up Feature

In a typical serial loop multi-processor configuration, the software protocol will usually identify the addressee(s) at the beginning of the message. In order to permit uninterested MPU's to ignore the remainder of the message, a wake-up feature is included whereby all further SCI receiver flag (and interrupt) processing can be inhibited until its data line goes idle. An SCI receiver is re-enabled by an idle string of ten consecutive 1's or by $\overline{RES}$. Software must provide for the required idle string between consecutive messages and prevent it within messages.

• Programmable Options

The following features of the SCI are programmable:

- format: Standard mark/space (NRZ)
- clock: external or internal bit rate clock
- Baud (or bit rate): one of 4 per E-clock frequency, or external bit rate (X8) input
- wake-up feature: enabled or disabled
- interrupt requests: enabled individually for transmitter and receiver
- clock output: internal bit rate clock enabled or disabled to P_{22}
- Port 2 (bit 3, 4): dedicated or not dedicated to serial I/O individually for transmitter and receiver.

• Serial Communications Registers

The Serial Communications Interface includes four addressable registers as depicted in Figure 25. It is controlled by the Rate and Mode Control Register and the Transmit/Receive Control and Status Register. Data is transmitted and received utilizing a write-only Transmit Register and a read-only Receive Register. The shift registers are not accessible to software.

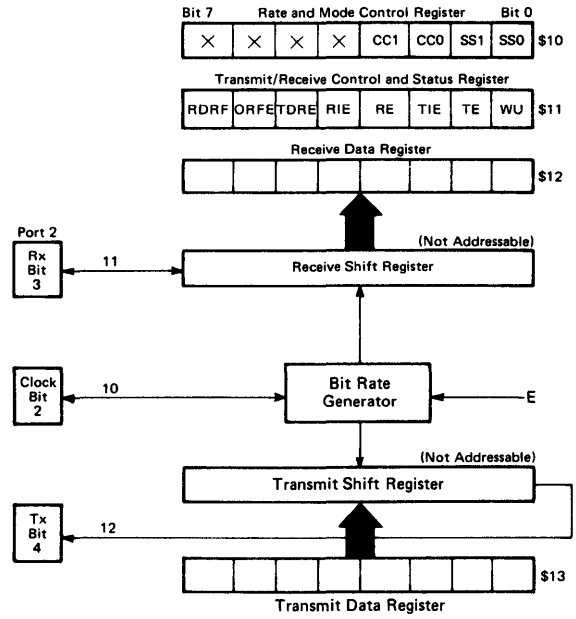


Figure 25 SCI Registers

Rate and Mode Control Register (RMCR) ($10)

The Rate and Mode Control Register controls the SCI bit rate, format, clock source, and under certain conditions, the configuration of P_{22}. The register consists of four write-only bits which are cleared by $\overline{RES}$. The two least significant bits control the bit rate of the internal clock and the remaining two bits control the format and clock source.

Rate and Mode Control Register (RMCR)

7	6	5	4	3	2	1	0	
X	X	X	X	CC1	CC0	SS1	SS0	$0010

Bit 1: Bit 0 SS1: SS0 Speed Select. These two bits select the Baud when using the internal clock. Four rates may be selected which are a function of the MCU input frequency. Table 6 lists bit time and rates for three selected MCU frequencies.

Nit 3: Bit 2 CC1:CC0 Clock Control Select. These two bits select the serial clock source. If CC1 is set, the DDR value for P_{22} is forced to the complement of CC0 and cannot be altered until CC1 is cleared. If CC1 is cleared after having been set, its DDR value is unchanged. Table 7 defines the clock source, and use of P_{22}.

If both CC1 and CC0 are set, an external TTL compatible clock must be connected to P_{22} at eight times (8X) the desired bit rate, but not greater than E, with a duty cycle of 50% (± 10%). If CC1:CC0 = 10, the internal bit rate clock is provided at P_{22} regardless of the values for TE or RE.

(Note) The source of SCI internal bit rate clock is the timer's free running counter. An MPU write to the counter can disturb serial operations.

Transmit/Receive Control and Status Register (TRCSR) ($11)

The Transmit/Receive Control and Status Register controls the transmitter, receiver, wake-up feature, and two individual interrupts and monitors the status of serial operations. All eight bits are readable while bits 0 to 4 are also writable. The register is initialized to $20 by $\overline{RES}$.

Transmit/Receive Control and Status Register (TRCSR)

7	6	5	4	3	2	1	0	
RDRF	ORFE	TDRE	RIE	RE	TIE	TE	WU	$0011

Bit 0 WU "Wake-up" on Idle Line. When set, WU enables the wake-up function; it is cleared by ten consecutive 1's or by $\overline{RES}$. WU will not set if the line is idle.

Bit 1 TE Transmit Enable. When set, P_{24} DDR bit is set, cannot be changed, and will remain set if TE is subsequently cleared. When TE is changed from clear to set, the transmitter is connected to P_{24} and a preamble of nine consecutive 1's is transmitted. TE is cleared by $\overline{RES}$.

Bit 2 TIE Transmit Interrupt Enable. When set, an $\overline{IRQ_2}$ interrupt is enabled when TDRE is set; when clear, the interrupt is inhibited. TE is cleared by $\overline{RES}$.

Bit 3 RE Receive Enable. When set, P23's DDR bit is cleared, cannot be changed, and will remain clear if RE is subsequently cleared. While RE is set, the SCI receiver is enabled. RE is cleared by $\overline{RES}$.

Bit 4 RIE Receiver Interrupt Enable. When set, an $\overline{IRQ_2}$ interrupt is enabled when RDRF and/or ORFE is set; when clear, the interrupt is inhibited. RIE is cleared by $\overline{RES}$.

Bit 5 TDRE Transmit Data Register Empty. TDRE is set when the Transmit Data Register is transferred to the output serial shift register or by $\overline{RES}$. It is cleared by reading the TRCSR (with TDRE set) and then writing to the Transmit Data Register. Additional data will be transmitted only if TDRE has been cleared.

Bit 6 ORFE Overrun Framing Error. If set, ORFE indicates either an overrun or framing error. An overrun is a new byte ready to transfer to the Receiver Data Register with RDRF still set. A receiver framing error has occurred when the byte boundaries of the bit stream are not synchronized to the bit counter. An overrun can be distinguished from a framing error by the value of RDRF: if RDRF is set, then an overrun has occurred; otherwise a framing error has been detected. Data is not transferred to the Receive Data Register in an overrun or framing error condition. ORFE is cleared by reading the TRCSR (with ORFE set) then the Receive Data Register, or by $\overline{RES}$.

Bit 7 RDRF Receive Data Register Full. RDRF is set when the input serial shift register is transferred to the Receive Data Register. It is cleared by reading the TRCSR (with RDRF set), and then the Receive Data Register, or by $\overline{RES}$.

Table 6 SCI Bit Times and Rates

		$4f_0$	2.4576 MHz	4.0 MHz
SS1	SS0	E	614.4 kHz	1.0 MHz
0	0	E ÷ 16	26 μs/38,400 Baud	16 μs/62,500 Baud
0	1	E ÷ 128	208μs/4,800 Baud	128μs/7812.5 Baud
1	0	E ÷ 1024	1.67ms/600 Baud	1.024ms/976.6 Baud
1	1	E ÷ 4096	6.67ms/150 Baud	4.096ms/244.1 Baud

Table 7 SCI Format and Clock Source Control

CC1, CC0	Format	Clock Source	Port 2 Bit 2	Port 2 Bit 3	Port 2 Bit 4
00	–	–	–	**	**
01	NRZ	Internal	Not Used	**	**
10	NRZ	Internal	Output*	**	**
11	NRZ	External	Input	**	**

* Clock output is available regardless of values for bits RE and TE.

** Bit 3 is used for serial input if RE = "1" in TRCS; bit 4 is used for serial output if TE = "1" in TRCS.

- **Internally Generated Clcok**

If the user wishes for the serial I/O to furnish a clock, the following requirements are applicable:

- the values of RE and TE are immaterial.
- CC1, CC0 must be set to 10
- the maximum clock rate will be E ÷ 16.
- the clock will be at 1X the bit rate and will have a rising edge at mid-bit.

- **Externally Generated Clock**

If the user wishes to provide an external clock for the serial I/O, the following requirements are applicable:

- the CC1, CC0, field in the Rate and Mode Control Register must be set to 11,
- the external clock must be set to 8 times (X8) the desired baud rate and
- the maximum external clock frequency is 1.0 MHz.

- **Serial Operations**

The SCI is initialized by writing control bytes first to the Rate and Mode Control Register and then to the Transmit/Receive Control and Status Register.

The Transmitter Enable (TE) and Receiver Enable (RE) bits may be left set for dedicated operations.

Transmit operations

The transmit operation is enabled by TE in the Transmit/Receive Control and Status Register. When TE is set, the output of the transmit serial shift register is connected to P_{24} and the serial output by first transmitting to a ten-bit preamble of 1's. Following the preamble, internal synchronization is established and the transmitter section is ready for operation.

At this point one of two situation exist:

1) if the Transmit Data Register is empty (TDRE = 1), a continuous string of ones will be sent indicating an idle line, or,
2) if a byte has been written to the Transmit Data Register (TDRE = 0), it is transferred to the output serial shift register and transmission will begin.

During the transfer itself, the start bit (0) is first transmitted. Then the 8 data bits (beginning with bit 0) followed by the stop bit (1), are transmitted. When the Transmitter Data Register has been emptied, the TDRE flag bit is set.

If the MCU fails to respond to the flag within the proper time, (TDRE is still set when the next normal transfer from the parallel data register to the serial output register should occur) then a 1 will be sent (instead of a 0) at "Start" bit time, followed by more 1's until more data is supplied to the data register. No 0's will be sent while TDRE remains a 1.

Receive Operations

The receive operation is enabled by RE which configures P_{23}. The receive operation is controled by the contents of the Transmit/Receive Control and Status Register and the Rate and Mode Control Register.

The receiver bit interval is divided into 8 sub-intervals for internal synchronization. In the NRZ Mode, the received bit stream is synchronized by the first 0 (space) encountered.

The approximate center of each bit time is strobed during the next 10 bits. If the tenth bit is not a 1 (stop bit) a framing error is assumed, and ORFE is set. If the tenth bit is a 1, the data is transferred to the Receive Data Register, and interrupt flag RDRF is set. If RDRF is still set at the next tenth bit time, ORFE will be set, indicating an over-run has occurred. When the MCU responds to either flag (RDRF or ORFE) by reading the status register followed by reading the Data Register, RDRF (or ORFE) will be cleared.

■ INSTRUCTION SET

The HD68P01 is upward source and object code compatible with the HD6800. Execution times of key instructions have been reduced and several new instructions have been added, including hardware multiply. A list of new operations added to the HD6800 instruction set is shown in Table 8.

In addition, two new special opcodes, 4E and 5E, are provided for test purposes. These opcodes force the Program Counter to increment like a 16-bit counter, causing address lines used in the expanded modes to increment until the device is reset. These opcodes have no mnemonics.

Table 8 New Instructions

Instruction	Description
ABX	Unsigned addition of Accumulator B to Index Register
ADDD	Adds (without carry) the double accumulator to memory and leaves the sum in the double accumulator
ASLD	Shifts the double accumulator left (towards MSB) one bit; the LSB is cleared and the MSB is shifted into the C-bit
BRN	Branch Never
LDD	Loads double accumulator from memory
LSRD	Shifts the double accumulator right (towards LSB) one bit; the MSB is cleared and the LSB is shifted into the C-bit
MUL	Unsigned multiply; multiplies the two accumulators and leaves the product in the double accumulator
PSHX	Pushes the Index Register to stack
PULX	Pulls the Index Register from stack
STD	Stores the double accumulator to memory
SUBD	Subtracts memory from the double accumulator and leaves the difference in the double accumulator

• Programming Model

A programming model for the HD68P01 is shown in Figure 10. Accumulator A can be concatenated with accumulator B and jointly referred to as accumulator D where A is the most significant byte. Any operation which modifies the double accumulator will also modify accumulator A and/or B. Other registers are defined as follows:

Program Counter

The program counter is a 16-bit register which always points to the next instruction.

Stack Pointer

The stack pointer is a 16-bit register which contains the address of the next available location in a pushdown/pullup (LIFO) queue. The stack resides in random access memory at a location defined by the programmer.

Index Register

The Index Register is a 16-bit register which can be used to store data or provide an address for the indexed mode of addressing.

Accumulators

The MCU contains two 8-bit accumulators, A and B, which are used to store operands and results from the arithmetic logic unit (ALU). They can also be concatenated and referred to as the D (double) accumulator.

Condition Code Registers

The condition code register indicates the results of an instruction and includes the following five condition bits: Negative (N), Zero (Z), Overflow (V), Carry/Borrow from MSB (C), and Half Carry from bit 3 (H). These bits are testable by the conditional branch instruction. Bit 4 is the interrupt mask (I-bit) and inhibits all maskable interrupts when set. The two unused bits, b6 and b7 are read as ones.

• Addressing Modes

The MCU provides six addressing modes which can be used to reference memory. A summary of addressing modes for all instructions is presented in Table 9, 10, 11, and 12 where execution times are provided in E-cycles. Instruction execution times are summarized in Table 13. With an input frequency of 4 MHz, E-cycles are equivalent to microseconds. A cycle-by-cycle description of bus activity for each instruction is provided in Table 14 and a description of selected instructions is shown in Figure 26.

Immediate Addressing

The operand or "immediate byte(s)" is contained in the following byte(s) of the instruction where the number of bytes matches the size of the register. These are two or three byte instructions.

Direct Addressing

The least significant byte of the operand address is contained in the second byte of the instruction and the most significant byte is assumed to be $00. Direct addressing allows the user to access $00 through $FF using two byte instructions and execution time is reduced by eliminating the additional memory access. In most applications, the 256-byte area is reserved for frequently referenced data.

Extended Addressing

The second and third bytes of the instruction contain the absolute address of the operand. These are three byte instructions.

Indexed Addressing

The unsigned offset contained in the second byte of the instruction is added with carry to the Index Register and used to reference memory without changing the Index Register. These are two byte instructions.

Table 9 Index Register and Stack Manipulation Instructions

Pointer Operations	Mnemonic	Immed			Direct			Index			Extend			Implied			Boolean/ Arithmetic Operation	Cond. Code Reg.					
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#		5	4	3	2	1	0
																		H	I	N	Z	V	C
Compare Index Reg	CPX	8C	4	3	9C	5	2	AC	6	2	BC	6	3				X − M: M + 1	•	•	↕	↕	↕	↕
Decrement Index Reg	DEX													09	3	1	X − 1 → X	•	•	•	↕	•	•
Decrement Stack Pntr	DES													34	3	1	SP − 1 → SP	•	•	•	•	•	•
Increment Index Reg	INX													08	3	1	X + 1 → X	•	•	•	↕	•	•
Increment Stack Pntr	INS													31	3	1	SP + 1 → SP	•	•	•	•	•	•
Load Index Reg	LDX	CE	3	3	DE	4	2	EE	5	2	FE	5	3				M → X_H, (M+1) → X_L	•	•	↕	↕	R	•
Load Stack Pntr	LDS	8E	3	3	9E	4	2	AE	5	2	BE	5	3				M → SP_H, (M+1) → SP_L	•	•	↕	↕	R	•
Store Index Reg	STX				DF	4	2	EF	5	2	FF	5	3				X_H → M, X_L → (M + 1)	•	•	↕	↕	R	•
Store Stack Pntr	STS				9F	4	2	AF	5	2	BF	5	3				SP_H → M, SP_L → (M + 1)	•	•	↕	↕	R	•
Index Reg → Stack Pntr	TXS													35	3	1	X − 1 → SP	•	•	•	•	•	•
Stack Pntr → Index Reg	TSX													30	3	1	SP + 1 → X	•	•	•	•	•	•
Add	ABX													3A	3	1	B + X → X	•	•	•	•	•	•
Push Data	PSHX													3C	4	1	X_L → M_{SP}, SP − 1 → SP X_H → M_{SP}, SP − 1 → SP	•	•	•	•	•	•
Pull Data	PULX													38	5	1	SP + 1 → SP, M_{SP} → X_H SP + 1 → SP, M_{SP} → X_L	•	•	•	•	•	•

The Condition Code Register notes are listed after Table 12.

Implied Addressing

The operand(s) are registers and no memory reference is required. These are single byte instructions.

Relative Addressing

Relative addressing is used only for branch instructions. If the branch condition is true, the Program Counter is overwritten with the sum of a signed single byte displacement in the second byte of the instruction and the current Program Counter. This provides a branch range of −126 to 129 bytes from the first byte of the instruction. These are two byte instructions.

Table 10 Accumulator and Memory Instructions

Accumulator and Memory Operations	Mnemonic	Immed			Direct			Index			Extend			Implied			Boolean Expression	Cond. Code Reg.					
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#		H	I	N	Z	V	C
Add Acmltrs	ABA													1B	2	1	A + B → A	↕	•	↕	↕	↕	↕
Add B to X	ABX													3A	3	1	B + X → X	•	•	•	•	•	•
Add with Carry	ADCA	89	2	2	99	3	2	A9	4	2	B9	4	3				A + M + C → A	↕	•	↕	↕	↕	↕
	ADCB	C9	2	2	D9	3	2	E9	4	2	F9	4	3				B + M + C → B	↕	•	↕	↕	↕	↕
Add	ADDA	8B	2	2	9B	3	2	AB	4	2	BB	4	3				A + M → A	↕	•	↕	↕	↕	↕
	ADDB	CB	2	2	DB	3	2	EB	4	2	FB	4	3				B + M → A	↕	•	↕	↕	↕	↕
Add Double	ADDD	C3	4	3	D3	5	2	E3	6	2	F3	6	3				D + M:M + 1 → D	•	•	↕	↕	↕	↕
And	ANDA	84	2	2	94	3	2	A4	4	2	B4	4	3				A · M → A	•	•	↕	↕	R	•
	ANDB	C4	2	2	D4	3	2	E4	4	2	F4	4	3				B · M → B	•	•	↕	↕	R	•
Shift Left, Arithmetic	ASL							68	6	2	78	6	3					•	•	↕	↕	↕	↕
	ASLA													48	2	1		•	•	↕	↕	↕	↕
	ASLB													58	2	1		•	•	↕	↕	↕	↕
Shift Left Dbl	ASLD													05	3	1		•	•	↕	↕	↕	↕
Shift Right, Arithmetic	ASR							67	6	2	77	6	3					•	•	↕	↕	↕	↕
	ASRA													47	2	1		•	•	↕	↕	↕	↕
	ASRB													57	2	1		•	•	↕	↕	↕	↕
Bit Test	BITA	85	2	2	95	3	2	A5	4	2	B5	4	3				A · M	•	•	↕	↕	R	•
	BITB	C5	2	2	D5	3	2	E5	4	2	F5	4	3				B · M	•	•	↕	↕	R	•
Compare Acmltrs	CBA													11	2	1	A - B	•	•	↕	↕	↕	↕
Clear	CLR							6F	6	2	7F	6	3				00 → M	•	•	R	S	R	R
	CLRA													4F	2	1	00 → A	•	•	R	S	R	R
	CLRB													5F	2	1	00 → B	•	•	R	S	R	R
Compare	CMPA	81	2	2	91	3	2	A1	4	2	B1	4	3				A - M	•	•	↕	↕	↕	↕
	CMPB	C1	2	2	D1	3	2	E1	4	2	F1	4	3				B - M	•	•	↕	↕	↕	↕
1's Complement	COM							63	6	2	73	6	3				$\overline{M} \rightarrow M$	•	•	↕	↕	R	S
	COMA													43	2	1	$\overline{A} \rightarrow A$	•	•	↕	↕	R	S
	COMB													53	2	1	$\overline{B} \rightarrow B$	•	•	↕	↕	R	S
Decimal Adj, A	DAA													19	2	1	Adj binary sum to BCD	•	•	↕	↕	↕	↕
Decrement	DEC							6A	6	2	7A	6	3				M - 1 → M	•	•	↕	↕	↕	•
	DECA													4A	2	1	A - 1 → A	•	•	↕	↕	↕	•
	DECB													5A	2	1	B - 1 → B	•	•	↕	↕	↕	•
Exclusive OR	EORA	88	2	2	98	3	2	A8	4	2	B8	4	3				A ⊕ M → A	•	•	↕	↕	R	•
	EORB	C8	2	2	D8	3	2	E8	4	2	F8	4	3				B ⊕ M → B	•	•	↕	↕	R	•
Increment	INC							6C	6	2	7C	6	3				M + 1 → M	•	•	↕	↕	↕	•
	INCA													4C	2	1	A + 1 → A	•	•	↕	↕	↕	•
	INCB													5C	2	1	B + 1 → B	•	•	↕	↕	↕	•
Load Acmltrs	LDAA	86	2	2	96	3	2	A6	4	2	B6	4	3				M → A	•	•	↕	↕	R	•
	LDAB	C6	2	2	D6	3	2	E6	4	2	F6	4	3				M → B	•	•	↕	↕	R	•
Load Double	LDD	CC	3	3	DC	4	2	EC	5	2	FC	5	3				M:M + 1 → D	•	•	↕	↕	R	•
Logical Shift, Left	LSL							68	6	2	78	6	3					•	•	↕	↕	↕	↕
	LSLA													48	2	1		•	•	↕	↕	↕	↕
	LSLB													58	2	1		•	•	↕	↕	↕	↕
	LSLD													05	3	1		•	•	↕	↕	↕	↕
Shift Right, Logical	LSR							64	6	2	74	6	3					•	•	R	↕	↕	↕
	LSRA													44	2	1		•	•	R	↕	↕	↕
	LSRB													54	2	1		•	•	R	↕	↕	↕
	LSRD													04	3	1		•	•	R	↕	↕	↕

(Continued)

Table 10 Accumulator and Memory Instructions (Continued)

Accumulator and Memory Operations	Mnemonic	Immed			Direct			Index			Extend			Implied			Boolean Expression	Cond. Code Reg.					
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#		H	I	N	Z	V	C
Multiply	MUL													3D	10	1	A X B → D	•	•	•	•	•	‡
2's Complement (Negate)	NEG							60	6	2	70	6	3				00 - M → M	•	•	‡	‡	‡	‡
	NEGA													40	2	1	00 - A → A	•	•	‡	‡	‡	‡
	NEGB													50	2	1	00 - B → B	•	•	‡	‡	‡	‡
No Operation	NOP													01	2	1	PC + 1 → PC	•	•	•	•	•	•
Inclusive OR	ORAA	8A	2	2	9A	3	2	AA	4	2	BA	4	3				A + M → A	•	•	‡	‡	R	•
	ORAB	CA	2	2	DA	3	2	EA	4	2	FA	4	3				B + M → B	•	•	‡	‡	R	•
Push Data	PSHA													36	3	1	A → Stack	•	•	•	•	•	•
	PSHB													37	3	1	B → Stack	•	•	•	•	•	•
Pull Data	PULA													32	4	1	Stack → A	•	•	•	•	•	•
	PULB													33	4	1	Stack → B	•	•	•	•	•	•
Rotate Left	ROL							69	6	2	79	6	3					•	•	‡	‡	‡	‡
	ROLA													49	2	1		•	•	‡	‡	‡	‡
	ROLB													59	2	1		•	•	‡	‡	‡	‡
Rotate Right	ROR							66	6	2	76	6	3					•	•	‡	‡	‡	‡
	RORA													46	2	1		•	•	‡	‡	‡	‡
	RORB													56	2	1		•	•	‡	‡	‡	‡
Subtract Acmltr	SBA													10	2	1	A - B → A	•	•	‡	‡	‡	‡
Subtract with Carry	SBCA	82	2	2	92	3	2	A2	4	2	B2	4	3				A - M - C → A	•	•	‡	‡	‡	‡
	SBCB	C2	2	2	D2	3	2	E2	4	2	F2	4	3				B - M - C → B	•	•	‡	‡	‡	‡
Store Acmltrs	STAA				97	3	2	A7	4	2	B7	4	3				A → M	•	•	‡	‡	R	•
	STAB				D7	3	2	E7	4	2	F7	4	3				B → M	•	•	‡	‡	R	•
	STD				DD	4	2	ED	5	2	FD	5	3				D → M:M + 1	•	•	‡	‡	R	•
Subtract	SUBA	80	2	2	90	3	2	A0	4	2	B0	4	3				A - M → A	•	•	‡	‡	‡	‡
	SUBB	C0	2	2	D0	3	2	E0	4	2	F0	4	3				B - M → B	•	•	‡	‡	‡	‡
Subtract Double	SUBD	83	4	3	93	5	2	A3	6	2	B3	6	3				D - M:M + 1 → D	•	•	‡	‡	‡	‡
Transfer Acmltr	TAB													16	2	1	A → B	•	•	‡	‡	R	•
	TBA													17	2	1	B → A	•	•	‡	‡	R	•
Test, Zero or Minus	TST							6D	6	2	7D	6	3				M - 00	•	•	‡	‡	R	R
	TSTA													4D	2	1	A - 00	•	•	‡	‡	R	R
	TSTB													5D	2	1	B - 00	•	•	‡	‡	R	R

The Condition Code Register notes are listed after Table 12.

Table 11 Jump and Branch Instructions

Operations	Mnemonic	Direct			Relative			Index			Extend			Implied			Branch Test	Cond. Code Reg.					
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#		5	4	3	2	1	0
																		H	I	N	Z	V	C
Branch Always	BRA				20	3	2										None	•	•	•	•	•	•
Branch Never	BRN				21	3	2										None	•	•	•	•	•	•
Branch If Carry Clear	BCC				24	3	2										C = 0	•	•	•	•	•	•
Branch If Carry Set	BCS				25	3	2										C = 1	•	•	•	•	•	•
Branch If = Zero	BEQ				27	3	2										Z = 1	•	•	•	•	•	•
Branch If ≥ Zero	BGE				2C	3	2										N ⊕ V = 0	•	•	•	•	•	•
Branch If > Zero	BGT				2E	3	2										Z + (N ⊕ V) = 0	•	•	•	•	•	•
Branch If Higher	BHI				22	3	2										C + Z = 0	•	•	•	•	•	•
Branch If Higher or Same	BHS				24	3	2										C = 0	•	•	•	•	•	•
Branch If ≤ Zero	BLE				2F	3	2										Z + (N ⊕ V) = 1	•	•	•	•	•	•
Branch If Carry Set	BLO				25	3	2										C = 1	•	•	•	•	•	•
Branch If Lower Or Same	BLS				23	3	2										C + Z = 1	•	•	•	•	•	•
Branch If < Zero	BLT				2D	3	2										N ⊕ V = 1	•	•	•	•	•	•
Branch If Minus	BMI				2B	3	2										N = 1	•	•	•	•	•	•
Branch If Not Equal Zero	BNE				26	3	2										N = 0	•	•	•	•	•	•
Branch If Overflow Clear	BVC				28	3	2										V = 0	•	•	•	•	•	•
Branch If Overflow Set	BVS				29	3	2										V = 1	•	•	•	•	•	•
Branch If Plus	BPL				2A	3	2										N = 0	•	•	•	•	•	•
Branch To Subroutine	BSR				8D	6	2										See Special Operations Figure 26	•	•	•	•	•	•
Jump	JMP							6E	3	2	7E	3	3					•	•	•	•	•	•
Jump To Subroutine	JSR	9D	5	2				AD	6	2	BD	6	3					•	•	•	•	•	•
No Operation	NOP													01	2	1		•	•	•	•	•	•
Return From Interrupt	RTI													3B	10	1	See Special Operations Figure 26	‡	‡	‡	‡	‡	‡
Return From Subroutine	RTS													39	5	1		•	•	•	•	•	•
Software Interrupt	SWI													3F	12	1		•	S	•	•	•	•
Wait For Interrupt	WAI													3E	9	1		•	•	•	•	•	•

The Condition Code Register notes are listed after Table 12.

Table 12 Condition Code Register Manipulation Instructions

Operations	Implied				Boolean Operation	Cond. Code Reg.					
	Mnemonic	OP	~	#		5	4	3	2	1	0
						H	I	N	Z	V	C
Clear Carry	CLC	0C	2	1	0 → C	•	•	•	•	•	R
Clear Interrupt Mask	CLI	0E	2	1	0 → I	•	R	•	•	•	•
Clear Overflow	CLV	0A	2	1	0 → V	•	•	•	•	R	•
Set Carry	SEC	0D	2	1	1 → C	•	•	•	•	•	S
Set Interrupt Mask	SEI	0F	2	1	1 → I	•	S	•	•	•	•
Set Overflow	SEV	0B	2	1	1 → V	•	•	•	•	S	•
Accumulator A → CCR	TAP	06	2	1	A → CCR	‡	‡	‡	‡	‡	‡
CCR → Accumulator A	TPA	07	2	1	CCR → A	•	•	•	•	•	•

LEGEND

OP Operation Code (Hexadecimal)
~ Number of MPU Cycles
M_{SP} Contents of memory location pointed to by Stack Pointer
Number of Program Bytes
+ Arithmetic Plus
- Arithmetic Minus
• Boolean AND
X Arithmetic Multiply
+ Boolean Inclusive OR
⊕ Boolean Exclusive OR
$\overline{M}$ Complement of M
→ Transfer Into
0 Bit = Zero
00 Byte = Zero

CONDITION CODE SYMBOLS

H Half-carry from bit 3
I Interrupt mask
N Negative (sign bit)
Z Zero (byte)
V Overflow, 2's complement
C Carry/Borrow from MSB
R Reset Always
S Set Always
‡ Affected
• Not Affected

Table 13 Instruction Execution Times in E-Cycles

	Addressing Mode					
	Immediate	Direct	Extended	Indexed	Implied	Relative
ABA	•	•	•	•	2	•
ABX	•	•	•	•	3	•
ADC	2	3	4	4	•	•
ADD	2	3	4	4	•	•
ADDD	4	5	6	6	•	•
AND	2	3	4	4	•	•
ASL	•	•	6	6	2	•
ASLD	•	•	•	•	3	•
ASR	•	•	6	6	2	•
BCC	•	•	•	•	•	3
BCS	•	•	•	•	•	3
BEQ	•	•	•	•	•	3
BGE	•	•	•	•	•	3
BGT	•	•	•	•	•	3
BHI	•	•	•	•	•	3
BHS	•	•	•	•	•	3
BIT	2	3	4	4	•	•
BLE	•	•	•	•	•	3
BLO	•	•	•	•	•	3
BLS	•	•	•	•	•	3
BLT	•	•	•	•	•	3
BMI	•	•	•	•	•	3
BNE	•	•	•	•	•	3
BPL	•	•	•	•	•	3
BRA	•	•	•	•	•	3
BRN	•	•	•	•	•	3
BSR	•	•	•	•	•	6
BVC	•	•	•	•	•	3
BVS	•	•	•	•	•	3
CBA	•	•	•	•	2	•
CLC	•	•	•	•	2	•
CLI	•	•	•	•	2	•
CLR	•	•	6	6	2	•
CLV	•	•	•	•	2	•
CMP	2	3	4	4	•	•
COM	•	•	6	6	2	•
CPX	4	5	6	6	•	•
DAA	•	•	•	•	2	•
DEC	•	•	6	6	2	•
DES	•	•	•	•	3	•
DEX	•	•	•	•	3	•
EOR	2	3	4	4	•	•
INC	•	•	6	6	•	•
INS	•	•	•	•	3	•

	Addressing Mode					
	Immediate	Direct	Extended	Indexed	Implied	Relative
INX	•	•	•	•	3	•
JMP	•	•	3	3	•	•
JSR	•	5	6	6	•	•
LDA	2	3	4	4	•	•
LDD	3	4	5	5	•	•
LDS	3	4	5	5	•	•
LDX	3	4	5	5	•	•
LSL	•	•	6	6	2	•
LSLD	•	•	•	•	3	•
LSR	•	•	6	6	2	•
LSRD	•	•	•	•	3	•
MUL	•	•	•	•	10	•
NEG	•	•	6	6	2	•
NOP	•	•	•	•	2	•
ORA	2	3	4	4	•	•
PSH	•	•	•	•	3	•
PSHX	•	•	•	•	4	•
PUL	•	•	•	•	4	•
PULX	•	•	•	•	5	•
ROL	•	•	6	6	2	•
ROR	•	•	6	6	2	•
RTI	•	•	•	•	10	•
RTS	•	•	•	•	5	•
SBA	•	•	•	•	2	•
SBC	2	3	4	4	•	•
SEC	•	•	•	•	2	•
SEI	•	•	•	•	2	•
SEV	•	•	•	•	2	•
STA	•	3	4	4	•	•
STD	•	4	5	5	•	•
STS	•	4	5	5	•	•
STX	•	4	5	5	•	•
SUB	2	3	4	4	•	•
SUBD	4	5	6	6	•	•
SWI	•	•	•	•	12	•
TAB	•	•	•	•	2	•
TAP	•	•	•	•	2	•
TBA	•	•	•	•	2	•
TPA	•	•	•	•	2	•
TST	•	•	6	6	2	•
TSX	•	•	•	•	3	•
TXS	•	•	•	•	3	•
WAI	•	•	•	•	9	•

■ SUMMARY OF CYCLE BY CYCLE OPERATION

Table 14 provides a detailed description of the information present on the Address Bus, Data Bus, and the Read/Write (R/$\overline{W}$) line during each cycle of each instruction.

The information is useful in comparing actual with expected results during debug to both software and hardware as the program is executed. The information is categorized in groups according to addressing mode and number of cycles per instruction. In general, instructions with the same addressing mode and number of cycles execute in the same manner. Exceptions are indicated in the table.

Note that during MPU reads of internal locations, the resultant value will not appear on the external Data Bus except in Mode 0. "High order" byte refers to the most significant byte of a 16-bit value.

Table 14 Cycle by Cycle Operation

Address Mode & Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
IMMEDIATE					
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	2	1 2	Op Code Address Op Code Address + 1	1 1	Op Code Operand Data
LDS LDX LDD	3	1 2 3	Op Code Address Op Code Address + 1 Op Code Address + 2	1 1 1	Op Code Operand Data (High Order Byte) Operand Data (Low Order Byte)
CPX SUBD ADDD	4	1 2 3 4	Op Code Address Op Code Address + 1 Op Code Address + 2 Address Bus FFFF	1 1 1 1	Op Code Operand Data (High Order Byte) Operand Data (Low Order Byte) Low Byte of Restart Vector
DIRECT					
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	3	1 2 3	Op Code Address Op Code Address + 1 Address of Operand	1 1 1	Op Code Address of Operand Operand Data
STA	3	1 2 3	Op Code Address Op Code Address + 1 Destination Address	1 1 0	Op Code Destination Address Data from Accumulator
LDS LDX LDD	4	1 2 3 4	Op Code Address Op Code Address + 1 Address of Operand Operand Address + 1	1 1 1 1	Op Code Address of Operand Operand Data (High Order Byte) Operand Data (Low Order Byte)
STS STX STD	4	1 2 3 4	Op Code Address Op Code Address + 1 Address of Operand Address of Operand + 1	1 1 0 0	Op Code Address of Operand Register Data (High Order Byte) Register Data (Low Order Byte)
CPX SUBD ADDD	5	1 2 3 4 5	Op Code Address Op Code Address + 1 Operand Address Operand Address + 1 Address Bus FFFF	1 1 1 1 1	Op Code Address of Operand Operand Data (High Order Byte) Operand Data (Low Order Byte) Low Byte of Restart Vector
JSR	5	1 2 3 4 5	Op Code Address Op Code Address + 1 Subroutine Address Stack Pointer Stack Pointer + 1	1 1 1 0 0	Op Code Irrelevant Data First Subroutine Op Code Return Address (Low Order Byte) Return Address (High Order Byte)

(Continued)

Table 14 Cycle by Cycle Operation (Continued)

Address Mode & Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
EXTENDED					
JMP	3	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Jump Address (High Order Byte)
		3	Op Code Address + 2	1	Jump Address (Low Order Byte)
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	4	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Address of Operand
		3	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	Address of Operand	1	Operand Data
STA	4	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Destination Address (High Order Byte)
		3	Op Code Address + 2	1	Destination Address (Low Order Byte)
		4	Operand Destination Address	0	Data from Accumulator
LDS LDX LDD	5	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Address of Operand (High Order Byte)
		3	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	Address of Operand	1	Operand Data (High Order Byte)
		5	Address of Operand + 1	1	Operand Data (Low Order Byte)
STS STX STD	5	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Address of Operand (High Order Byte)
		3	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	Address of Operand	0	Operand Data (High Order Byte)
		5	Address of Operand + 1	0	Operand Data (Low Order Byte)
ASL LSR ASR NEG CLR ROL COM ROR DEC TST* INC	6	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Address of Operand (High Order Byte)
		3	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	Address of Operand	1	Current Operand Data
		5	Address Bus FFFF	1	Low Byte of Restart Vector
		6	Address of Operand	0	New Operand Data
CPX SUBD ADDD	6	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Operand Address (High Order Byte)
		3	Op Code Address + 2	1	Operand Address (Low Order Byte)
		4	Operand Address	1	Operand Data (High Order Byte)
		5	Operand Address + 1	1	Operand Data (Low Order Byte)
		6	Address Bus FFFF	1	Low Byte of Restart Vector
JSR	6	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Address of Subroutine (High Order Byte)
		3	Op Code Address + 2	1	Address of Subroutine (Low Order Byte)
		4	Subroutine Starting Address	1	Op Code of Next Instruction
		5	Stack Pointer	0	Return Address (Low Order Byte)
		6	Stack Pointer – 1	0	Return Address (High Order Byte)

* In the TST instruction, the line condition of the sixth cycle does the following: R/$\overline{W}$ = "High", AB = FFFF, DB = Low Byte of Reset Vector.

(Continued)

Table 14 Cycle by Cycle Operation (Continued)

Address Mode & Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
INDEXED					
JMP	3	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
ADC EOR	4	1	Op Code Address	1	Op Code
ADD LDA		2	Op Code Address + 1	1	Offset
AND ORA		3	Address Bus FFFF	1	Low Byte of Restart Vector
BIT SBC		4	Index Register Plus Offset	1	Operand Data
CMP SUB					
STA	4	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	0	Operand Data
LDS	5	1	Op Code Address	1	Op Code
LDX		2	Op Code Address + 1	1	Offset
LDD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	1	Operand Data (High Order Byte)
		5	Index Register Plus Offset + 1	1	Operand Data (Low Order Byte)
STS	5	1	Op Code Address	1	Op Code
STX		2	Op Code Address + 1	1	Offset
STD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	0	Operand Data (High Order Byte)
		5	Index Register Plus Offset + 1	0	Operand Data (Low Order Byte)
ASL LSR	6	1	Op Code Address	1	Op Code
ASR NEG		2	Op Code Address + 1	1	Offset
CLR ROL		3	Address Bus FFFF	1	Low Byte of Restart Vector
COM ROR		4	Index Register Plus Offset	1	Current Operand Data
DEC TST *		5	Address Bus FFFF	1	Low Byte of Restart Vector
INC		6	Index Register Plus Offset	0	New Operand Data
CPX	6	1	Op Code Address	1	Op Code
SUBD		2	Op Code Address + 1	1	Offset
ADDD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register + Offset	1	Operand Data (High Order Byte)
		5	Index Register + Offset + 1	1	Operand Data (Low Order Byte)
		6	Address Bus FFFF	1	Low Byte of Restart Vector
JSR	6	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register + Offset	1	First Subroutine Op Code
		5	Stack Pointer	0	Return Address (Low Order Byte)
		6	Stack Pointer – 1	0	Return Address (High Order Byte)

* In the TST instruction, the line condition of the sixth cycle does the following: R/$\overline{W}$ = "High", AB = FFFF, DB = Low Byte of Reset Vector.

(Continued)

Table 14 Cycle by Cycle Operation (Continued)

Address Mode & Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
IMPLIED					
ABA DAA SEC ASL DEC SEI ASR INC SEV CBA LSR TAB CLC NEG TAP CLI NOP TBA CLR ROL TPA CLV ROR TST COM SBA	2	1 2	Op Code Address Op Code Address + 1	1 1	Op Code Op Code of Next Instruction
ABX	3	1 2 3	Op Code Address Op Code Address + 1 Address Bus FFFF	1 1 1	Op Code Irrelevant Data Low Byte of Restart Vector
ASLD LSRD	3	1 2 3	Op Code Address Op Code Address + 1 Address Bus FFFF	1 1 1	Op Code Irrelevant Data Low Byte of Restart Vector
DES INS	3	1 2 3	Op Code Address Op Code Address + 1 Previous Register Contents	1 1 1	Op Code Op Code of Next Instruction Irrelevant Data
INX DEX	3	1 2 3	Op Code Address Op Code Address + 1 Address Bus FFFF	1 1 1	Op Code Op Code of Next Instruction Low Byte of Restart Vector
PSHA PSHB	3	1 2 3	Op Code Address Op Code Address + 1 Stack Pointer	1 1 0	Op Code Op Code of Next Instruction Accumulator Data
TSX	3	1 2 3	Op Code Address Op Code Address + 1 Stack Pointer	1 1 1	Op Code Op Code of Next Instruction Irrelevant Data
TXS	3	1 2 3	Op Code Address Op Code Address + 1 Address Bus FFFF	1 1 1	Op Code Op Code of Next Instruction Low Byte of Restart Vector
PULA PULB	4	1 2 3 4	Op Code Address Op Code Address + 1 Stack Pointer Stack Pointer + 1	1 1 1 1	Op Code Op Code of Next Instruction Irrelevant Data Operand Data from Stack
PSHX	4	1 2 3 4	Op Code Address Op Code Address + 1 Stack Pointer Stack Pointer + 1	1 1 0 0	Op Code Irrelevant Data Index Register (Low Order Byte) Index Register (High Order Byte)
PULX	5	1 2 3 4 5	Op Code Address Op Code Address + 1 Stack Pointer Stack Pointer + 1 Stack Pointer +2	1 1 1 1 1	Op Code Irrelevant Data Irrelevant Data Index Register (High Order Byte) Index Register (Low Order Byte)
RTS	5	1 2 3 4 5	Op Code Address Op Code Address + 1 Stack Pointer Stack Pointer + 1 Stack Pointer + 2	1 1 1 1 1	Op Code Irrelevant Data Irrelevant Data Address of Next Instruction (High Order Byte) Address of Next Instruction (Low Order Byte)

(Continued)

Table 14 Cycle by Cycle Operation (Continued)

Address Mode & Instruction	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
WAI **	9	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Op Code of Next Instruction
		3	Stack Pointer	0	Return Address (Low Order Byte)
		4	Stack Pointer – 1	0	Return Address (High Order Byte)
		5	Stack Pointer – 2	0	Index Register (Low Order Byte)
		6	Stack Pointer – 3	0	Index Register (High Order Byte)
		7	Stack Pointer – 4	0	Contents of Accumulator A
		8	Stack Pointer – 5	0	Contents of Accumulator B
		9	Stack Pointer – 6	0	Contents of Cond. Code Register
MUL	10	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Irrelevant Data
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Address Bus FFFF	1	Low Byte of Restart Vector
		5	Address Bus FFFF	1	Low Byte of Restart Vector
		6	Address Bus FFFF	1	Low Byte of Restart Vector
		7	Address Bus FFFF	1	Low Byte of Restart Vector
		8	Address Bus FFFF	1	Low Byte of Restart Vector
		9	Address Bus FFFF	1	Low Byte of Restart Vector
		10	Address Bus FFFF	1	Low Byte of Restart Vector
RTI	10	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Irrelevant Data
		3	Stack Pointer	1	Irrelevant Data
		4	Stack Pointer + 1	1	Contents of Cond. Code Reg. from Stack
		5	Stack Pointer + 2	1	Contents of Accumulator B from Stack
		6	Stack Pointer + 3	1	Contents of Accumulator A from Stack
		7	Stack Pointer + 4	1	Index Register from Stack (High Order Byte)
		8	Stack Pointer + 5	1	Index Register from Stack (Low Order Byte)
		9	Stack Pointer + 6	1	Next Instruction Address from Stack (High Order Byte)
		10	Stack Pointer + 7	1	Next Instruction Address from Stack (Low Order Byte)
SWI	12	1	Op Code Address	1	Op Code
		2	Op Code Address + 1	1	Irrelevant Data
		3	Stack Pointer	0	Return Address (Low Order Byte)
		4	Stack Pointer – 1	0	Return Address (High Order Byte)
		5	Stack Pointer – 2	0	Index Register (Low Order Byte)
		6	Stack Pointer – 3	0	Index Register (High Order Byte)
		7	Stack Pointer – 4	0	Contents of Accumulator A
		8	Stack Pointer – 5	0	Contents of Accumulator B
		9	Stack Pointer – 6	0	Contents of Cond. Code Register
		10	Stack Pointer – 7	1	Irrelevant Data
		11	Vector Address FFFA (Hex)	1	Address of Subroutine (High Order Byte)
		12	Vector Address FFFB (Hex)	1	Address of Subroutine (Low Order Byte)

** While the MCU is in the "Wait" state, its bus state will appear as a series of the MCU reads of an address which is seven locations less than the original contents of the Stack Pointer. Contrary to the HD6800, none of the ports are driven to the high impedance state by a WAI instruction.

(Continued)

Table 14 Cycle by Cycle Operation (Continued)

Address Mode & Instruction	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
RELATIVE					
BCC BHT BNE BLO BCS BLE BPL BHS BEQ BLS BRA BRN BGE BLT BVC BGT BMT BVS	3	1 2 3	Op Code Address Op Code Address + 1 Address Bus FFFF	1 1 1	Op Code Branch Offset Low Byte of Restart Vector
BSR	6	1 2 3 4 5 6	Op Code Address Op Code Address + 1 Address Bus FFFF Subroutine Starting Address Stack Pointer Stack Pointer – 1	1 1 1 1 0 0	Op Code Branch Offset Low Byte of Restart Vector Op Code of Next Instruction Return Address (Low Order Byte) Return Address (High Order Byte)

■ SUMMARY OF UNDEFINED INSTRUCTIONS OPERATION

The MCU has 36 undefined instructions. When these are carried out, the contents of Register and Memory in MPU change at random.

When the op codes (4E, 5E) are used to execute, the MPU continues to increase the program counter and it will not stop until the Reset signal enters. These op codes are used to test the LSI.

Table 15 Op Codes Map

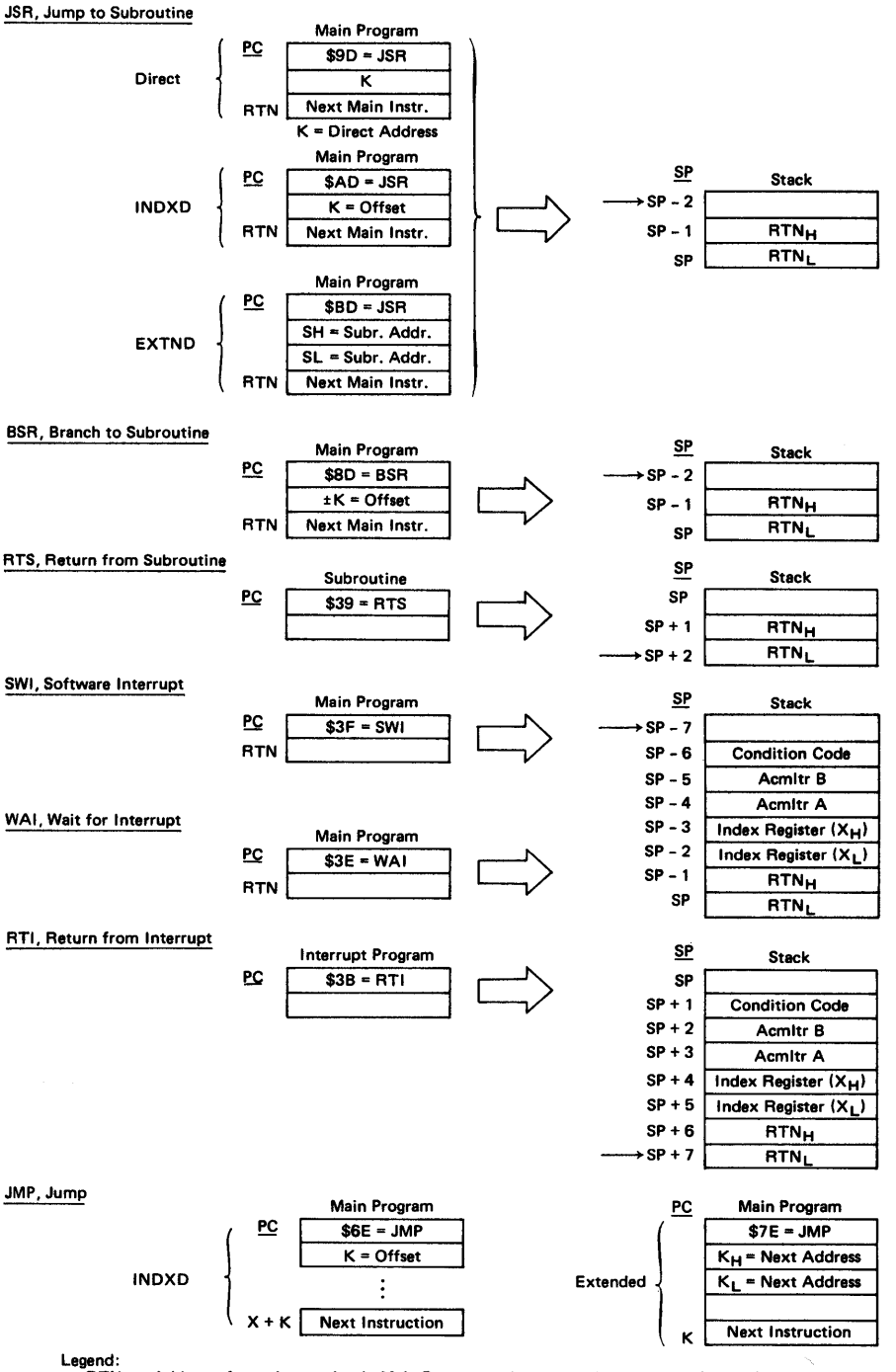
HD68P01 MICROCOMPUTER INSTRUCTIONS																		
OP CODE						ACC A	ACC B	IND	EXT	ACCA or SP				ACCB or X				
										IMM	DIR	IND	EXT	IMM	DIR	IND	EXT	
HI		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111	
LO		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0000	0		SBA	BRA	TSX	NEG				SUB								0
0001	1	NOP	CBA	BRN	INS					CMP								1
0010	2			BHI	PULA (+1)					SBC								2
0011	3			BLS	PULB (+1)	COM				*	SUBD (+2)			*	ADDD (+2)			3
0100	4	LSRD (+1)		BCC	DES	LSR				AND								4
0101	5	ASLD (+1)		BCS	TXS					BIT								5
0110	6	TAP	TAB	BNE	PSHA	ROR				LDA								6
0111	7	TPA	TBA	BEQ	PSHB	ASR					STA				STA			7
1000	8	INX (+1)		BVC	PULX (+2)	ASL				EOR								8
1001	9	DEX (+1)	DAA	BVS	RTS (+2)	ROL				ADC								9
1010	A	CLV		BPL	ABX	DEC				ORA								A
1011	B	SEV	ABA	BMI	RTI (+7)					ADD								B
1100	C	CLC		BGE	PSHX (+1)	INC				*	CPX (+2)			*	LDD (+1)			C
1101	D	SEC		BLT	MUL (+7)	TST				BSR (+4)	JSR (+2)			* (+1)	STD (+1)			D
1110	E	CLI		BGT	WAI (+6)	**		JMP (–3)		*	LDS (+1)			*	LDX (+1)			E
1111	F	SEI		BLE	SWI (+9)	CLR				* (+1)	STS (+1)			* (+1)	STX (+1)			F
BYTE/CYCLE		1/2	1/2	2/3	1/3	1/2	1/2	2/6	3/6	2/2	2/3	2/4	3/4	2/2	2/3	2/4	3/4	

(NOTES) 1. Undefined Op codes are marked with ▨.
2. ( ) indicate that the number in parenthesis must be added to the cycle count for that instruction.
3. The instructions shown below are all 3 bytes and are marked with "*".
Immediate addressing mode of SUBD, CPX, LDS, ADDD, LDD and LDX instructions, and undefined op codes (8F, CD, CF).
4. The Op codes (4E, 5E) are 1 byte/∞ cycles instructions, and are marked with "**".

■ COMPATIBLE WITH THE HD6801S AND THE HD6801V

In order to be pin compatible with the HD6801S, ROM of the HD68P01 must be located at $F800 – $FFFF. Memory addresses $E000 to $F7FF are not usable. The other addresses are available same as the HD6801S's.

In order to be pin compatible with the HD6801V ROM of the HD68P01 must be located at $F000 – $FFFF. Memory addresses $E000 to $EFFF are not usable. The other addresses are available same as the HD6801V's.



Legend:
RTN = Address of next instruction in Main Program to be executed upon return from subroutine
RTN_H = Most significant byte of Return Address
RTN_L = Least significant byte of Return Address
→ = Stack Pointer After Execution
K = 8-bit Unsigned Value

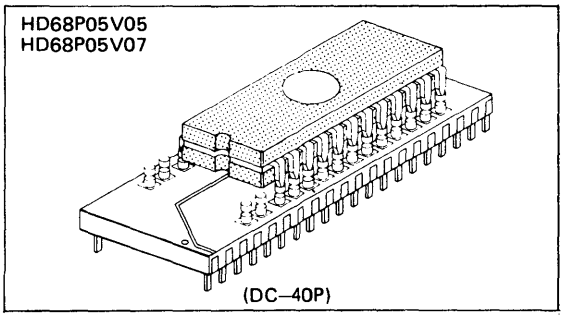
Figure 26 Special Operations

HD68P05V05, HD68P05V07

MCU (Microcomputer Unit)

—PRELIMINARY—

The HD68P05V is the 8-bit Microcomputer Unit (MCU) which contains a CPU, on-chip clock, RAM, I/O and timer. It is designed for the user who needs an economical microcomputer with the proven capabilities of the HD6800-based instruction set. Setting EPROM on the package, this MCU has the equivalent function as the HD6805U and HD6805V. HD68P05V05 uses HN462532 as EPROM. HD68P05V07 uses HN462732 as EPROM. The following are some of the hardware and software highlights of the MCU.



■ HARDWARE FEATURES

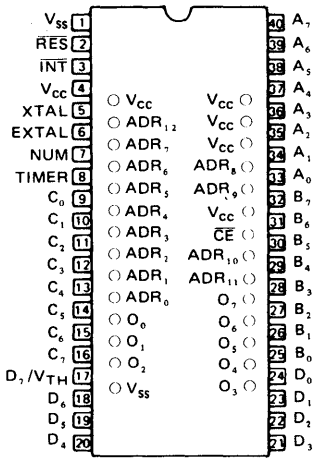
- 8-Bit Architecture
- 96 Bytes of RAM
- Memory Mapped I/O
- Internal 8-Bit Timer with 7-Bit Prescaler
- Vectored Interrupts – External, Timer and Software
- 24 I/O Ports + 8 Input Port
  (8 Lines LED Compatible; 7 Voltage Comparator Inputs)
- On-Chip Clock Circuit
- Master Reset
- Complete Development System Support by Evaluation Kit
- 5 Vdc Single Supply

■ SOFTWARE FEATURES

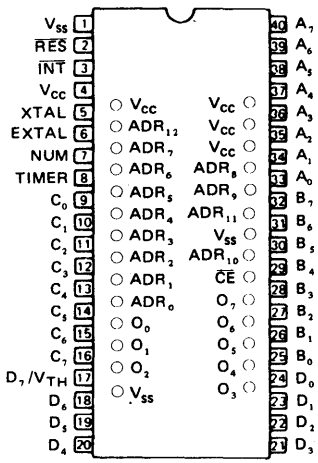
- Similar to HD6800
- Byte Efficient Instruction Set
- Easy to Program
- True Bit Manipulation
- Bit Test and Branch Instructions
- Versatile Interrupt Handing
- Powerful Indexed Addressing for Tables
- Full Set of Conditional Branches
- Memory Usable as Registers/Flags
- Single Instruction Memory Examine/Change
- 10 Powerful Addressing Modes
- All Addressing Modes Apply to ROM, RAM and I/O
- Compatible Instruction Set with HD6805

■ PIN ARRANGEMENT (Top View)

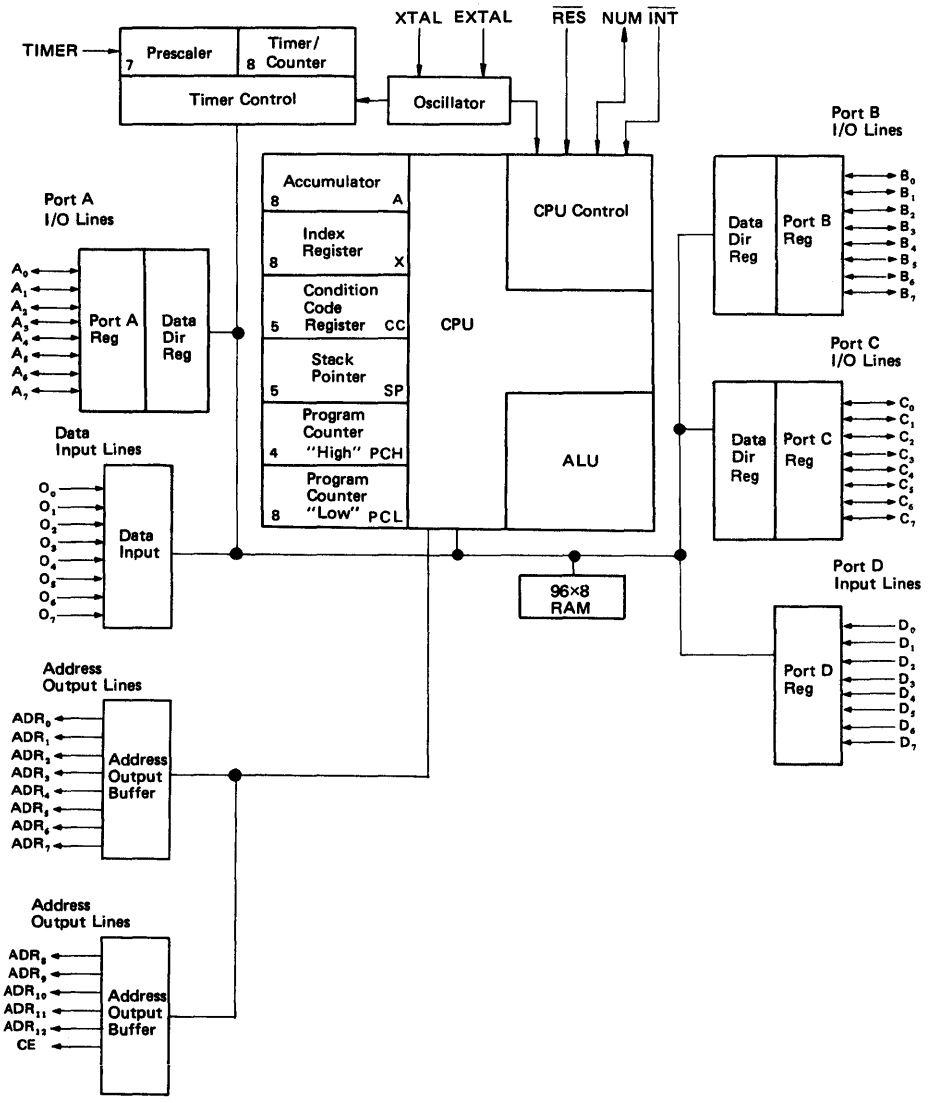
HD68P05V05



HD68P05V07



■ BLOCK DIAGRAM



■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC} *	-0.3 ~ +7.0	V
Input Voltage (EXCEPT TIMER)	V_{in} *	-0.3 ~ +7.0	V
Input Voltage (TIMER)		-0.3 ~ +12.0	V
Operating Temperature	T_{opr}	0 ~ +70	°C
Storage Temperature	T_{stg}	- 55 ~ +150	°C

* With respect to V_{SS} (SYSTEM GND)

(NOTE) Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ ELECTRICAL CHARACTERISTICS

• DC CHARACTERISTICS (V_{CC}=5.25V ± 0.5V, V_{SS}=GND, Ta=0~+70°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Input "High" Voltage	$\overline{RES}$	V_{IH}		4.0	–	V_{CC}	V
	$\overline{INT}$			3.0	–	V_{CC}	V
	All Other			2.0	–	V_{CC}	V
Input "Low" Voltage	$\overline{RES}$	V_{IL}		-0.3	–	0.8	V
	$\overline{INT}$			-0.3	–	0.8	V
	XTAL (Crystal Mode)			-0.3	–	0.6	V
	All Other			-0.3	–	0.8	V
Power Dissipation		P_D		–	400	700	mW
Low Voltage Recover		LVR		–	–	4.75	V
Input Leak Current	TIMER	I_{IL}	V_{in}=0.4V~V_{CC}	-20	–	20	μA
	$\overline{INT}$			-50	–	50	μA
	XTAL (Crystal Mode)			-1200	–	0	μA

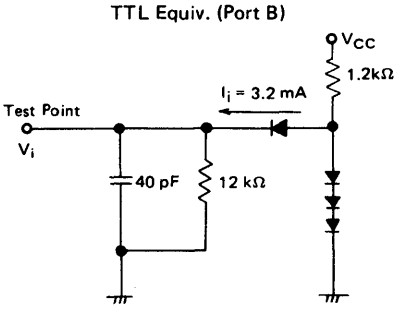
• AC CHARACTERISTICS (V_{CC}=5.25V ± 0.5V, V_{SS}=GND, Ta=0~+70°C, unless otherwise noted.)

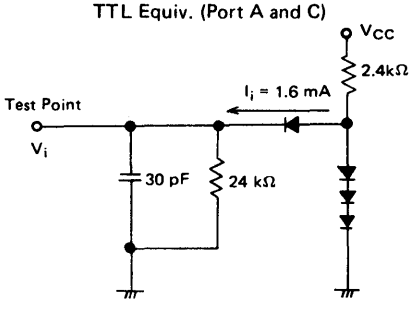
Item		Symbol	Test Condition	min	typ	max	Unit
Clock Frequency		f_{cl}		0.4	–	4.0	MHz
Cycle Time		t_{cyc}		1.0	–	10	μs
$\overline{INT}$ Pulse Width		t_{IWL}		$\frac{t_{cyc}}{250}$ +	–	–	ns
$\overline{RES}$ Pulse Width		t_{RWL}		$\frac{t_{cyc}}{250}$ +	–	–	ns
TIMER Pulse Width		t_{TWL}		$\frac{t_{cyc}}{250}$ +	–	–	ns
Oscillation Start-up Time (Crystal Mode)		t_{OSC}	C_L=22pF±20%, R_S=60Ω max.	–	–	100	ms
Delay Time Reset		t_{RHL}	External Cap. = 2.2 μF	100	–	–	ms
Input Capacitance	EXTAL	C_{in}	V_{in}=0V	–	25	35	pF
	All Other			–	6	10	pF

● PORT ELECTRICAL CHARACTERISTICS (V_{CC} = 5.25V ± 0.5V, V_{SS} = GND, Ta = 0 ~ +70°C unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Output "High" Voltage	Port A	V_{OH}	I_{OH} = −10 μA	3.5	–	–	V
			I_{OH} = −100 μA	2.4	–	–	V
	Port B		I_{OH} = −200 μA	2.4	–	–	V
			I_{OH} = −1 mA	1.5	–	–	V
	Port C		I_{OH} = −100 μA	2.4	–	–	V
Output "Low" Voltage	Port A and C	V_{OL}	I_{OL} = 1.6 mA	–	–	0.4	V
	Port B		I_{OL} = 3.2 mA	–	–	0.4	V
			I_{OL} = 10 mA	–	–	1.0	V
Input "High" Voltage	Port A, B, C, and D*	V_{IH}		2.0	–	V_{CC}	V
Input "Low" Voltage		V_{IL}		−0.3	–	0.8	V
Input Leak Current	Port A	I_{IL}	V_{in} = 0.8V	−500	–	–	μA
			V_{in} = 2V	−300	–	–	μA
	Port B, C, and D		V_{in} = 0.4V ~ V_{CC}	− 20	–	20	μA
Input "High" Voltage	Port D** (D_0 ~ D_6)	V_{IH}		–	V_{TH}+0.2	—	V
Input "Low" Voltage	Port D** (D_0 ~ D_6)	V_{IL}		–	V_{TH}-0.2	—	V
Threshold Voltage	Port D**(D_7)	V_{TH}		0	—	0.8×V_{CC}	V

* Port D as digital input
** Port D as analog input





(NOTE) 1. Load capacitance includes the floating capacitance of the probe and the jig etc.
2. All diodes are 1S2074 Ⓗ or equivalent.

Figure 1 Bus Timing Test Loads

■ SIGNAL DESCRIPTION

The input and output signals for the MCU shown in PIN ARRANGEMENT are described in the following paragraphs.

● V_{CC} and V_{SS}

Power is supplied to the MCU using these two pins. V_{CC} is +5.25V ±0.5V. V_{SS} is the ground connection.

● $\overline{INT}$

This pin provides the capability for applying an external interrupt to the MCU Refer to INTERRUPTS for additional information.

● XTAL and EXTAL

These pins provide control input for the on-chip clock circuit. A crystal (AT cut, 4 MHz maximum) can be connected to these pins to provide the internal oscillator with varying degrees of stability. Refer to INTERNAL OSCILLATOR for recommendations about these inputs.

● TIMER

This pin allows an external input to be used to decrement the internal timer circuitry. Refer to TIMER for additional information about the timer circuitry.

● $\overline{RES}$

This pin allows resetting of the MCU at times other than the automatic resetting capability already in the MCU. Refer to RESETS for additional information.

● NUM

This pin is not for user application and should be connected to ground.

- **Input/Output Lines ($A_0 \sim A_7$, $B_0 \sim B_7$, $C_0 \sim C_7$)**

These 24 lines are arranged into three 8-bit ports (A, B and C). All lines are programmable as either inputs or outputs under software control of the data direction registers. Refer to INPUTS/OUTPUTS for additional information.

- **Input Lines ($D_0 \sim D_7$)**

These are 8-bit input lines, which has two functions. Firstly, these become TTL compatible inputs, by reading $003 address. The other function of them is 7 Voltage comparators, by reading $007 address. Please refer to INPUT PORT for more detail.

■ REGISTERS

The MCU has five registers available to the programmer. They are shown in Figure 2 and are explained in the following paragraphs.

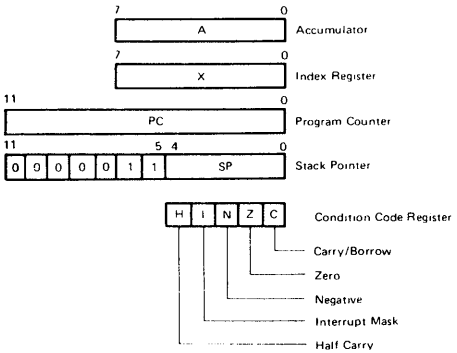


Figure 2 Programming Model

- **Accumulator (A)**

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.

- **Index Register (X)**

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit address that may be added to an offset value to create an effective address. The index register can also be used for limited calculations and data manipulations when using read/modify/write instructions. When not required by a code sequence being executed, the index register can be used as a temporary storage area.

- **Program Counter (PC)**

The program counter is a 13-bit register that contains the address of the next instruction to be executed.

- **Stack Pointer (SP)**

The stack pointer is a 13-bit register that contains the address of the next free location on the stack. Initially, the stack pointer is set to location $007F and is decremented as data is being pushed onto the stack and incremented as data is being pulled from the stack. The six most significant bits of the stack pointer are permanently set to 00000011. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set to location $007F. Subroutines and interrupts may be nested down to location $0061 which allows the programmer to use up to 15 levels of subroutine calls.

- **Condition Code Register (CC)**

The condition code register is a 5-bit register in which each bit is used to indicate or flag the results of the instruction just executed. These bits can be individually tested by a program and specific action taken as a result of their state. Each individual condition code register bit is explained in the following paragraphs.

- **Half Carry (H)**

Used during arithmetic operations (ADD and ADC) to indicate that a carry occurred between bits 3 and 4.

- **Interrupt (I)**

This bit is set to mask the timer and external interrupt ($\overline{INT}$). If an interrupt occurs while this bit is set it is latched and will be processed as soon as the interrupt bit is reset.

- **Negative (N)**

Used to indicate that the result of the last arithmetic, logical or data manipulation was negative (bit 7 in result equal to a logical one).

- **Zero (Z)**

Used to indicate that the result of the last arithmetic, logical or data manipulation was zero.

- **Carry/Borrow (C)**

Used to indicate that a carry or borrow out of the arithmetic logic unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions, shifts, and rotates.

■ TIMER

The MCU timer circuitry is shown in Figure 3. The 8-bit counter is loaded under program control and counts down toward zero as soon as the clock input is applied. When the timer reaches zero the timer interrupt request bit (bit 7) in the timer control register is set. The MCU responds to this interrupt by saving the present MCU state in the stack, fetching the timer interrupt vector from locations $0FF8 and $0FF9 and executing the interrupt routine. The timer interrupt can be masked by setting the timer interrupt mask bit (bit 6) in the time control

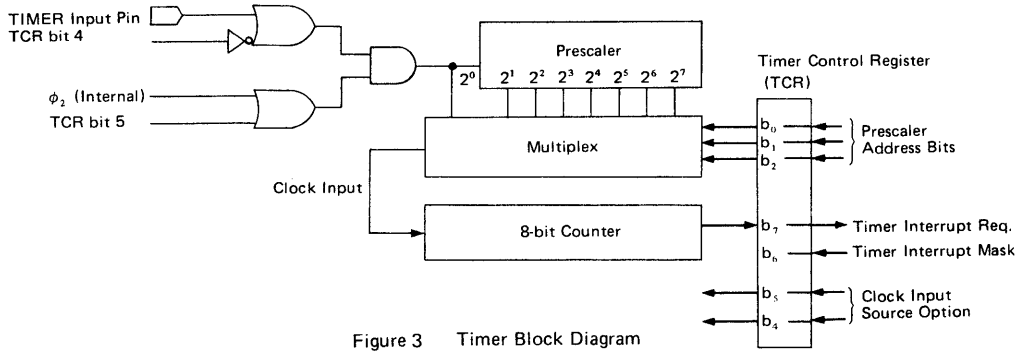


Figure 3 Timer Block Diagram

register. The interrupt bit (I bit) in the condition code register will also prevent a timer interrupt from being processed.

The clock input to the timer can be from an external source applied to the TIMER input pin or it can be the internal ϕ_2 signal. Note that when the ϕ_2 signal is used as the source it can be gated by an input applied to the TIMER input pin allowing the user to easily perform pulse-width measurements. A prescaler option can be applied to the clock input that extends the timing interval up to a maximum of 128 counts before being applied to the counter. The timer continues to count past zero and its present count can be monitored at any time by monitoring the timer data register. This allows a program to determine the length of time since a timer interrupt has occured and not disturb the counting process.

The timer data register is 8-bit read/write register with address $008 on memory-map. This timer data register and the prescaler are initialize with all logical ones at reset time.

The timer interrupt request bit (bit 7 of timer control register) is set to one by hardware when timer count reaches zero, and is cleared by program or by hardware reset. The bit 6 of timer control register is writable by program. Both of those bits can be read by MPU.

The bit 5 and bit 4 of the timer control register is a clock input source. The combinations are shown in Table 1. The bit 3 is not used. The bit 2, bit 1 and bit 0 are used to select a dividing ratio of the prescaler. The options of the dividing ratio are shown in Table 2. An internal clock is selected as a clock input source and the dividing ratio of a prescaler is set at "Bypass Prescaler" at reset time.

Table 1 Clock Input Source Option

Timer Control Register (TCR)		Clock Input Source
b_5	b_4	
0	1	ϕ_2 (Internal Clock)
1	1	TIMER Input Pin

Table 2 Prescaler Dividing Ratio Option

Timer Control Register (TCR)			Prescaler Dividing Ratio
b_2	b_1	b_0	
0	0	0	Bypass Prescaler
0	0	1	Prescaler ÷ 2
0	1	0	Prescaler ÷ 4
0	1	1	Prescaler ÷ 8
1	0	0	Prescaler ÷ 16
1	0	1	Prescaler ÷ 32
1	1	0	Prescaler ÷ 64
1	1	1	Prescaler ÷ 128

■ RESETS

The MCU can be reset three ways: by initial powerup, by the external reset input ($\overline{RES}$) and by an internal low voltage detect circuit (mask option) see Figure 4. All the I/O port are initialized to Input mode (DDR's are cleared) during RESET.

Upon power up, a minimum of 100 milliseconds is needed before allowing the reset input to go "High".

This time allows the internal crystal oscillator to stabilize. Connecting a capacitor to the $\overline{RES}$ input as shown in Figure 5 will provide sufficient delay.

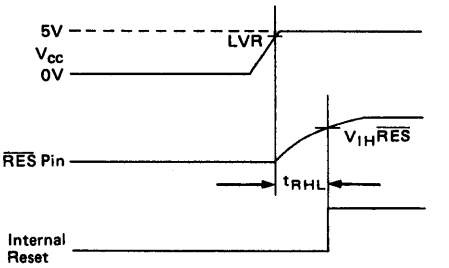


Figure 4 Power and $\overline{RES}$ Timing

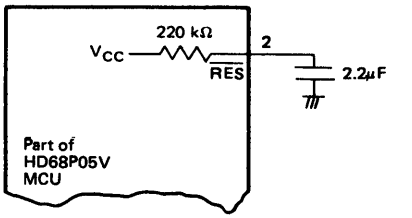


Figure 5 Power Up Reset Delay Circuit

■ INTERNAL OSCILLATOR

The internal oscillator circuit has been designed to require a minimum of external components. The use of a crystal (AT cut, 4 MHz max) is sufficient to drive the internal oscillator with varying degrees of stability. The different connection methods are shown in Figure 6. Crystal specifications are given in Figure 7.

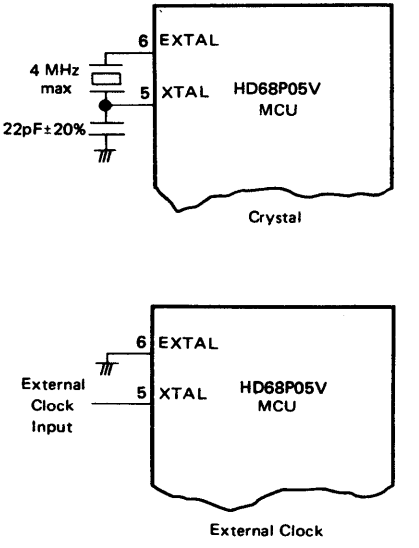
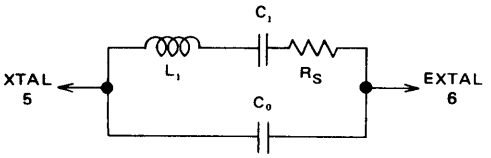


Figure 6 Internal Oscillator



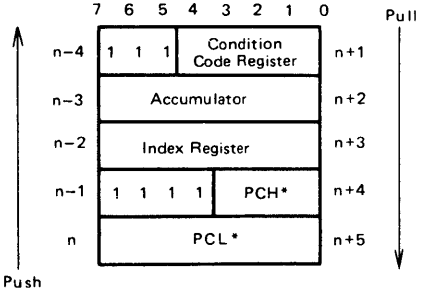
AT – Cut Parallel Resonance Crystal
C_0 = 7 pF max.
f = 4 MHz
R_S = 60 Ω max.

Figure 7 Crystal Parameters

■ INTERRUPTS

The MCU can be interrupted three different ways: through the external interrupt ($\overline{INT}$) input pin, the internal timer interrupt request, and a software interrupt instruction (SWI). When any interrupt occurs, processing is suspended, the present MCU state is pushed onto the stack in the order shown in Fig. 8, the interrupt bit (I) in the condition code register is set, the address of the interrupt routine is obtained from the appropriate interrupt vector address, and the interrupt routine is executed. Since the stack pointer decrements during pushes, the low order byte (PCL) of the program counter is stacked first; then the high order five bits (PCH) are stacked. This ensures that the program counter is loaded correctly as the stack pointer increments when it pulls data from the stack. A subroutine call will canse only the program counter (PCH, PCL) contents to be pushed onto the stack. This interrupt bit (I) in the condition code register is set, the address of the interrupt routine is obtained from the appropriate interrupt vector address, and the interrupt routine is executed. The interrupt service routines normally end with a return from interrupt (RTI) instruction which allows the MCU to resume processing of the program prior to the interrupt. Table 3 provides a listing of the interrupts, their priority, and the vector address that contain the starting address of the appropriate interrupt routine.

A flowchart of the interrupt processing sequence is given in Fig. 9.



* For subroutine calls, only PCH and PCL are stacked.

Figure 8 Interrupt Stacking Order

Table 3 Interrupt Priorities

Interrupt	Priority	Vector Address
$\overline{RES}$	1	$0FFE and $0FFF
SWI	2	$0FFC and $0FFD
$\overline{INT}$	3	$0FFA and $0FFB
TIMER	4	$0FF8 and $0FF9

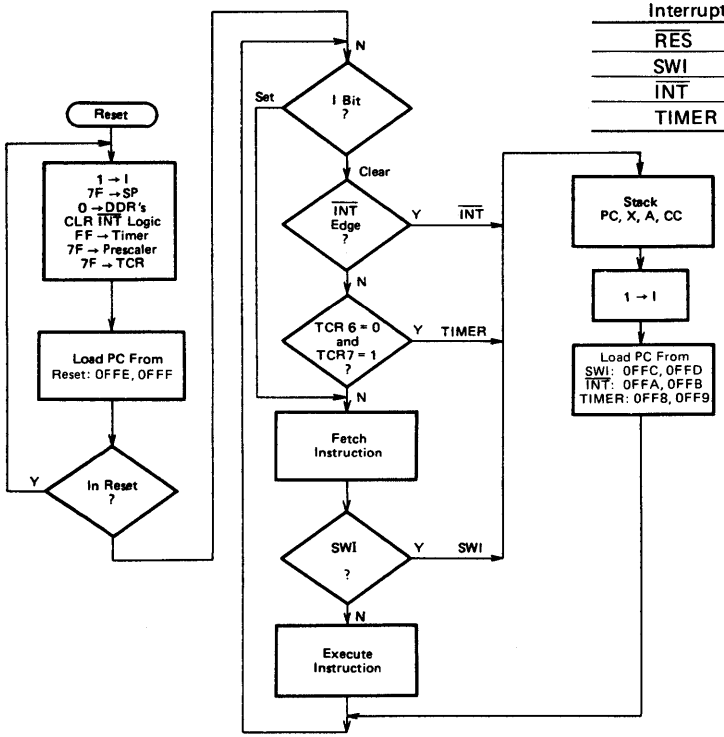
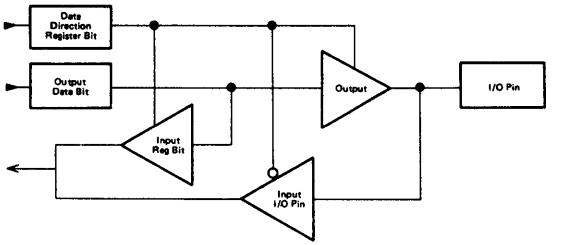


Figure 9 Interrupt Processing Flowchart

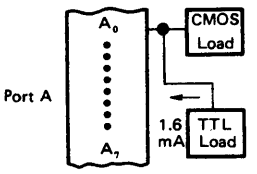
■ INPUT/OUTPUT

There are 24 input/output pins. All pins are programmable as either inputs or outputs under software control of the data direction registers. When programmed as outputs, all I/O pins read latched output data regardless of the logic level at the output pin due to output loading (see Fig. 10). When port B is programmed for outputs, it is capable of sinking 10 millamperes on each pin (V_{OL} = 1V max). All input/output lines are TTL compatible as both inputs and outputs. Port A lines are CMOS compatible as outputs while port B and C lines are CMOS compatible as inputs. Figure 11 provides some examples of port connections.

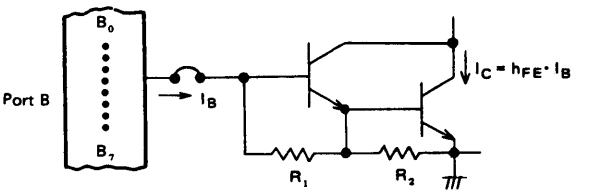


Data Direction Register Bit	Output Data Bit	Output State	Input to MCU
1	0	0	0
1	1	1	1
0	×	3-State	Pin

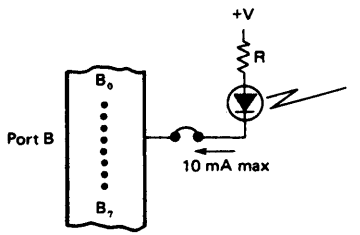
Figure 10 Typical Port I/O Circuitry



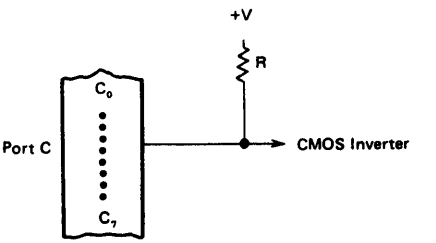
Port A Programmed as output(s) driving CMOS and TTL Load directly.
(a)



Port B Programmed as output(s) driving Darlington base directly.
(b)



Port B Programmed as output(s) driving LED(s) directly.
(c)



Port C Programmed as output(s) driving CMOS using external pull-up resistors.
(d)

Figure 11 Typical Port Connections

■ INPUT

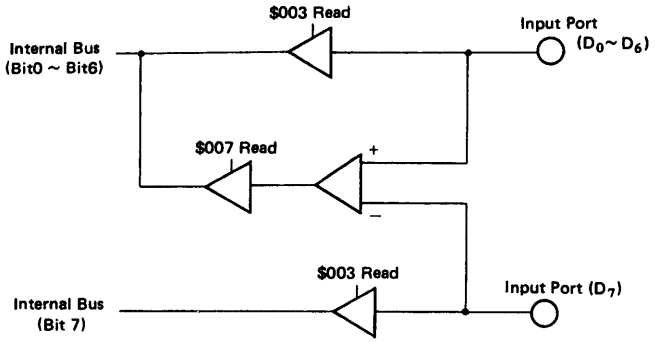
Port D is 8-bit input port, which has two functions. One of them is usual digital signal input port and the other is voltage compare type input port. In the former case, the input data can be read by MPU at $003 address. In the latter case, D_7 (pin 17) is the input pin of V_{TH} (reference level), and the other seven input pins ($D_0 \sim D_6$) are analog level inputs, which are compared with V_{TH} (see Figure 12(a), (b)).

"1" or "0" signals appear at internal data bus, if the input levels are higher or lower respectively when $007 address is read. This function is effective in such case that unusual logic level inputs are used. A capacitive touch panel interface and a diode isolated keyboard interface are the examples. Figure 12(c) shows the application of Port D to A/D converter, and Figure 12(d) shows 3 levels inputs.

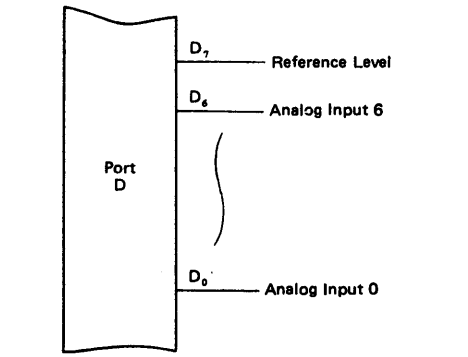
■ BIT MANIPULATION

The MCU has the ability to set or clear any single random access memory or input/output bit (except the data direction registers) with a single instruction (BSET, BCLR). Any bit in the page zero read only memory can be tested, using the BRSET and BRCLR instructions, and the program branches as a result of its state. This capability to work with any bit in RAM, ROM or I/O allows the user to have individual flags in RAM or to handle single I/O bits as control lines. The example in Figure 13 illustrates the usefulness of the bit manipulation and test instructions. Assume that bit 0 of port A is connected to a zero crossing detector circuit and that bit 1 of port A is connected to the trigger of a TRIAC which power the controlled hardware.

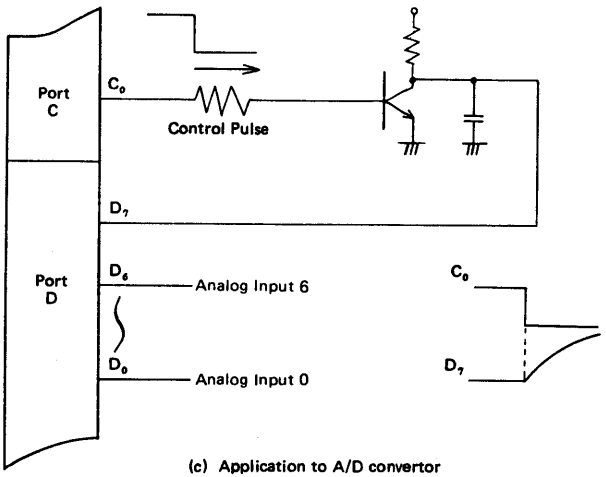
This program, which uses only seven ROM locations, provides turn-on of the TRIAC within 14 microseconds of the zero crossing. The timer could also be incorporated to provide turn-on at some later time which would permit pulse-width modulation of the controlled power.



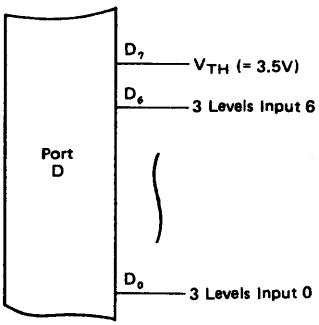
(a) The logic configuration of Port D



(b) Seven analog inputs and a reference level input of Port D



(c) Application to A/D convertor



(d) Application to 3 levels input

Input Voltage	($003)	($007)
0V ~ 0.8V	0	0
2.0V ~ 3.3V	1	0
3.7V ~ V_{CC}	1	1

Figure 12 Configuration and Application of Port D

```
SELF 1    BRCLR 0, PORT A, SELF 1
          BSET 1, PORT A
          BCLR 1, PORT A
```

Figure 13 Bit Manipulation Example

■ ADDRESSING MODES

The MCU has ten addressing modes available for use by the programmer. They are explained and illustrated briefly in the following paragraphs.

- **Immediate**

Refer to Figure 14. The immediate addressing mode accesses constants which do not change during program execution. Such instructions are two bytes long. The effective address (EA) is the PC and the operand is fetched from the byte following the opcode.

- **Direct**

Refer to Figure 15. In direct addressing, the address of the operand is contained in the second byte of the instruction. Direct addressing allows the user to directly address the lowest 256 bytes in memory. All RAM space, I/O registers and 128 bytes of ROM are located in page zero to take advantage of this efficient memory addressing mode.

- **Extended**

Refer to Figure 16. Extended addressing is used to reference any location in memory space. The EA is the contents of the two bytes following the opcode. Extended addressing instructions are three bytes long.

- **Relative**

Refer to Figure 17. The relative addressing mode applies only to the branch instructions. In this mode the contents of the byte following the opcode is added to the program counter when the branch is taken. EA=(PC)+2+Rel. Rel is the contents of the location following the instruction opcode with bit 7 being the sign bit. If the branch is not taken Rel=0, when a branch takes place, the program goes to somewhere within the range of +129 bytes to -127 of the present instruction. These instructions are two bytes long.

- **Indexed (No Offset)**

Refer to Figure 18. This mode of addressing accesses the lowest 256 bytes of memory. These instructions are one byte long and their EA is the contents of the index register.

- **Indexed (8-bit Offset)**

Refer to Figure 19. The EA is calculated by adding the contents of the byte following the opcode to the contents of the index register. In this mode, 511 low memory locations are accessable. These instructions occupy two bytes.

- **Indexed (16-bit Offset)**

Refer to Figure 20. This addressing mode calculates the EA by adding the contents of the two bytes following the opcode to the index register. Thus, the entire memory space may be accessed. Instructions which use this addressing mode are three bytes long.

- **Bit Set/Clear**

Refer to Figure 21. This mode of addressing applies to instructions which can set or clear any bit on page zero. The lower three bits in the opcode specify the bit to be set or cleared while the byte following the opcode specifies the address in page zero.

- **Bit Test and Branch**

Refer to Figure 22. This mode of addressing applies to instructions which can test any bit in the first 256 locations ($00-$FF) and branch to any location relative to the PC. The byte to be tested is addressed by the byte following the opcode. The individual bit within that byte to be tested is addressed by the lower three bits of the opcode. The third byte is the relative address to be added to the program counter if the branch condition is met. These instructions are three bytes long. The value of the bit tested is written to the carry bit in the condition code register.

- **Implied**

Refer to Figure 23. The implied mode of addressing has no EA. All the information necessary to execute an instruction is contained in the opcode. Direct operations on the accumulator and the index register are included in this mode of addressing. In addition, control instructions such as SWI, RTI belong to this group. All implied addressing instructions are one byte long.

■ INSTRUCTION SET

The MCU has a set of 59 basic instructions. They can be divided into five different types: register/memory, read/modify/write, branch, bit manipulation, and control. The following paragraphs briefly explain each type. All the instructions within a given type are presented in individual tables.

- **Register/Memory Instructions**

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to Table 4.

- **Read/Modify/Write Instructions**

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read/modify/write instructions since it does not perform the write. Refer to Table 5.

- **Branch Instructions**

The branch instructions cause a branch from the program when a certain condition is met. Refer to Table 6.

- **Bit Manipulation Instructions**

These instructions are used on any bit in the first 256 bytes of the memory. One group either sets or clears. The other group performs the bit test and branch operations. Refer to Table 7.

- **Control Instructions**

The control instructions control the MCU operations during program execution. Refer to Table 8.

- **Alphabetical Listing**

The complete instruction set is given in alphabetical order in Table 9.

- **Opcode Map**

Table 10 is an opcode map for the instructions used on the MCU.

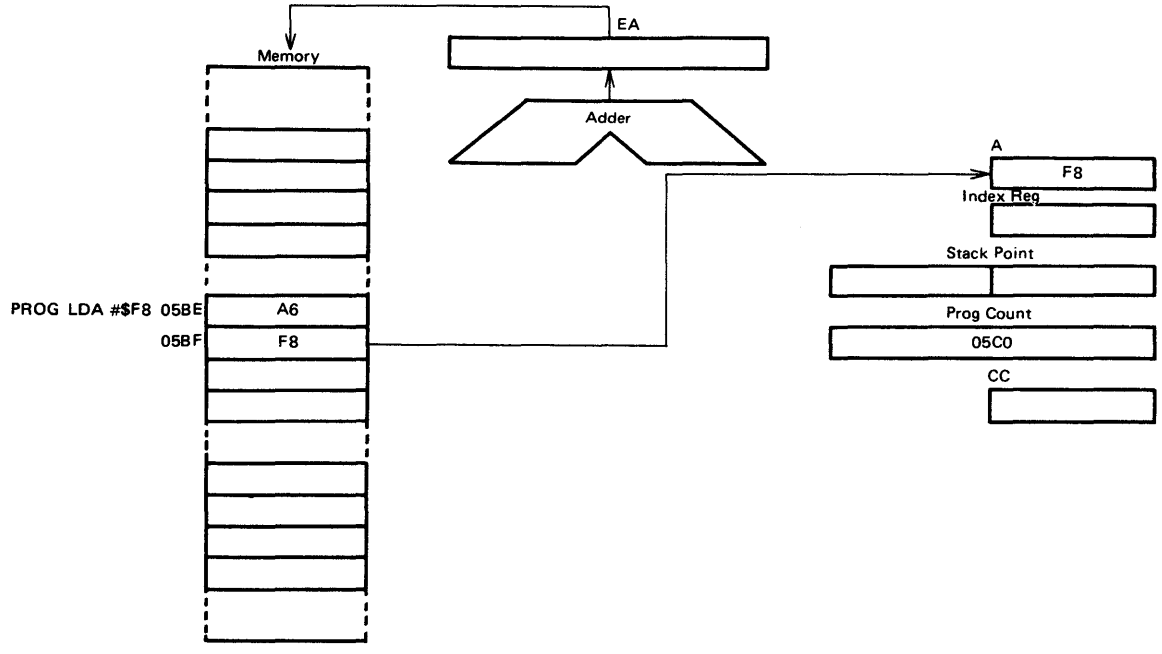


Figure 14 Immediate Addressing Example

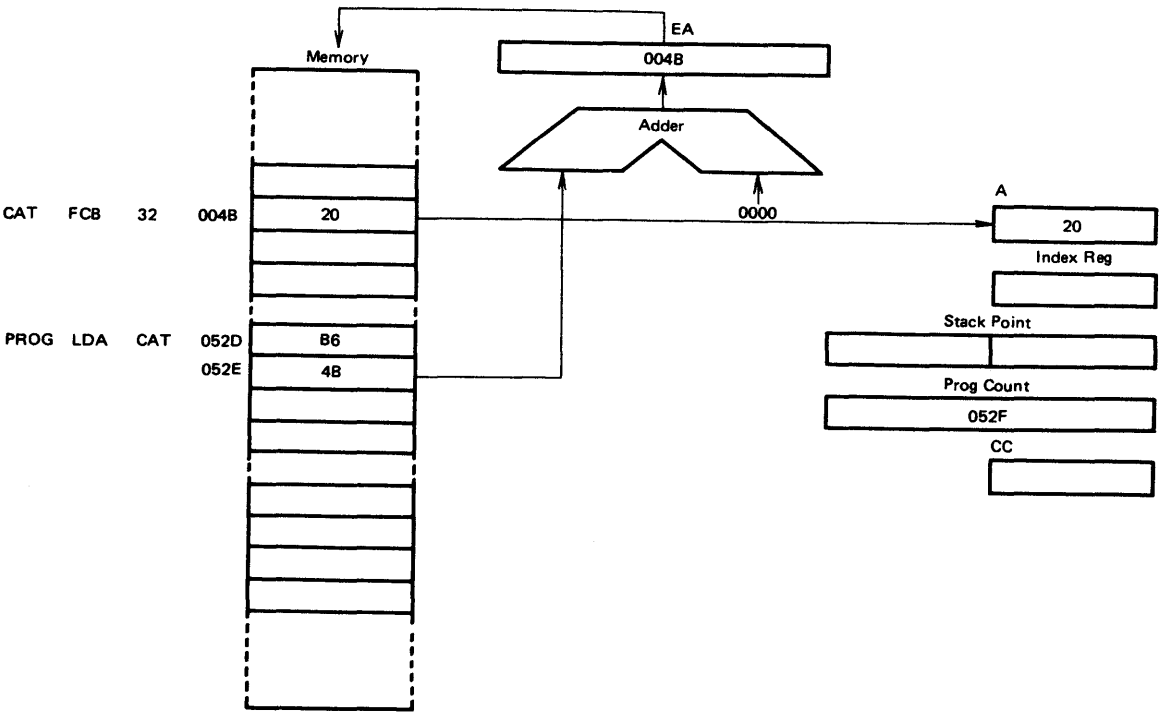


Figure 15 Direct Addressing Example

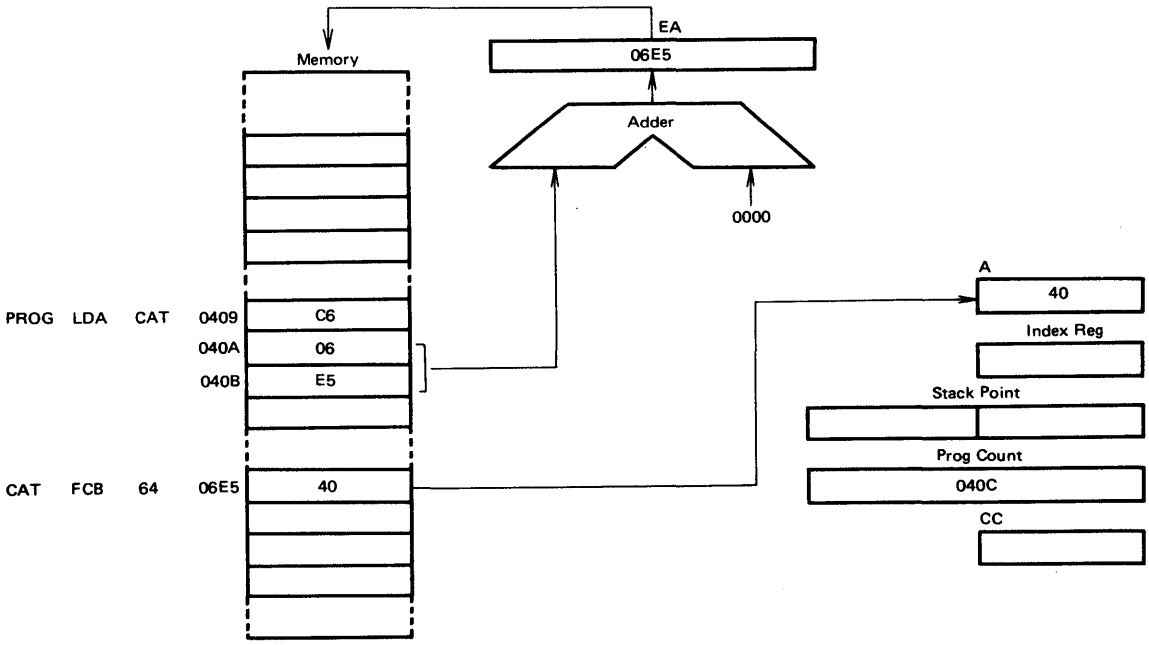


Figure 16 Extended Addressing Example

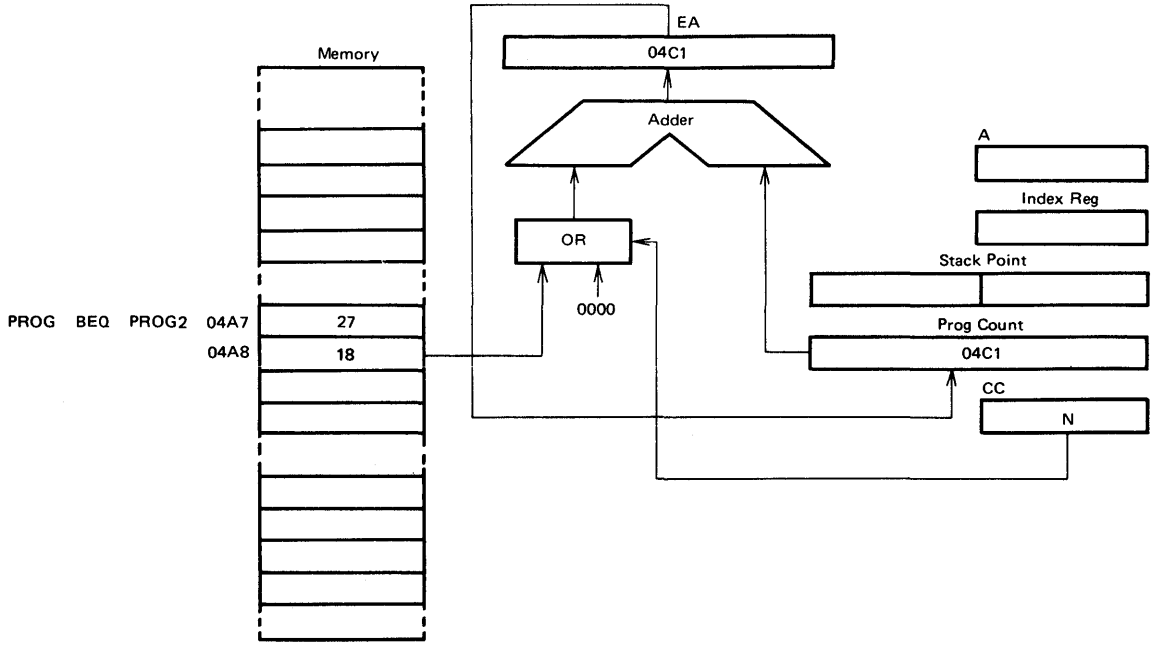


Figure 17 Relative Addressing Example

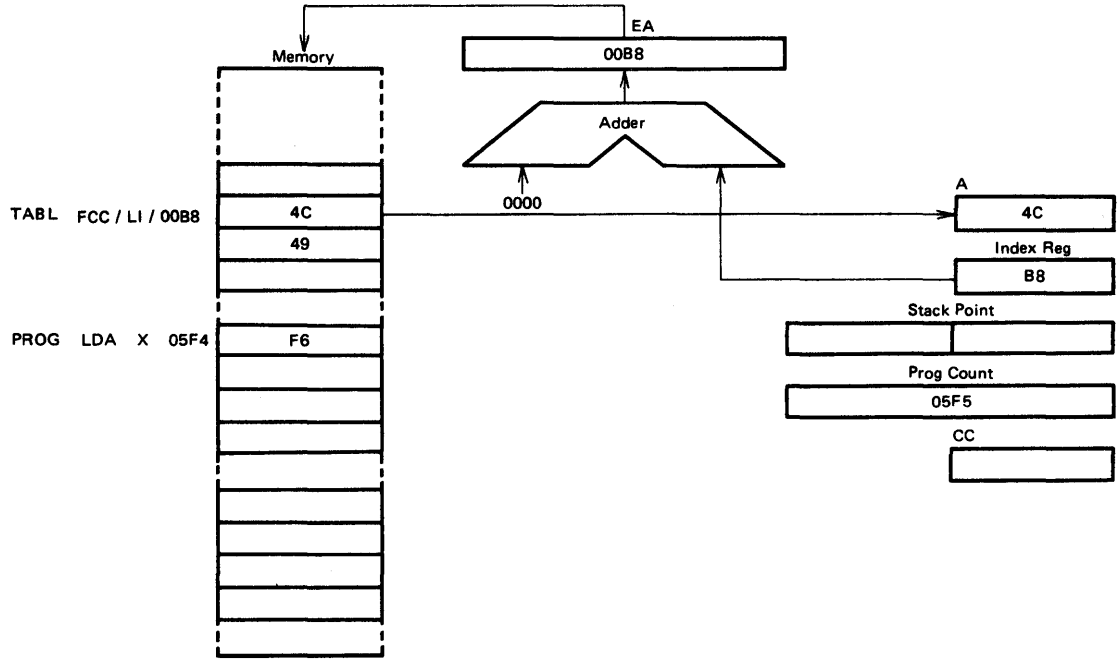


Figure 18 Indexed (No Offset) Addressing Example

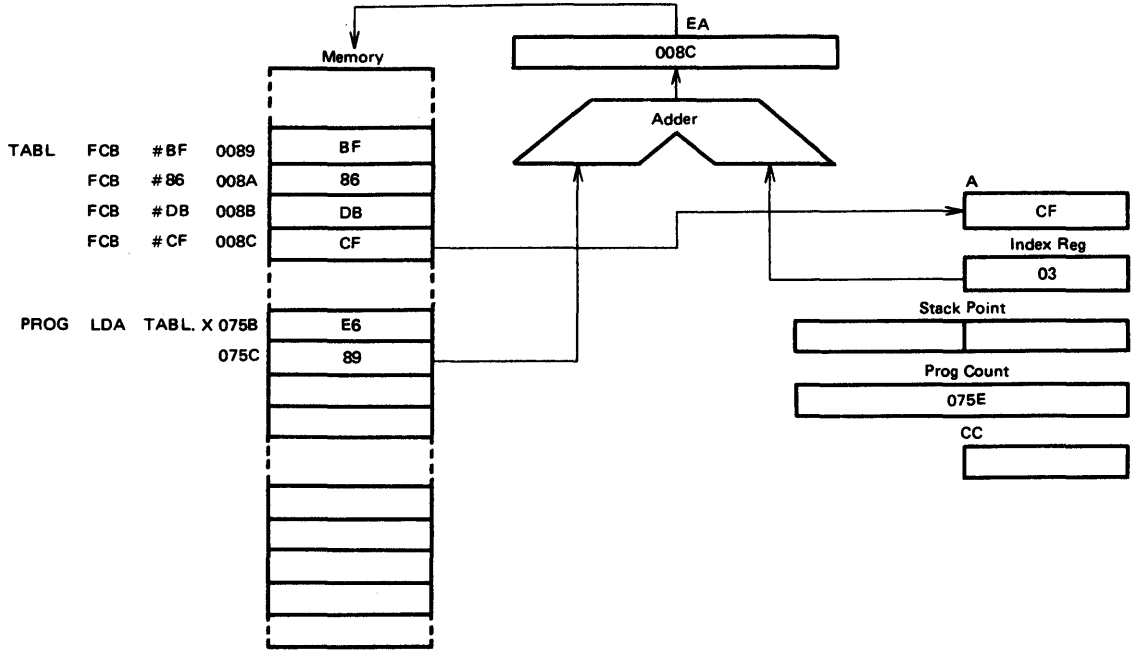


Figure 19 Indexed (8-Bit Offset) Addressing Example

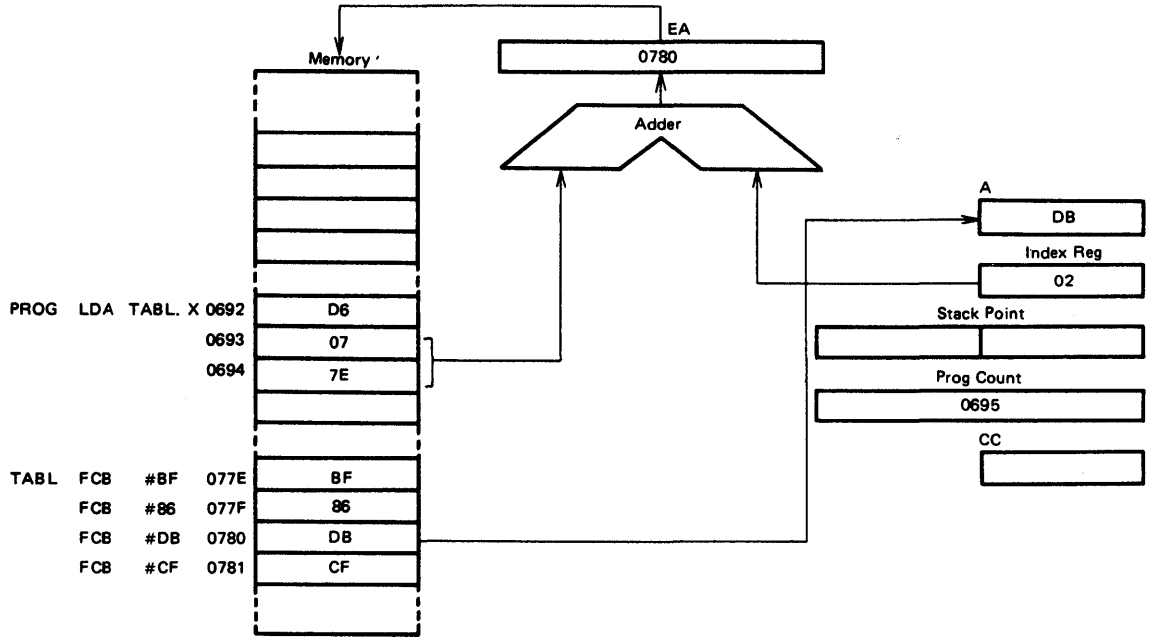


Figure 20 Indexed (16-Bit Offset) Addressing Example

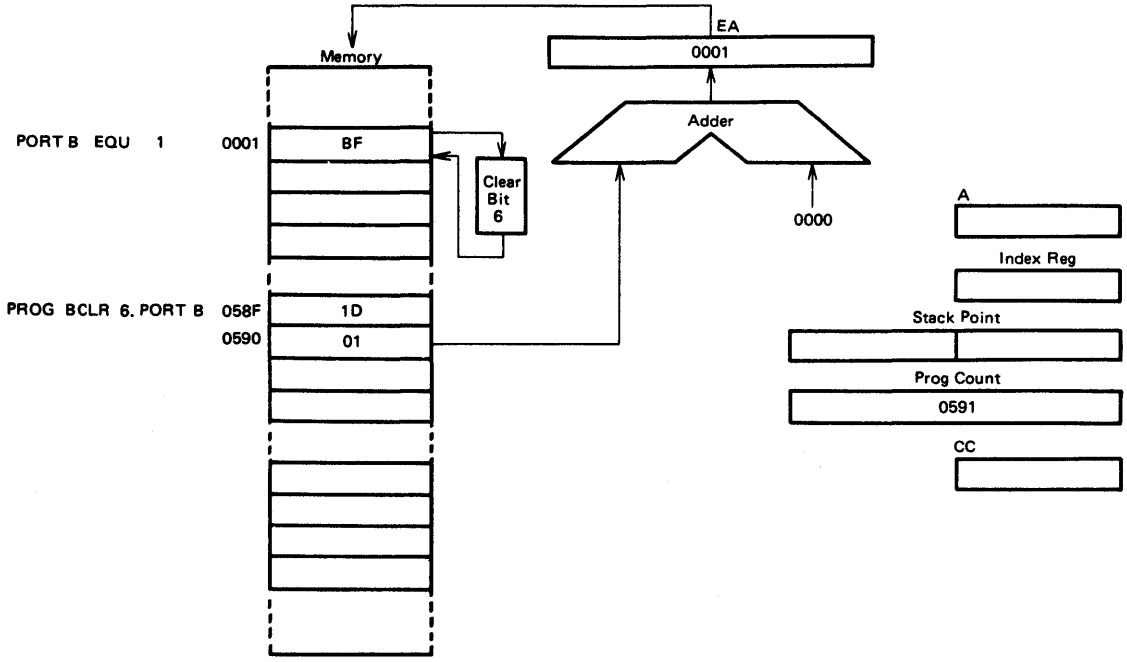


Figure 21 Bit Set/Clear Addressing Example

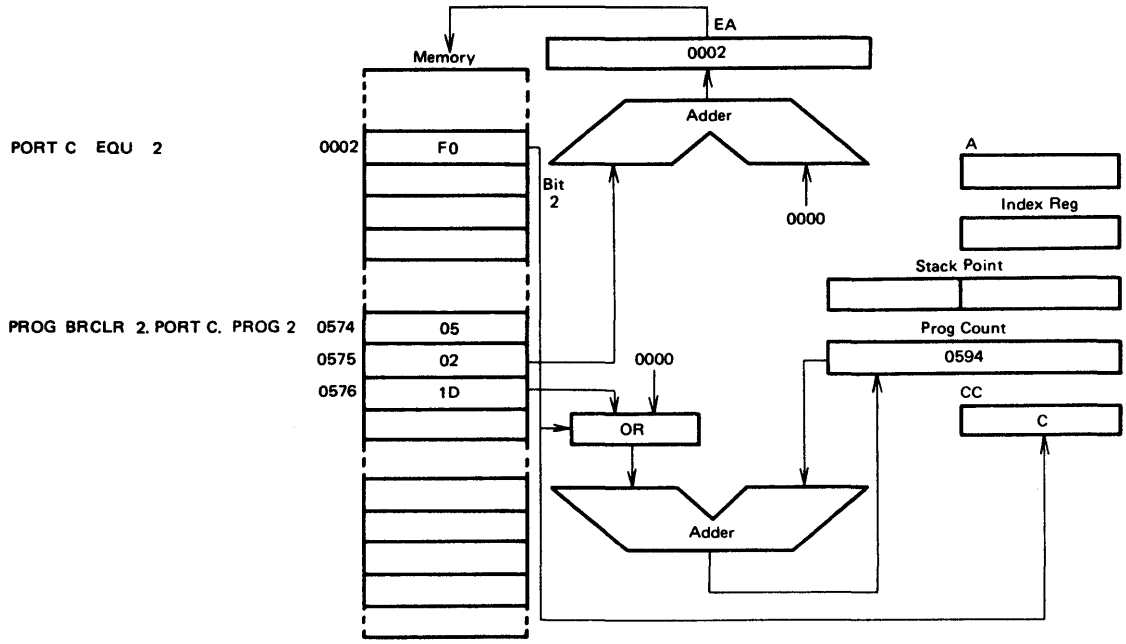


Figure 22 Bit Test and Branch Addressing Example

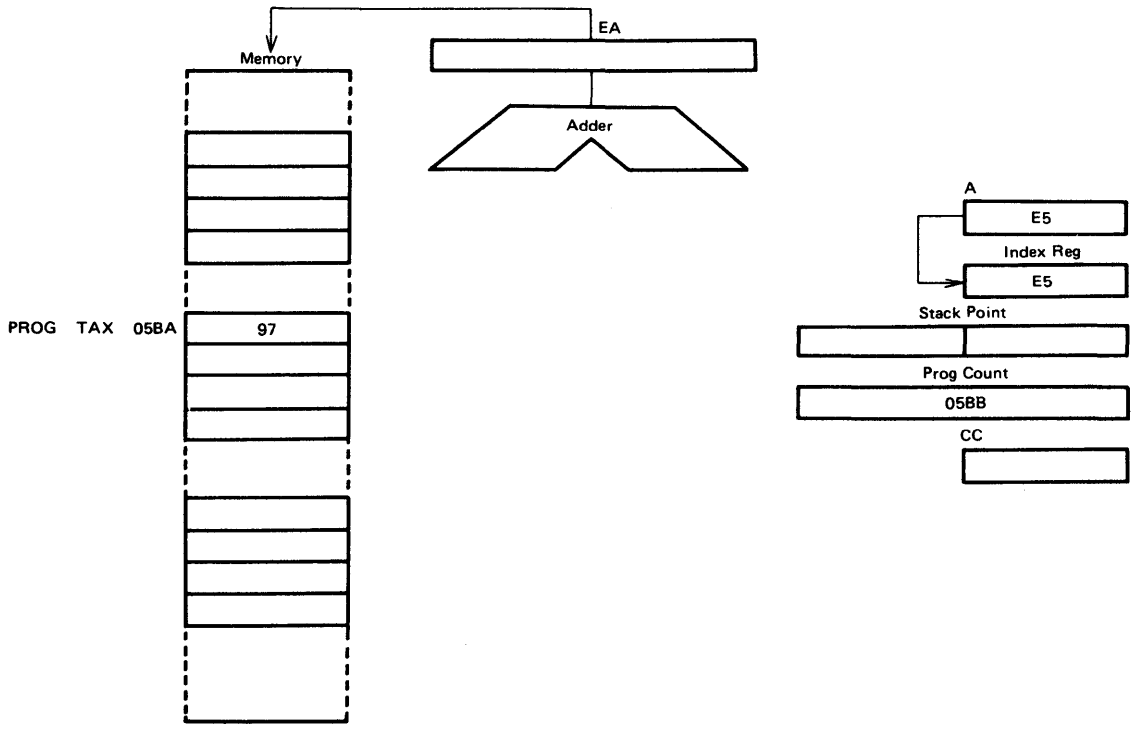


Figure 23 Implied Addressing Example

Table 4 Register/Memory Instructions

Function	Mnemonic	Addressing Modes																	
		Immediate			Direct			Extended			Indexed (No Offset)			Indexed (8-Bit Offset)			Indexed (16-Bit Offset)		
		Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Load A from Memory	LDA	A6	2	2	B6	2	4	C6	3	5	F6	1	4	E6	2	5	D6	3	6
Load X from Memory	LDX	AE	2	2	BE	2	4	CE	3	5	FE	1	4	EE	2	5	DE	3	6
Store A in Memory	STA	–	–	–	B7	2	5	C7	3	6	F7	1	5	E7	2	6	D7	3	7
Store X in Memory	STX	–	–	–	BF	2	5	CF	3	6	FF	1	5	EF	2	6	DF	3	7
Add Memory to A	ADD	AB	2	2	BB	2	4	CB	3	5	FB	1	4	EB	2	5	DB	3	6
Add Memory and Carry to A	ADC	A9	2	2	B9	2	4	C9	3	5	F9	1	4	E9	2	5	D9	3	6
Subtract Memory	SUB	A0	2	2	B0	2	4	C0	3	5	F0	1	4	E0	2	5	D0	3	6
Subtract Memory from A with Borrow	SBC	A2	2	2	B2	2	4	C2	3	5	F2	1	4	E2	2	5	D2	3	6
AND Memory to A	AND	A4	2	2	B4	2	4	C4	3	5	F4	1	4	E4	2	5	D4	3	6
OR Memory with A	ORA	AA	2	2	BA	2	4	CA	3	5	FA	1	4	EA	2	5	DA	3	6
Exclusive OR Memory with A	EOR	A8	2	2	B8	2	4	C8	3	5	F8	1	4	E8	2	5	D8	3	6
Arithmetic Compare A with Memory	CMP	A1	2	2	B1	2	4	C1	3	5	F1	1	4	E1	2	5	D1	3	6
Arithmetic Compare X with Memory	CPX	A3	2	2	B3	2	4	C3	3	5	F3	1	4	E3	2	5	D3	3	6
Bit Test Memory with A (Logical Compare)	BIT	A5	2	2	B5	2	4	C5	3	5	F5	1	4	E5	2	5	D5	3	6
Jump Unconditional	JMP	–	–	–	BC	2	3	CC	3	4	FC	1	3	EC	2	4	DC	3	5
Jump to Subroutine	JSR	–	–	–	BD	2	7	CD	3	8	FD	1	7	ED	2	8	DD	3	9

Table 5 Read/Modify/Write Instructions

Function	Mnemonic	Addressing Modes														
		Implied (A)			Implied (X)			Direct			Indexed (No Offset)			Indexed (8-Bit Offset)		
		Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Increment	INC	4C	1	4	5C	1	4	3C	2	6	7C	1	6	6C	2	7
Decrement	DEC	4A	1	4	5A	1	4	3A	2	6	7A	1	6	6A	2	7
Clear	CLR	4F	1	4	5F	1	4	3F	2	6	7F	1	6	6F	2	7
Complement	COM	43	1	4	53	1	4	33	2	6	73	1	6	63	2	7
Negate (2's Complement)	NEG	40	1	4	50	1	4	30	2	6	70	1	6	60	2	7
Rotate Left Thru Carry	ROL	49	1	4	59	1	4	39	2	6	79	1	6	69	2	7
Rotate Right Thru Carry	ROR	46	1	4	56	1	4	36	2	6	76	1	6	66	2	7
Logical Shift Left	LSL	48	1	4	58	1	4	38	2	6	78	1	6	68	2	7
Logical Shift Right	LSR	44	1	4	54	1	4	34	2	6	74	1	6	64	2	7
Arithmetic Shift Right	ASR	47	1	4	57	1	4	37	2	6	77	1	6	67	2	7
Arithmetic Shift Left	ASL	48	1	4	58	1	4	38	2	6	78	1	6	68	2	7
Test for Negative or Zero	TST	4D	1	4	5D	1	4	3D	2	6	7D	1	6	6D	2	7

Table 6 Branch Instructions

Function	Mnemonic	Relative Addressing Mode		
		Op Code	# Bytes	# Cycles
Branch Always	BRA	20	2	4
Branch Never	BRN	21	2	4
Branch IF Higher	BHI	22	2	4
Branch IF Lower or Same	BLS	23	2	4
Branch IF Carry Clear	BCC	24	2	4
(Branch IF Higher or Same)	(BHS)	24	2	4
Branch IF Carry Set	BCS	25	2	4
(Branch IF Lower)	(BLO)	25	2	4
Branch IF Not Equal	BNE	26	2	4
Branch IF Equal	BEQ	27	2	4
Branch IF Half Carry Clear	BHCC	28	2	4
Branch IF Half Carry Set	BHCS	29	2	4
Branch IF Plus	BPL	2A	2	4
Branch IF Minus	BMI	2B	2	4
Branch IF Interrupt Mask Bit is Clear	BMC	2C	2	4
Branch IF Interrupt Mask Bit is Set	BMS	2D	2	4
Branch IF Interrupt Line is Low	BIL	2E	2	4
Branch IF Interrupt Line is High	BIH	2F	2	4
Branch to Subroutine	BSR	AD	2	8

Table 7 Bit Manipulation Instructions

Function	Mnemonic	Addressing Modes					
		Bit Set/Clear			Bit Test and Branch		
		Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Branch IF Bit n is set	BRSET n (n=0 ..... 7)	—	—	—	2•n	3	10
Branch IF Bit n is clear	BRCLR n (n=0 .....7)	—	—	—	01+2•n	3	10
Set Bit n	BSET n (n=0 ..... 7)	10+2•n	2	7	—	—	—
Clear bit n	BCLR n (n=0 ..... 7)	11+2•n	2	7	—	—	—

Table 8 Control Instructions

Function	Mnemonic	Implied		
		Op Code	# Bytes	# Cycles
Transfer A to X	TAX	97	1	2
Transfer X to A	TXA	9F	1	2
Set Carry Bit	SEC	99	1	2
Clear Carry Bit	CLC	98	1	2
Set Interrupt Mask Bit	SEI	9B	1	2
Clear Interrupt Mask Bit	CLI	9A	1	2
Software Interrupt	SWI	83	1	11
Return from Subroutine	RTS	81	1	6
Return from Interrupt	RTI	80	1	9
Reset Stack Pointer	RSP	9C	1	2
No-Operation	NOP	9D	1	2

Table 9 Instruction Set

Mnemonic	Addressing Modes										Condition Code				
	Implied	Immediate	Direct	Extended	Relative	Indexed (No Offset)	Indexed (8 Bits)	Indexed (16 Bits)	Bit Set/ Clear	Bit Test & Branch	H	I	N	Z	C
ADC		x	x	x		x	x	x			∧	•	∧	∧	∧
ADD		x	x	x		x	x	x			∧	•	∧	∧	∧
AND		x	x	x		x	x	x			•	•	∧	∧	•
ASL	x		x			x	x				•	•	∧	∧	∧
ASR	x		x			x	x				•	•	∧	∧	∧
BCC					x						•	•	•	•	•
BCLR									x		•	•	•	•	•
BCS					x						•	•	•	•	•
BEQ					x						•	•	•	•	•
BHCC					x						•	•	•	•	•
BHCS					x						•	•	•	•	•
BHI					x						•	•	•	•	•
BHS					x						•	•	•	•	•
BIH					x						•	•	•	•	•
BIL					x						•	•	•	•	•
BIT		x	x	x		x	x	x			•	•	∧	∧	•
BLO					x						•	•	•	•	•
BLS					x						•	•	•	•	•
BMC					x						•	•	•	•	•
BMI					x						•	•	•	•	•
BMS					x						•	•	•	•	•
BNE					x						•	•	•	•	•
BPL					x						•	•	•	•	•
BRA					x						•	•	•	•	•
BRN					x						•	•	•	•	•
BRCLR										x	•	•	•	•	∧
BRSET										x	•	•	•	•	∧
BSET									x		•	•	•	•	•
BSR					x						•	•	•	•	•
CLC	x										•	•	•	•	0
CLI	x										•	0	•	•	•
CLR	x		x			x	x				•	•	0	1	•
CMP		x	x	x		x	x	x			•	•	∧	∧	∧
COM	x		x			x	x				•	•	∧	∧	1
CPX		x	x	x		x	x	x			•	•	∧	∧	∧
DEC	x		x			x	x				•	•	∧	∧	•
EOR		x	x	x		x	x	x			•	•	∧	∧	•
INC	x		x			x	x				•	•	∧	∧	•
JMP			x	x		x	x	x			•	•	•	•	•
JSR			x	x		x	x	x			•	•	•	•	•
LDA		x	x	x		x	x	x			•	•	∧	∧	•
LDX		x	x	x		x	x	x			•	•	∧	∧	•

(to be continued)

Condition Code Symbols:
- H Half Carry (From Bit 3)
- I Interrupt Mask
- N Negative (Sign Bit)
- Z Zero
- C Carry Borrow
- ∧ Test and Set if True, Cleared Otherwise
- • Not Affected

Table 9 Instruction Set

Mnemonic	Addressing Modes										Condition Code				
	Implied	Immediate	Direct	Extended	Relative	Indexed (No Offset)	Indexed (8 Bits)	Indexed (16 Bits)	Bit Set/ Clear	Bit Test & Branch	H	I	N	Z	C
LSL	x		x			x	x				•	•	∧	∧	∧
LSR	x		x			x	x				•	•	0	∧	∧
NEG	x		x			x	x				•	•	∧	∧	∧
NOP	x										•	•	•	•	•
ORA		x	x	x		x	x	x			•	•	∧	∧	•
ROL	x		x			x	x				•	•	∧	∧	∧
ROR	x		x			x	x				•	•	∧	∧	∧
RSP	x										•	•	•	•	•
RTI	x										?	?	?	?	?
RTS	x										•	•	•	•	•
SBC		x	x	x		x	x	x			•	•	∧	∧	∧
SEC	x										•	•	•	•	1
SEI	x										•	1	•	•	•
STA			x	x		x	x	x			•	•	∧	∧	•
STX			x	x		x	x	x			•	•	∧	∧	•
SUB		x	x	x		x	x	x			•	•	∧	∧	∧
SWI	x										•	1	•	•	•
TAX	x										•	•	•	•	•
TST	x		x			x	x				•	•	∧	∧	•
TXA	x										•	•	•	•	•

Condition Code Symbols:
- H Half Carry (From Bit 3)
- I Interrupt Mask
- N Negative (Sign Bit)
- Z Zero
- C Carry/Borrow
- ∧ Test and Set if True, Cleared Otherwise
- • Not Affected
- ? Load CC Register From Stack

Table 10 Opcode Map

	Bit Manipulation		Branch	Read/Modify/Write					Control		Register/Memory						
	Test & Branch	Set/ Clear	Rel	DIR	A	X	,X1	,X0	IMP	IMP	IMM	DIR	EXT	,X2	,X1	,X0	
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	← HIGH
0	BRSET0	BSET0	BRA	NEG					RTI*	–	SUB						0
1	BRCLR0	BCLR0	BRN	–					RTS*	–	CMP						1
2	BRSET1	BSET1	BHI	–					–	–	SBC						2
3	BRCLR1	BCLR1	BLS	COM					SWI*	–	CPX						3 L
4	BRSET2	BSET2	BCC	LSR					–	–	AND						4 O
5	BRCLR2	BCLR2	BCS	–					–	–	BIT						5 W
6	BRSET3	BSET3	BNE	ROR					–	–	LDA						6
7	BRCLR3	BCLR3	BEQ	ASR					–	TAX	–	STA(+1)					7
8	BRSET4	BSET4	BHCC	LSL/ASL					–	CLC	EOR						8
9	BRCLR4	BCLR4	BHCS	ROL					–	SEC	ADC						9
A	BRSET5	BSET5	BPL	DEC					–	CLI	ORA						A
B	BRCLR5	BCLR5	BMI	–					–	SEI	ADD						B
C	BRSET6	BSET6	BMC	INC					–	RSP	–	JMP(–1)					C
D	BRCLR6	BCLR6	BMS	TST					–	NOP	BSR*	JSR(–3)					D
E	BRSET7	BSET7	BIL	–					–	–	LDX						E
F	BRCLR7	BCLR7	BIH	CLR					–	TXA	–	STX(+1)					F
	3/10	2/7	2/4	2/6	1/4	1/4	2/7	1/6	1/*	1/2	2/2	2/4	3/5	3/6	2/5	1/4	

(NOTE) 1. Undefined opcodes are marked with "–".
2. The number at the bottom of each column denote the number of bytes and the number of cycles required (Bytes/Cycles).
Mnemonics followed by a "*" require a different number of cycles as follows:

RTI	9
RTS	6
SWI	11
BSR	8

3. ( ) indicate that the number in parenthesis must be added to the cycle count for that instruction.

■ HD68P05V USED FOR HD6805U/V

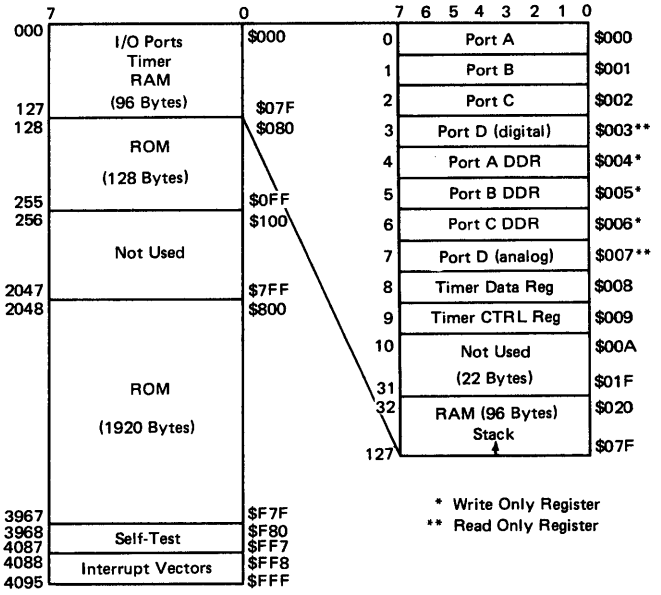
Fig. 25 provides the memory configuration of MCU. Fig. 25(a) provides the configuration of HD68P05V used for HD6805U. "Not Used" memory map may be used for HD68P05V but not used for HD6805U. If used for HD6805V, HD68P05V will have the configuration shown in Fig. 25(b).

The timer part of HD6805U/V is mask-option. If used for HD6805U/V, HD68P05V sets the bit 0 to 5 of timer control register in the program just after reset and selects the dividing ratio of the prescaler and the clock input source. Fig. 24 shows an example of the program which selects the external clock as an input source at 128 dividing.

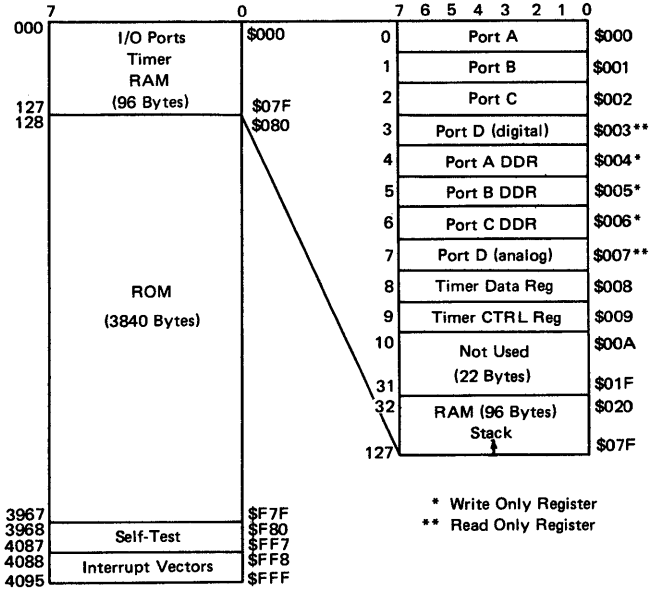
If the program specified by the HD68P05V is masked as HD6805U/V, the command to operate this bit is ignored because HD6805U/V doesn't have the bit 0 to 5 of the timer control register.

```
LDA  #$77
STA  TCR ($009)
```

Figure 24 Example to initialize timer control register (TCR)



(a) HD6805U Configuration



(b) HD6805V Configuration

Figure 25 MCU Memory Configuration

8-BIT MICROCOMPUTER HMCS6800 MULTI-CHIP SERIES

HD6800, HD68A00, HD68B00—MPU (Micro Processing Unit)

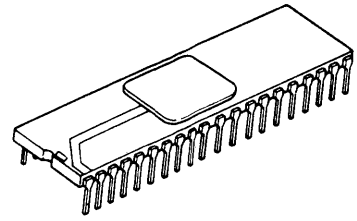
The HD6800 is a monolithic 8-bit microprocessor forming the central control function for Hitachi's HMCS6800 family. Compatible with TTL, the HD6800 as with all HMCS6800 system parts, requires only one 5V power supply, and no external TTL devices for bus interface. The HD68A00 and HD68B00 are high speed versions.

The HD6800 is capable of addressing 65k bytes of memory with its 16-bit address lines. The 8-bit data bus is bi-directional as well as 3-state, making direct memory addressing and multiprocessing applications realizable.

■ FEATURES

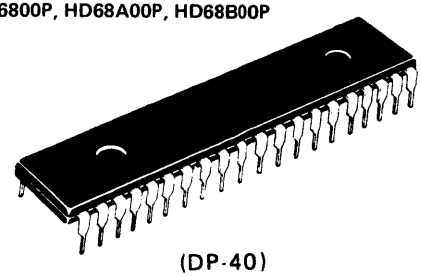
- Versatile 72 Instruction – Variable Length (1~3 Byte)
- Seven Addressing Modes – Direct, Relative, Immediate, Indexed, Extended, Implied and Accumulator
- Variable Length Stack
- Vectored Restart
- Maskable Interrupt
- Separate Non-Maskable Interrupt – Internal Registers Saved in Stack
- Six Internal Registers – Two Accumulators, Index Register, Program Counter, Stack Pointer and Condition Code Register
- Direct Memory Accessing (DMA) and Multiple Processor Capability
- Clock Rates as High as 2.0 MHz (HD6800 ··· 1 MHz, HD68A00 ··· 1.5 MHz, HD68B00 ··· 2.0 MHz)
- Halt and Single Instruction Execution Capability
- Compatible with MC6800, MC68A00 and MC68B00

HD6800, HD68A00, HD68B00



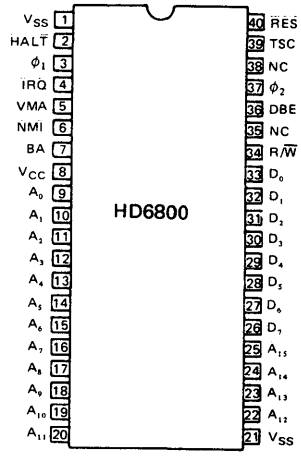
(DC-40)

HD6800P, HD68A00P, HD68B00P



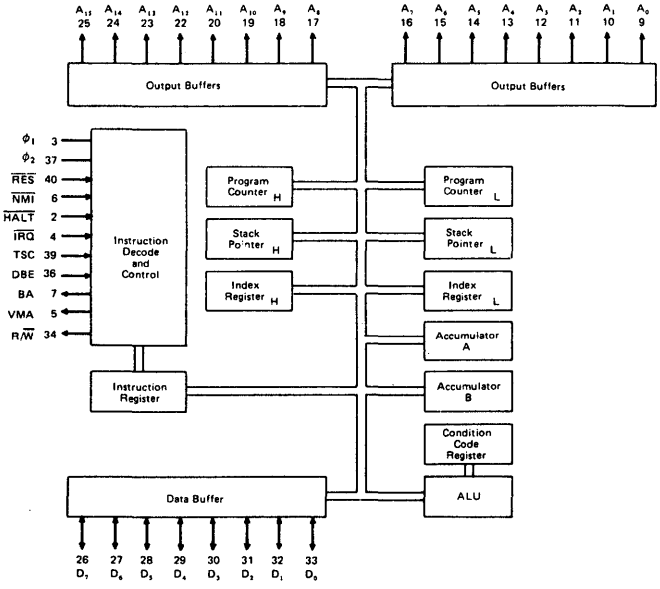
(DP-40)

■ PIN ARRANGEMENT



(Top View)

■ BLOCK DIAGRAM



■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC}*	−0.3 ~ +7.0	V
Input Voltage	V_{in}*	−0.3 ~ +7.0	V
Operating Temperature	T_{opr}	− 20 ~ + 75	°C
Storage Temperature	T_{stg}	− 55 ~ +150	°C

* With respect to V_{SS} (SYSTEM GND)

(NOTE) Permanent LSI damage may occur if maximum rating are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ RECOMMENDED OPERATING CONDITION

Item	Symbol	min	typ	max	Unit
Supply Voltage	V_{CC}*	4.75	5.0	5.25	V
Input Voltage	V_{IL}*	−0.3	–	0.8	V
	V_{IH}*	2.0	–	V_{CC}	V
Operating Temperature	T_{opr}	−20	25	75	°C

* With respect to V_{SS} (SYSTEM GND)

■ ELECTRICAL CHARACTERISTICS

● DC CHARACTERISTICS (V_{CC} = 5V ± 5%, V_{SS} = 0V, Ta = −20~+75°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ*	max	Unit
Input "High" Voltage	Logic**	V_{IH}		2.0	–	V_{CC}	V
Input "Low" Voltage	Logic**	V_{IL}		− 0.3	—	0.8	V
Clock Input "High" Voltage	ϕ_1, ϕ_2	V_{IHC}		V_{CC} − 0.6	–	V_{CC} + 0.3	V
Clock Input "Low" Voltage	ϕ_1, ϕ_2	V_{ILC}		−0.3	–	0.4	V
Output "High" Voltage	D_0~D_7	V_{OH}	I_{OH} = −205μA	2.4	–	–	V
	A_0~A_{15}, R/$\overline{W}$ VMA		I_{OH} = −145μA	2.4	–	–	V
	BA		I_{OH} = −100μA	2.4	–	–	V
Output "Low" Voltage		V_{OL}	I_{OL} = 1.6mA	–	–	0.4	V
Input Leakage Current	Logic***	I_{in}	V_{in} = 0~5.25V, All other pins are connected to GND	-2.5	–	2.5	μA
	ϕ_1, ϕ_2			-100	–	100	μA
Three-State (Off-state) Input Current	D_0~D_7	I_{TSI}	V_{in} = 0.4 ~ 2.4V	-10	–	10	μA
	A_0~A_{15}, R/$\overline{W}$			-100	–	100	μA
Power Dissipation		P_D		–	0.5	1.0	W
Input Capacitance	Logic***	C_{in}	V_{in} = 0V, Ta = 25°C, f = 1 MHz	–	6.5	10	pF
	D_0~D_7			–	10	12.5	pF
	ϕ_1			–	25	35	pF
	ϕ_2			–	45	70	pF
Output Capacitance	A_0~A_{15}, R/$\overline{W}$ VMA, BA	C_{out}	V_{in} = 0V, Ta = 25°C, f = 1 MHz	–	–	12	pF

* Ta = 25°C, V_{CC} = 5V
** All inputs except ϕ_1 and ϕ_2
*** All inputs except ϕ_1, ϕ_2 and D_0~D_7

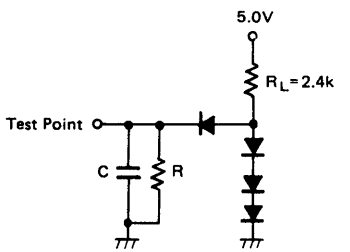
• AC CHARACTERISTICS (V_{CC} = 5V ± 5%, V_{SS} = 0V, Ta = -20~+75°C, unless otherwise noted.)

1. TIMING CHARACTERISTICS OF CLOCK PULSE ϕ_1 and ϕ_2

Item		Symbol	Test Condition	HD6800			HD68A00			HD68B00			Unit
				min	typ	max	min	typ	max	min	typ	max	
Frequency of Operation		f		0.1	–	1.0	0.1	–	1.5	0.1	–	2.0	MHz
Cycle Time		t_{cyc}	Fig. 10	1.000	–	10	0.666	–	10	0.500	–	10	μs
Clock Pulse Width	ϕ_1, ϕ_2	PW_{CH1}, PW_{CH2}	Fig. 10	400	–	4,500	230	–	4,500	180	–	4,500	ns
Rise and Fall Times	ϕ_1, ϕ_2	t_r, t_f	Fig. 10	–	–	100	–	–	100	–	–	100	ns
Delay Time (Clock Internal)		t_d	Fig. 10	0	–	4,500	0	–	4,500	0	–	4,500	ns
Clock "High" Level Time		t_{UT}	Fig. 10	900	–	–	600	–	–	440	–	–	ns

2. READ/WRITE CHARACTERISTICS

Item		Symbol	Test Condition	HD6800			HD68A00			HD68B00			Unit
				min	typ	max	min	typ	max	min	typ	max	
Address Delay Time	C=90pF	t_{AD1}	Fig. 11, Fig. 12	–	–	270	–	–	180	–	–	150	ns
	C=30pF	t_{AD2}	Fig. 11, Fig. 12	–	–	250	–	–	165	–	–	135	ns
Data Setup Time (Read)		t_{DSR}	Fig. 11	100	–	–	60	–	–	40	–	–	ns
Peripheral Read Access Time $t_{acc} = t_{UT} - (t_{AD} + t_{DSR})$		t_{acc}	Fig. 11	–	–	530	–	–	360	–	–	250	ns
Input Data Hold Time		t_H	Fig. 11	10	–	–	10	–	–	10	–	–	ns
Output Data Hold Time		t_H	Fig. 12	20	–	–	20	–	–	20	–	–	ns
Address Hold Time (Address, R/$\overline{W}$, VMA)		t_{AH}	Fig. 11, Fig. 12	10	–	–	10	–	–	10	–	–	ns
Enable "High" Time for DBE Input		t_{EH}	Fig. 12	450	–	–	280	–	–	220	–	–	ns
Data Delay Time (Write)		t_{DDW}	Fig. 12	–	–	225	–	–	200	–	–	160	ns
Data Bus Enable Down Time (During ϕ_1 Up Time)		$t_{\overline{DBE}}$	Fig. 12	150	–	–	120	–	–	75	–	–	ns
Data Bus Enable Delay Time		t_{DBED}	Fig. 12	300	–	–	250	–	–	180	–	–	ns
Data Bus Enable Rise and Fall Times		t_{DBEr} t_{DBEf}	Fig. 12	–	–	25	–	–	25	–	–	25	ns
Processor Control Setup Time		t_{PCS}		200	–	–	140	–	–	110	–	–	ns
Processor Control Rise and Fall Times		t_{PCr} t_{PCf}		–	–	100	–	–	100	–	–	100	ns
Bus Available Delay Time (BA)		t_{BA}		–	–	250	–	–	165	–	–	135	ns
Three-State Delay Time		t_{TSD}		–	–	270	–	–	270	–	–	220	ns



C = 130pF for D_0~D_7
= 90pF for A_0~A_{15}, R/$\overline{W}$, and VMA
= 30pF for BA
R = 11kΩ for D_0~D_7
= 16kΩ for A_0~A_{15}, R/$\overline{W}$ and VMA
= 24kΩ for BA
C includes Stray Capacitance.
All diodes are 1S2074 Ⓗ or equivalent.

Figure 1 Bus Timing Test Load

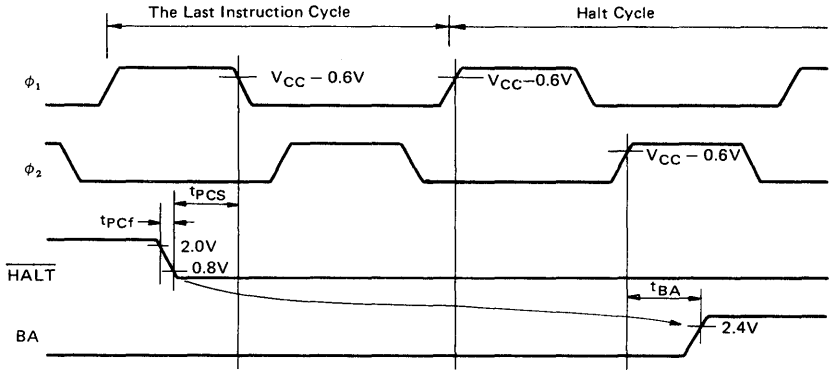


Figure 2 Timing of $\overline{\text{HALT}}$ and BA

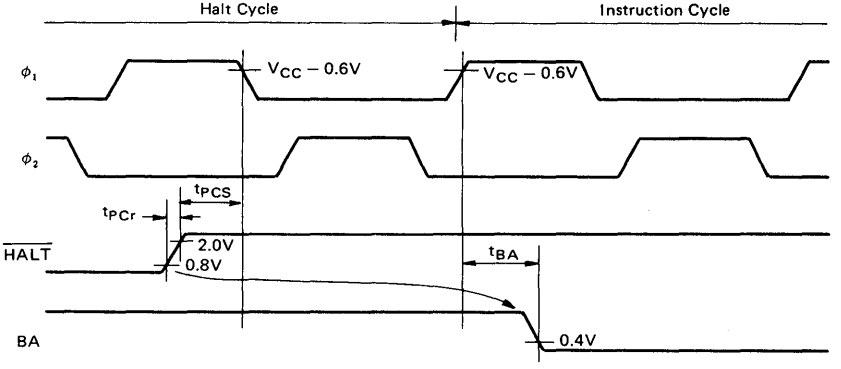


Figure 3 Timing of $\overline{\text{HALT}}$ and BA

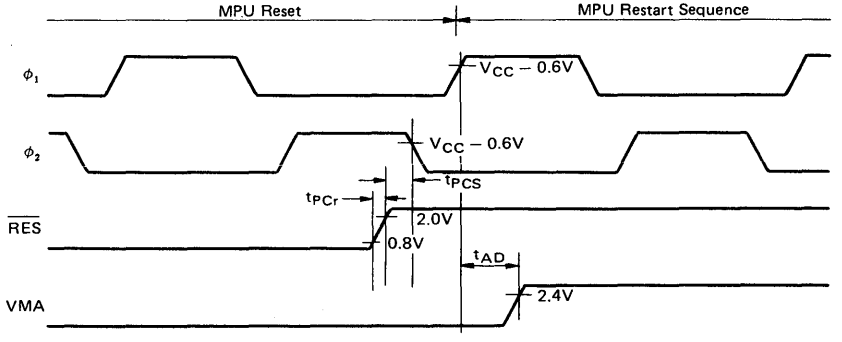


Figure 4 $\overline{\text{RES}}$ and MPU Restart Sequence

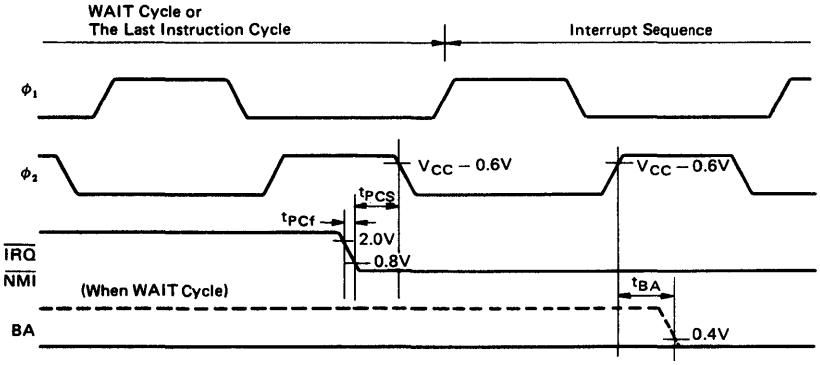


Figure 5 $\overline{IRQ}$ and $\overline{NMI}$ Interrupt Timing

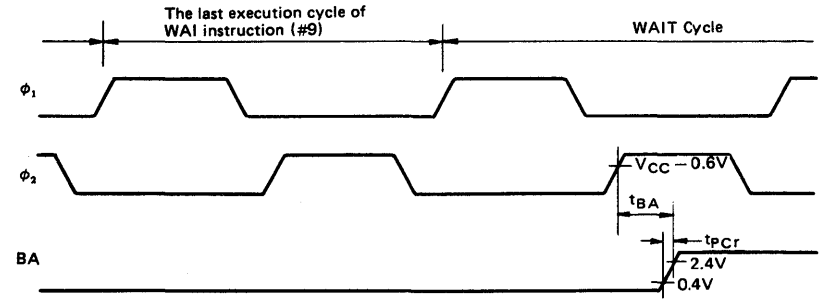


Figure 6 WAI Instruction and BA Timing

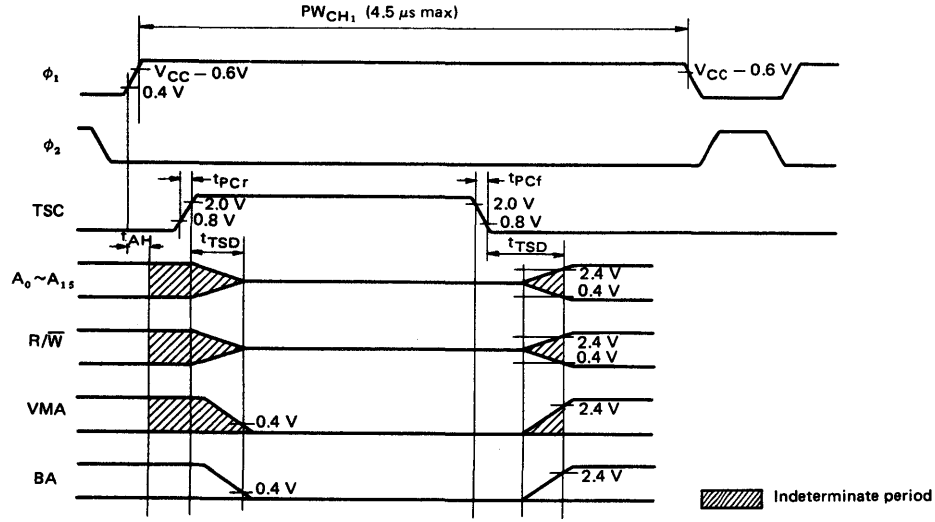


Figure 7 TSC Input and MPU Output

■ MPU REGISTERS

The MPU provides several registers in Fig. 8, which is available for use by the programmer.

Each register is described below.

- **Program Counter (PC)**

The program counter is a two byte (16-bit) register that points to the current program address.

- **Stack Pointer (SP)**

The stack pointer is a two byte register that contains the address of the next available location in an external push-down/pop-up stack. This stack is normally a random access Read/Write memory that may have any location (address) that is convenient. In those applications that require storage of information in the stack when power is lost, the stack must be non-volatile.

- **Index Register (IX)**

The index register is a two byte register that is used to store data or a sixteen bit memory address for the Indexed mode of memory addressing.

- **Accumulators (ACCA, ACCB)**

The MPU contains two 8-bit accumulators that are used to hold operands and results from an arithmetic logic unit (ALU).

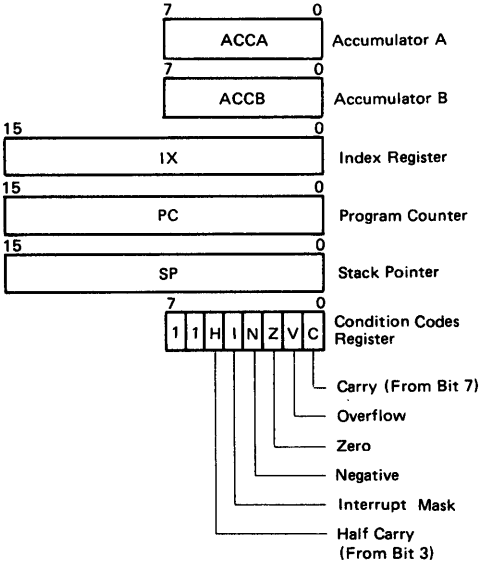


Figure 8 Programming Model of the Microprocessing Unit

- **Condition Code Register (CCR)**

The condition code register indicates the results of an Arithmetic Logic Unit operation: Negative (N), Zero (Z), Overflow (V), Carry from bit 7 (C), and half carry from bit 3(H). These bits of the Condition Code Register are used as testable conditions for the conditional branch instructions. Bit 4 is the interrupt mask bit (I). The unused bits of the Condition Code Register (b6 and b7) are "1". The detail block diagram of the microprossing unit is shown in Fig. 9.

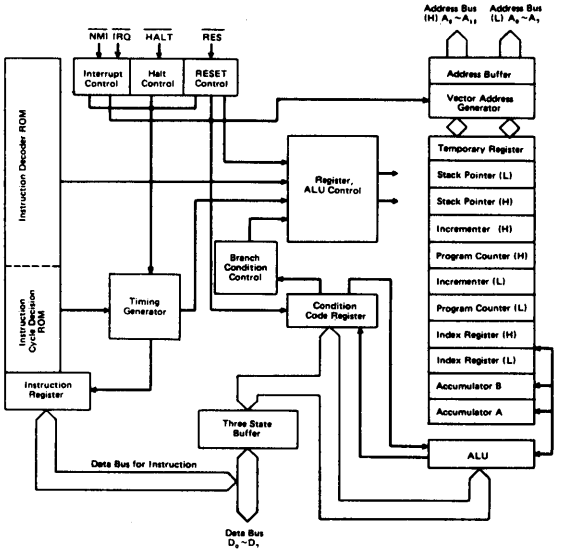


Figure 9 Internal Block Diagram of MPU

■ MPU SIGNAL DESCRIPTION

Proper operations of the MPU requires that certain control and timing signals (Fig. 9) be provided to accomplish specific functions. The functions of pins are explained in this section.

- **Clock (ϕ_2, ϕ_2)**

Two pins are used to provide the clock signals. A two-phase non-overlapping clock is provided as shown in Fig. 10.

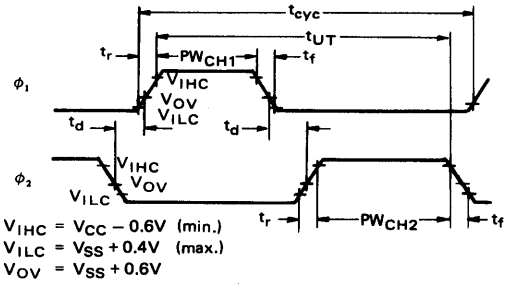


Figure 10 Clock Timing Waveform

- **Address Bus (A_0~A_{15})**

Sixteen pins are used for the address bus. The outputs are three-state bus drivers capable of driving one standard TTL load and 90pF. When the output is turned off, it is essentially an open circuit. This permits the MPU to be used in DMA applications. Putting TSC in its high state forces the Address bus to go into the three-state mode.

- **Data Bus (D_0~D_7)**

Eight pins are used for the data bus. It is bidirectional, transferring data to and from the memory and peripheral devices. It also has three-state output buffers capable of driving one standard TTL load and 130pF. Data Bus is placed in the three-state mode when DBE is "Low."

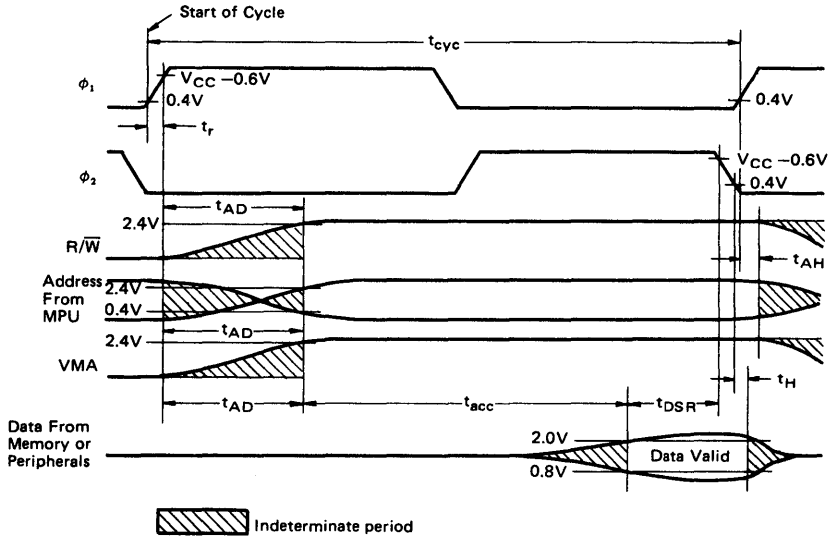


Figure 11 Read from Memory or Peripherals

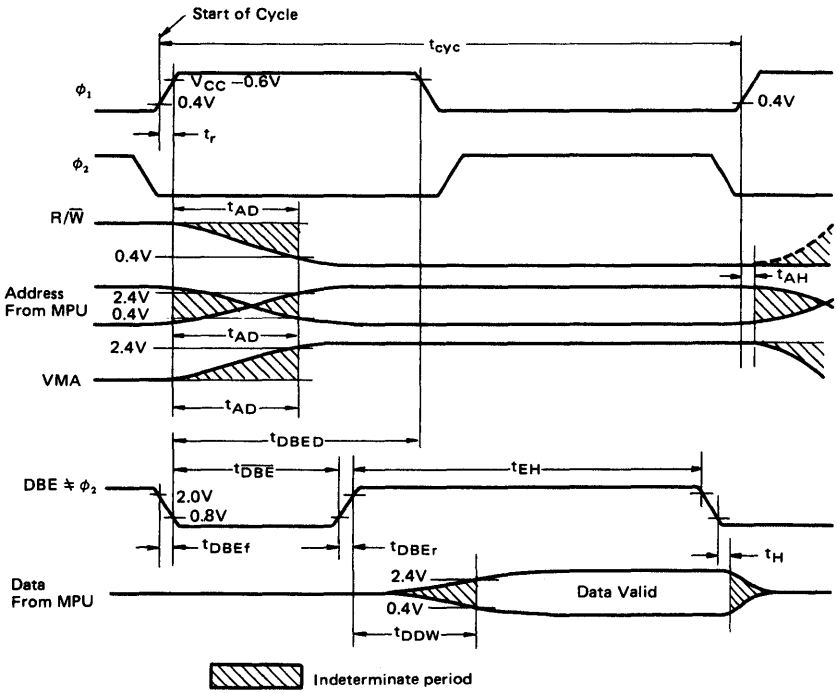


Figure 12 Write to Memory or Peripherals

- **Data Bus Enable (DBE)**

This input is the three-state control signal for the MPU data bus and will enable the bus drivers when in the "High" state; will make the bus driver off when in the "Low" state. This input is TTL compatible; however in normal operation, it would be driven by ϕ_2 clock. During an MPU read cycle, the data bus drivers will be disabled internally. When it is desired that another device control the data bus such as in Direct Memory Access (DMA) applications, DBE should be held "Low."

If additional data setup or hold time is required on an MPU write, the DBE down time can be decreased as shown in Fig. 13 (DBE $\downarrow \phi_2$). The minimum down time for DBE is $t_{\overline{DBE}}$ as shown and must occur within ϕ_1 up time. As for the characteristical values in Fig. 12, refer to the table of electrical characteristics.

- **Bus Available (BA)**

The BA signal will normally be in the "Low" state. When activated, it will go to the "High" state indicating that the microprocessor has stopped and that the address bus is available. This will occur if the $\overline{HALT}$ line is in the "Low" state or the processor is in the WAIT state as a result of the execution of a WAIT instruction. At such time, all three-state output drivers will go to their off state and other outputs to their normally inactive level. The processor is removed from the WAIT state by the occurrence of a maskable (mask bit I = 0) or nonmaskable interrupt. This output is capable of driving one standard TTL load and 30pF. If TSC is in the "High" state, Bus Available will be "Low".

- **Read/Write (R/$\overline{W}$)**

This TTL compatible output signals the peripherals and memory devices whether the MPU is in a Read ("High") or Write ("Low") state. The normal standby state of this signal is Read ("High"). Three-State Control going "High" will turn R/$\overline{W}$ to the off (high impedance) state. Also, when the processor is halted, it will be in the off state. This output is capable of driving one standard TTL load and 90pF.

- **Reset ($\overline{RES}$)**

The $\overline{RES}$ input is used to reset and start the MPU from a power down condition resulting from a power failure or initial start-up of the processor. This input can also be used to reinitialize the machine at any time after start-up.

If a "High" level is detected in this input, this will signal the MPU to begin the reset sequence. During the reset sequence, the contents of the last two locations (FFFE, FFFF) in memory will be loaded into the Program Counter to point to the beginning of the reset routine. During the reset routine, the interrupt mask bit is set and must be cleared under program control before the MPU can be interrupted by $\overline{IRQ}$. While $\overline{RES}$ is "Low" (assuming a minimum of 8 clock cycles have occured) the MPU output signals will be in the following states; VMA = "Low", BA = "Low", Data Bus = high impedance, R/$\overline{W}$ = "High" (read state), and the Address Bus will contain the reset address FFFE. Fig. 13 illustrates a power up sequence using the Reset control line. After the power supply reaches 4.75V, a minimum of eight clock cycles are required for the processor to stabilize in preparation for restarting. During these eight cycles, VMA will be in an indeterminate state so any devices that are enabled by VMA which could accept a false write during this time (such as a battery-backed RAM) must be disabled until VMA is forced "Low" after eight cycles. $\overline{RES}$ can go "High" asynchronously with the system clock any time after the eighth cycle.

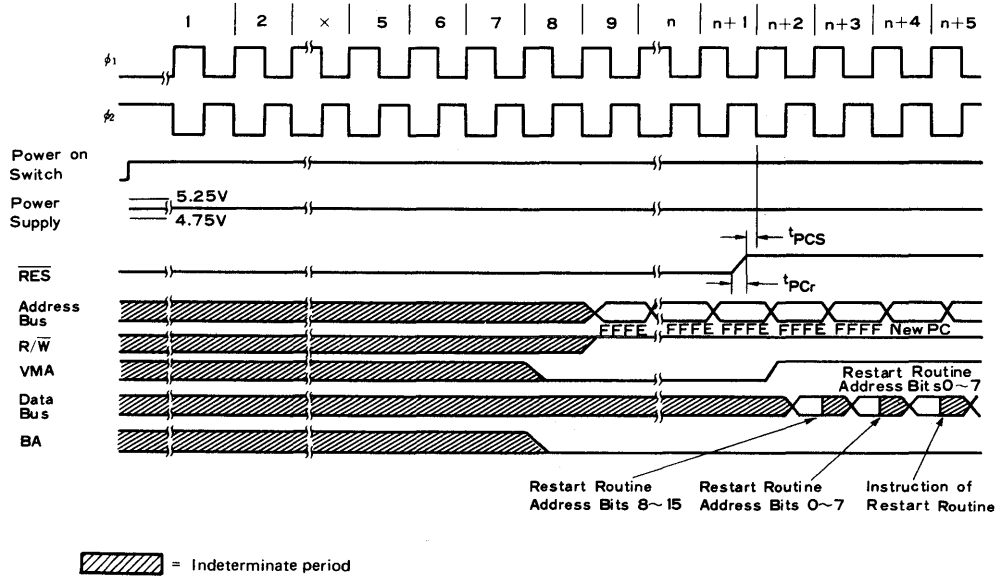


Figure 13 $\overline{RES}$ Timing

The Reset control line may also be used to reinitialize the MPU system at any time during its operation. This is accomplished by pulsing $\overline{RES}$ "Low" for the duration of a minimum of three complete ϕ_2 cycles. The $\overline{RES}$ pulse can be completely asynchronous with the MPU system clock and will be recognized during ϕ_2 if setup time t_{PCS} is met.

- **Interrupt Request ($\overline{IRQ}$)**

This level sensitive input requests that an interrupt sequence be generated within the machine. The processor will wait until it completes the current instruction that is being executed before it recognizes the request. If the interrupt mask bit in the Condition Code Register is not set, the machine will begin an interrupt sequence. The Index Register, Program Counter, Accumulators, and Condition Code Register are stored away on the stack.

Next the MPU will respond to the interrupt request by setting the interrupt mask bit "1" so that no further interrupts may occur. At the end of the cycle, a 16-bit address will be loaded that points to a vectoring address which is located in memory locations FFF8 and FFF9. An address loaded at these locations causes the MPU to branch to an interrupt routine in memory. Interrupt timing is shown in Fig. 14.

The $\overline{HALT}$ line must be in the "High" state for interrupts to be serviced. Interrupts will be latched internally while $\overline{HALT}$ is "Low". The $\overline{IRQ}$ has a high impedance pullup device internal to the chip; however a 3kΩ external resistor to V_{CC} should be used for wire-OR and optimum control of interrupts.

- **Non-Maskable Interrupt ($\overline{NMI}$) and Wait for Interrupt (WAI)**

The MPU is capable of handling two types of interrupts: maskable ($\overline{IRQ}$) as described earlier, and non-maskable ($\overline{NMI}$). $\overline{IRQ}$ is maskable by the interrupt mask in the Condition Code Register while $\overline{NMI}$ is not maskable. The handling of these interrupts by the MPU is the same except that each has its own vector address. The behavior of the MPU when interrupted is shown in Fig. 14 which details the MPU response to an interrupt while the MPU is executing the control program. The interrupt shown could be either $\overline{IRQ}$ or $\overline{NMI}$ and can be asynchronous with respect to ϕ_2. The interrupt is shown going "Low" at time t_{PCS} in cycle #1 which precedes the first cycle of an instruction (OP code fetch). This instruction is not executed but instead the Program Counter (PC), Index Register (IX), Accumulators (ACCX), and the Condition Code Register (CCR) are pushed onto the stack.

The Interrupt Mask bit is set to prevent further interrupts. The address of the interrupt service routine is then fetched from FFFC, FFFD for an $\overline{NMI}$ interrupt and from FFF8, FFF9 for an $\overline{IRQ}$ interrupt. Upon completion of the interrupt service routine, the execution of RTI will pull the PC, IX, ACCX, and CCR off of the stack; the Interrupt Mask bit is restored to its condition prior to interrupts. Fig. 15 is a similar interrupt sequence, except in this case, a WAIT instruction has been executed in preparation for the interrupt. This technique speeds up the MPU's response to the interrupt because the stacking of

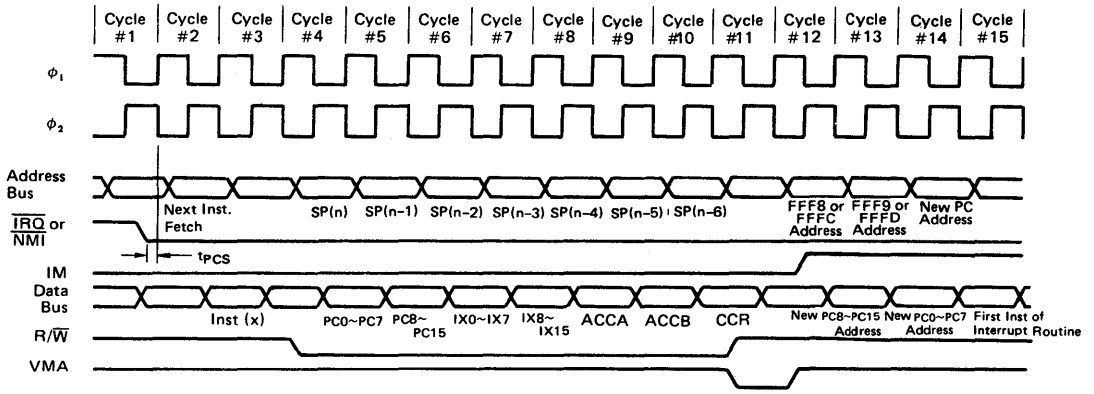
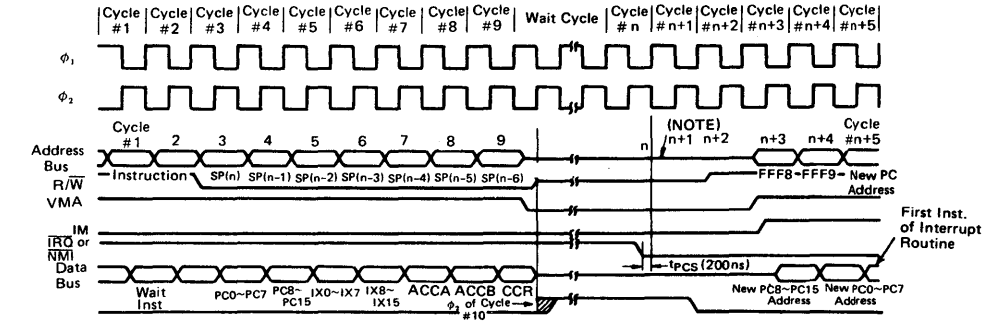


Figure 14 Interrupt Timing



(NOTE) Midrange waveform indicates high impedance state.

Figure 15 WAI Instruction Timing

the PC, IX, ACCX, and the CCR is already done.

While the MPU is waiting for the interrupt, Bus Available will go "High" indicating the following states of the control lines: VMA is "Low", and the Address Bus, R/$\overline{W}$ and Data Bus are all in the high impedance state. After the interrupt occurs, it is serviced as previously described.

Table 1 Memory Map for Interrupt Vectors

Vector		Description
MS	LS	
FFFE	FFFF	Restart
FFFC	FFFD	Non-maskable Interrupt
FFFA	FFFB	Software Interrupt
FFF8	FFF9	Interrupt Request

Refer to Figure 18 for program flow for Interrupts.

- **Three State Control (TSC)**

When the Three State Control (TSC) line is "High" level, the Address Bus and the R/$\overline{W}$ line are placed in a high impedance State. VMA and BA are forced "Low" when TSC = "High" to prevent false reads or writes on any device enabled by VMA. It is necessary to delay program execution while TSC is held "High". This is done by insuring that no transitions of ϕ_1 (or ϕ_2) occur during this period. (Logic levels of the clocks are irrelevant so long as they do not change.)

Since the MPU is a dynamic device, the ϕ_1 clock can be stopped for a maximum time PW_{CH1} without destroying data within the MPU. TSC then can be used in a short Direct Memory Access (DMA) application.

Fig. 16 shows the effect of TSC on the MPU. The Address Bus and R/$\overline{W}$ line will reach the high impedance state at t_{TSD} (three-state delay), with VMA being forced "Low". In this example, the Data Bus is also in the high impedance state while ϕ_2 is being held "Low" since DBE=ϕ_2. At this point in time, a DMA transfer could occur on cycles #3 and #4. When TSC is returned "Low," the MPU address and R/$\overline{W}$ lines return to the bus. Because it is too late in cycle #5 to access memory, this cycle is dead and used for synchronization. Program execution resumes in cycle #6.

- **Valid Memory Address (VMA)**

This output indicates to peripheral devices that there is a valid address on the address bus. In normal operation, this signal should be utilized for enabling peripheral interfaces such as the PIA and ACIA. This signal is not three-state. One standard TTL load and 90pF may be directly driven by this active "High" signal.

- **Halt ($\overline{HALT}$)**

When this input is in the "Low" state, all activity in the machine will be halted. This input is level sensitive.

The $\overline{HALT}$ line provides an input to the MPU to allow control or program execution by an outside source. If $\overline{HALT}$ is "High", the MPU will execute the instructions; if it is "Low", the MPU will go to a halted or idle mode. A response signal, Bus Available (BA) provides an indication of the current MPU status. When BA is "Low", the MPU is in the process of executing the control program; if BA is "High", the MPU has halted and all internal activity has stopped.

When BA is "High", the Address Bus, Data Bus, and R/$\overline{W}$ line will be in a high impedance state, effectively removing the MPU from the system bus. VMA is forced "Low" so that the floating system bus will not activate any device on the bus that is enabled by VMA.

While the MPU is halted, all program activity is stopped, and if either an $\overline{NMI}$ or $\overline{IRQ}$ interrupt occurs, it will be latched into the MPU and acted on as soon as the MPU is taken out of the halted mode. If a $\overline{RES}$ command occurs while the MPU is halted, the following states occur: VMA = "Low", BA = "Low", Data Bus = high impedance, R/$\overline{W}$ = "High" (read state), and the Address Bus will contain address FFFE as long as $\overline{RES}$ is "Low". As soon as the $\overline{HALT}$ line goes "High", the MPU will go to locations FFFE and FFFF for the address of the reset routine.

Fig. 18 shows the timing relationships involved when halting the MPU. The instruction illustrated is a one byte, 2 cycle instruction such as CLRA. When $\overline{HALT}$ goes "Low", the MPU will halt after completing execution of the current instruction. The transition of $\overline{HALT}$ must occur t_{PCS} before the trailing edge of ϕ_1 of the last cycle of an instruction (point A of Fig. 18). $\overline{HALT}$ must not go "Low" any time later than the minimum t_{PCS} specified.

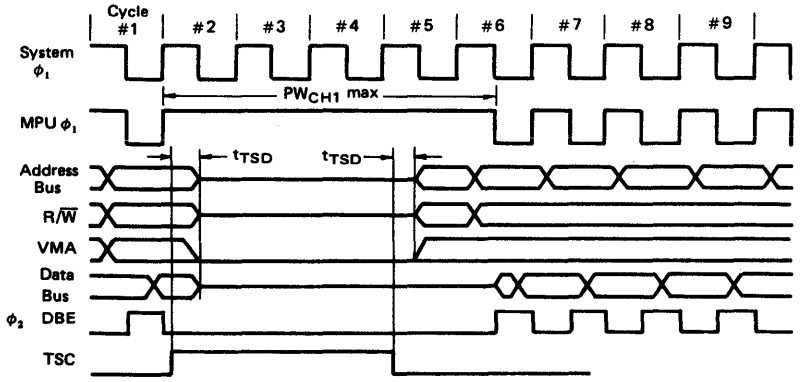


Figure 16 TSC Control Timing

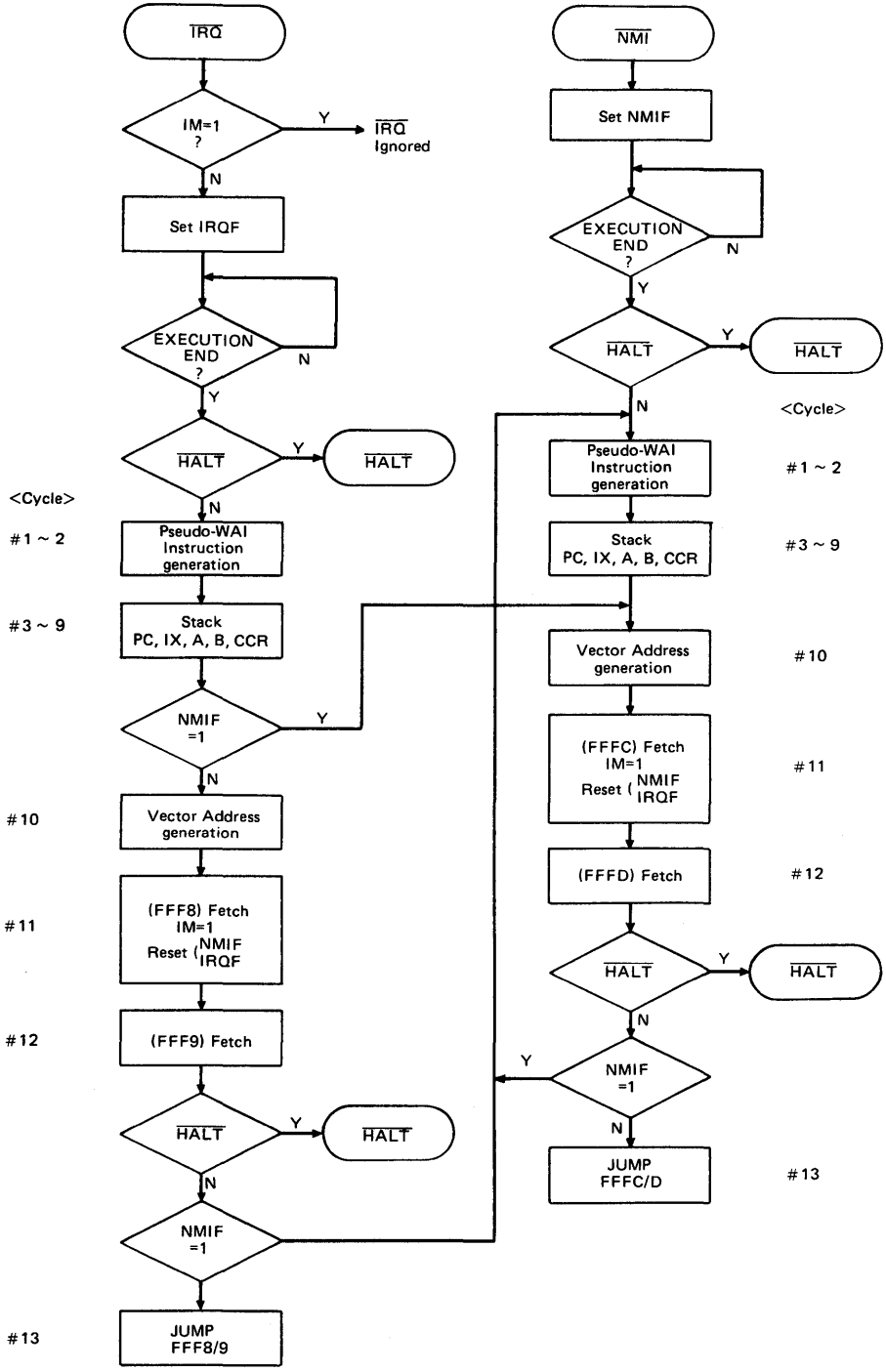
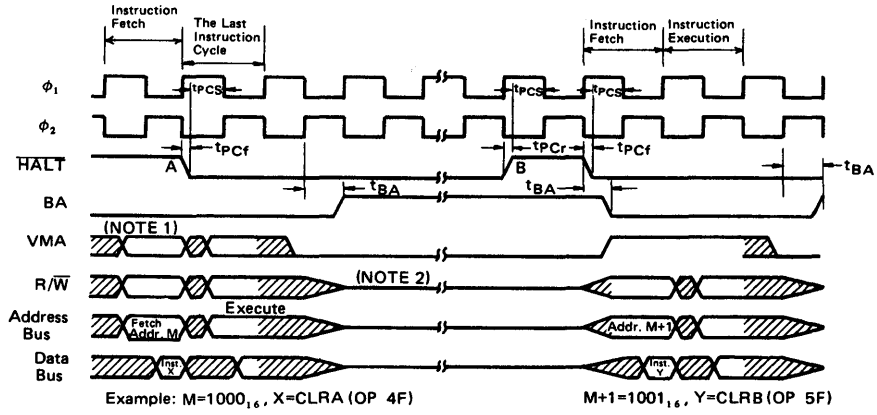


Figure 17 MPU Interrupt Flow Chart



(NOTE) 1. Oblique lines indicate indeterminate range of data.
2. Midrange waveform indicates high impedance state.

Figure 18 $\overline{\text{HALT}}$ and Single Instruction Execution for System Dubug

Table 2 Operation States of MPU and Signal Outputs (Except the Execution of Instruction)

Signals	Halt state	Reset state	Halt and Reset state	WAI state	TSC state
BA	"H"	"L"	"L"	"H"	"L"
VMA	"L"	"L"	"L"	"L"	"L"
R/$\overline{W}$	"T"	"H"	"H"	"T"	"T"
$A_0 \sim A_{15}$	"T"	$(FFFE)_{16}$	$(FFFE)_{16}$	"T"	"T"
$D_0 \sim D_7$	"T"	"T"	"T"	"T"	–

"T" indicates high impedance state.

The fetch of the OP code by the MPU is the first cycle of the instruction. If $\overline{\text{HALT}}$ had not been "Low" at Point A but went "Low" during ϕ_2 of the cycle, the MPU would have halted after completion of the following instruciton. BA will go "High" by time t_{BA} (bus available delay time) after the last instruction cycle. At this point in time, VMA is "Low" and R/$\overline{W}$, Address Bus, and the Data Bus are in the high impedance state.

To debug programs it is advantageous to step through programs instruction by instruction. To do this, $\overline{\text{HALT}}$ must be brought "High" for one MPU cycle and then returned "Low" as shown at point B of Fig. 18. Again, the transitions of $\overline{\text{HALT}}$ must occur t_{PCS} before the trailing edge of ϕ_1. BA will go "Low" at t_{BA} after the leading edge of the next ϕ_1, indicating that the Address Bus, Data Bus, VMA and R/$\overline{W}$ lines are back on the bus. A single byte, 2 cycle instruction such as LSR is used for this example also. During the first cycle, the instruction Y is fetched from address M+1. BA returns "High" at t_{BA} on the last cycle of the instruction indicating the MPU is off the bus, if instruction Y had been three cycles, the width of the BA "Low" time would have been increased by one cycle.

Table 2 shows the relation between the state of MPU and signal outputs.

■ MPU INSTRUCTION SET

This Section will provide a brief introduction and discuss their use in developing HD6800 MPU control programs. The HD6800 MPU has a set of 72 different executable source instructions. Included are binary and decimal arithmetic, logical, shift, rotate, load, store, conditional or unconditional branch, interrupt and stack manipulation instructions.

Each of the 72 executable instructions of the source language assembles into 1 to 3 bytes of machine code. The number of bytes depends on the particular instruction and on the addressing mode. (The addressing modes which are available for use with the various executive instructions are discussed later.)

The coding of the first (or only) byte corresponding to an executable instruction is sufficient to identify the instruction and the addressing mode. The hexadecimal equivalents of the binary codes, which result from the translation of the 72 instructions in all valid modes of addressing, are shown in Table 3. There are 197 valid machine codes, 59 of the 256 possible codes being unassigned.

When an instruction translates into two or three bytes of code, the second byte, or second and third bytes contain(s) an operand, an address, or information from which an address is obtained during execution.

Microprocessor instructions are often devided into three general classifications; (1) memory reference, so called because they operate on specific memory locations; (2) operating instructions that function without needing a memory reference; (3) I/O instructions for transferring data between the microprocessor and peripheral devices.

In many instances, the HD6800 MPU performs the same operation on both its internal accumulators and the external memory locations. In addition, the HD6800 MPU allow the MPU to treat peripheral devices exactly like other memory locations, hence, no I/O instructions as such are required. Because of these features, other classifications are more suitable for introducing the HD6800's instruction set: (1) Accumulator and memory operations; (2) Program control operations; (3) Condition Code Register operations.

For Accumulator and Memory Operations, refer to Table 4.

Table 3 Hexadecimal Values of Machine Codes

MSB \ LSB	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	*	NOP (IMP)	*	*	*	*	TAP (IMP)	TPA (IMP)	INX (IMP)	DEX (IMP)	CLV (IMP)	SEV (IMP)	CLC (IMP)	SEC (IMP)	CLI (IMP)	SEI (IMP)
1	SBA (A, B)	CBA (A, B)	*	*	*	*	TAB (IMP)	TBA (IMP)	*	DAA (IMP)	*	ABA (IMP)	*	*	*	*
2	BRA (REL)	*	BHI (REL)	BLS (REL)	BCC (REL)	BCS (REL)	BNE (REL)	BEQ (REL)	BVC (REL)	BVS (REL)	BPL (REL)	BMI (REL)	BGE (REL)	BLT (REL)	BGT (REL)	BLE (REL)
3	TSX (IMP)	INS (IMP)	PUL (A)	PUL (B)	DES (IMP)	TXS (IMP)	PSH (A)	PSH (B)	*	RTS (IMP)	*	RTI (IMP)	*	*	WAI (IMP)	SWI (IMP)
4	NEG (A)	*	*	COM (A)	LSR (A)	*	ROR (A)	ASR (A)	ASL (A)	ROL (A)	DEC (A)	*	INC (A)	TST (A)	*	CLR (A)
5	NEG (B)	*	*	COM (B)	LSR (B)	*	ROR (B)	ASR (B)	ASL (B)	ROL (B)	DEC (B)	*	INC (B)	TST (B)	*	CLR (B)
6	NEG (IND)	*	*	COM (IND)	LSR (IND)	*	ROR (IND)	ASR (IND)	ASL (IND)	ROL (IND)	DEC (IND)	*	INC (IND)	TST (IND)	JMP (IND)	CLR (IND)
7	NEG (EXT)	*	*	COM (EXT)	LSR (EXT)	*	ROR (EXT)	ASR (EXT)	ASL (EXT)	ROL (EXT)	DEC (EXT)	*	INC (EXT)	TST (EXT)	JMP (EXT)	CLR (EXT)
8	SUB (IMM)(A)	CMP (IMM)(A)	SBC (IMM)(A)	*	AND (IMM)(A)	BIT (IMM)(A)	LDA (IMM)(A)	*	EOR (IMM)(A)	ADC (IMM)(A)	ORA (IMM)(A)	ADD (IMM)(A)	CPX (IMM)(A)	BSR (REL)	LDS (IMM)	*
9	SUB (DIR)(A)	CMP (DIR)(A)	SBC (DIR)(A)	*	AND (DIR)(A)	BIT (DIR)(A)	LDA (DIR)(A)	STA (DIR)(A)	EOR (DIR)(A)	ADC (DIR)(A)	ORA (DIR)(A)	ADD (DIR)(A)	CPX (DIR)(A)	*	LDS (DIR)	STS (DIR)
A	SUB (IND)(A)	CMP (IND)(A)	SBC (IND)(A)	*	AND (IND)(A)	BIT (IND)(A)	LDA (IND)(A)	STA (IND)(A)	EOR (IND)(A)	ADC (IND)(A)	ORA (IND)(A)	ADD (IND)(A)	CPX (IND)(A)	JSR (IND)	LDS (IND)	STS (IND)
B	SUB (EXT)(A)	CMP (EXT)(A)	SBC (EXT)(A)	*	AND (EXT)(A)	BIT (EXT)(A)	LDA (EXT)(A)	STA (EXT)(A)	EOR (EXT)(A)	ADC (EXT)(A)	ORA (EXT)(A)	ADD (EXT)(A)	CPX (EXT)(A)	JSR (EXT)	LDS (EXT)	STS (EXT)
C	SUB (IMM)(B)	CMP (IMM)(B)	SBC (IMM)(B)	*	AND (IMM)(B)	BIT (IMM)(B)	LDA (IMM)(B)	*	EOR (IMM)(B)	ADC (IMM)(B)	ORA (IMM)(B)	ADD (IMM)(B)	*	*	LDX (IMM)	*
D	SUB (DIR)(B)	CMP (DIR)(B)	SBC (DIR)(B)	*	AND (DIR)(B)	BIT (DIR)(B)	LDA (DIR)(B)	STA (DIR)(B)	EOR (DIR)(B)	ADC (DIR)(B)	ORA (DIR)(B)	ADD (DIR)(B)	*	*	LDX (DIR)(B)	STX (DIR)(B)
E	SUB (IND)(B)	CMP (IND)(B)	SBC (IND)(B)	*	AND (IND)(B)	BIT (IND)(B)	LDA (IND)(B)	STA (IND)(B)	EOR (IND)(B)	ADC (IND)(B)	ORA (IND)(B)	ADD (IND)(B)	*	*	LDX (IND)	STX (IND)
F	SUB (EXT)(B)	CMP (EXT)(B)	SBC (EXT)(B)	*	AND (EXT)(B)	BIT (EXT)(B)	LDA (EXT)(B)	STA (EXT)(B)	EOR (EXT)(B)	ADC (EXT)(B)	ORA (EXT)(B)	ADD (EXT)(B)	*	*	LDX (EXT)	STX (EXT)

DIR = Direct Addressing Mode
EXT = Extended Addressing Mode
IMM = Immediate Addressing Mode
IND = Index Addressing Mode
IMP = Implied Addressing Mode
REL = Relative Addressing Mode
A = Accumulator A
B = Accumulator B

Table 4 Accumulator and Memory Operations

Operation	Mnemonic	IMMED OP	IMMED ~	IMMED #	DIRECT OP	DIRECT ~	DIRECT #	INDEX OP	INDEX ~	INDEX #	EXTND OP	EXTND ~	EXTND #	IMPLIED OP	IMPLIED ~	IMPLIED #	Boolean/ Arithmetic Operation	5 H	4 I	3 N	2 Z	1 V	0 C
Add	ADDA	8B	2	2	9B	3	2	AB	5	2	BB	4	3				A + M → A	‡	‡	‡	‡	‡	‡
	ADDB	CB	2	2	DB	3	2	EB	5	2	FB	4	3				B + M → B	‡	‡	‡	‡	‡	‡
Add Acmltrs	ABA													1B	2	1	A + B → A	‡	‡	‡	‡	‡	‡
Add with Carry	ADCA	89	2	2	99	3	2	A9	5	2	B9	4	3				A + M + C → A	‡	‡	‡	‡	‡	‡
	ADCB	C9	2	2	D9	3	2	E9	5	2	F9	4	3				B + M + C → B	‡	‡	‡	‡	‡	‡
And	ANDA	84	2	2	94	3	2	A4	5	2	B4	4	3				A • M → A	•	•	‡	‡	R	•
	ANDB	C4	2	2	D4	3	2	E4	5	2	F4	4	3				B • M → B	•	•	‡	‡	R	•
Bit Test	BITA	85	2	2	95	3	2	A5	5	2	B5	4	3				A • M	•	•	‡	‡	R	•
	BITB	C5	2	2	D5	3	2	E5	5	2	F5	4	3				B • M	•	•	‡	‡	R	•
Clear	CLR							6F	7	2	7F	6	3				00 → M	•	•	R	S	R	R
	CLRA													4F	2	1	00 → A	•	•	R	S	R	R
	CLRB													5F	2	1	00 → B	•	•	R	S	R	R
Compare	CMPA	81	2	2	91	3	2	A1	5	2	B1	4	3				A − M	•	•	‡	‡	‡	‡
	CMPB	C1	2	2	D1	3	2	E1	5	2	F1	4	3				B − M	•	•	‡	‡	‡	‡
Compare Acmltrs	CBA													11	2	1	A − B	•	•	‡	‡	‡	‡
Complement, 1's	COM							63	7	2	73	6	3				M̄ → M	•	•	‡	‡	R	S
	COMA													43	2	1	Ā → A	•	•	‡	‡	R	S
	COMB													53	2	1	B̄ → B	•	•	‡	‡	R	S
Complement, 2's (Negate)	NEG							60	7	2	70	6	3				00 − M → M	•	•	‡	‡	①	②
	NEGA													40	2	1	00 − A → A	•	•	‡	‡	①	②
	NEGB													50	2	1	00 − B → B	•	•	‡	‡	①	②
Decimal Adjust, A	DAA													19	2	1	Converts Binary Add of BCD Characters into BCD Format	•	•	‡	‡	‡	③
Decrement	DEC							6A	7	2	7A	6	3				M − 1 → M	•	•	‡	‡	④	•
	DECA													4A	2	1	A − 1 → A	•	•	‡	‡	④	•
	DECB													5A	2	1	B − 1 → B	•	•	‡	‡	④	•
Exclusive OR	EORA	88	2	2	98	3	2	A8	5	2	B8	4	3				A ⊕ M → A	•	•	‡	‡	R	•
	EORB	C8	2	2	D8	3	2	E8	5	2	F8	4	3				B ⊕ M → B	•	•	‡	‡	R	•
Increment	INC							6C	7	2	7C	6	3				M + 1 → M	•	•	‡	‡	⑤	•
	INCA													4C	2	1	A + 1 → A	•	•	‡	‡	⑤	•
	INCB													5C	2	1	B + 1 → B	•	•	‡	‡	⑤	•
Load Acmltr	LDAA	86	2	2	96	3	2	A6	5	2	B6	4	3				M → A	•	•	‡	‡	R	•
	LDAB	C6	2	2	D6	3	2	E6	5	2	F6	4	3				M → B	•	•	‡	‡	R	•
Or, Inclusive	ORAA	8A	2	2	9A	3	2	AA	5	2	BA	4	3				A + M → A	•	•	‡	‡	R	•
	ORAB	CA	2	2	DA	3	2	EA	5	2	FA	4	3				B + M → B	•	•	‡	‡	R	•
Push Data	PSHA													36	4	1	A → Msp, SP − 1 → SP	•	•	•	•	•	•
	PSHB													37	4	1	B → Msp, SP − 1 → SP	•	•	•	•	•	•
Pull Data	PULA													32	4	1	SP + 1 → SP, Msp → A	•	•	•	•	•	•
	PULB													33	4	1	SP + 1 → SP, Msp → B	•	•	•	•	•	•
Rotate Left	ROL							69	7	2	79	6	3				M	•	•	‡	‡	⑥	‡
	ROLA													49	2	1	A	•	•	‡	‡	⑥	‡
	ROLB													59	2	1	B — C b7 ← b0	•	•	‡	‡	⑥	‡
Rotate Right	ROR							66	7	2	76	6	3				M	•	•	‡	‡	⑥	‡
	RORA													46	2	1	A	•	•	‡	‡	⑥	‡
	RORB													56	2	1	B — C b7 → b0	•	•	‡	‡	⑥	‡
Shift Left, Arithmetic	ASL							68	7	2	78	6	3				M	•	•	‡	‡	⑥	‡
	ASLA													48	2	1	A — C b7 b0 ← 0	•	•	‡	‡	⑥	‡
	ASLB													58	2	1	B — C b7 b0	•	•	‡	‡	⑥	‡
Shift Right, Arithmetic	ASR							67	7	2	77	6	3				M	•	•	‡	‡	⑥	‡
	ASRA													47	2	1	A	•	•	‡	‡	⑥	‡
	ASRB													57	2	1	B — b7 b0 C	•	•	‡	‡	⑥	‡
Shift Right, Logic	LSR							64	7	2	74	6	3				M	•	•	R	‡	⑥	‡
	LSRA													44	2	1	A — 0 → b7 b0 C	•	•	R	‡	⑥	‡
	LSRB													54	2	1	B — b7 b0 C	•	•	R	‡	⑥	‡
Store Acmltr	STAA				97	4	2	A7	6	2	B7	5	3				A → M	•	•	‡	‡	R	•
	STAB				D7	4	2	E7	6	2	F7	5	3				B → M	•	•	‡	‡	R	•
Subtract	SUBA	80	2	2	90	3	2	A0	5	2	B0	4	3				A − M → A	•	•	‡	‡	‡	‡
	SUBB	C0	2	2	D0	3	2	E0	5	2	F0	4	3				B − M → B	•	•	‡	‡	‡	‡
Subtract Acmltrs	SBA													10	2	1	A − B → A	•	•	‡	‡	‡	‡
Subtr with Carry	SBCA	82	2	2	92	3	2	A2	5	2	B2	4	3				A − M − C → A	•	•	‡	‡	‡	‡
	SBCB	C2	2	2	D2	3	2	E2	5	2	F2	4	3				B − M − C → B	•	•	‡	‡	‡	‡
Transfer Acmltrs	TAB													16	2	1	A → B	•	•	‡	‡	R	•
	TBA													17	2	1	B → A	•	•	‡	‡	R	•
Test Zero or Minus	TST							6D	7	2	7D	6	3				M − 00	•	•	‡	‡	R	R
	TSTA													4D	2	1	A − 00	•	•	‡	‡	R	R
	TSTB													5D	2	1	B − 00	•	•	‡	‡	R	R

LEGEND:

OP	Operation Code (Hexadecimal)	+	Boolean Inclusive OR
~	Number of MPU Cycles	⊕	Boolean Exclusive OR
#	Number of Program Bytes	M̄	Complement of M
+	Arithmetic Plus	→	Transfer Into
−	Arithmetic Minus	0	Bit = Zero
•	Boolean AND	00	Byte = Zero
Msp	Contents of memory location pointed to be Stack Pointer		

CONDITION CODE SYMBOLS:

H	Half-carry from bit 3	R	Reset Always
I	Interrupt mask	S	Set Always
N	Negative (sign bit)	‡	Test and set if true, cleared otherwise
Z	Zero (byte)	•	Not Affected
V	Overflow, 2's complement		
C	Carry from bit 7		

(Note) Accumulator addressing mode instructions are included in the column for IMPLIED addressing.

CONDITION CODE REGISTER NOTES:

(Bit set if test is true and cleared otherwise)

① (Bit V) Test: Result = 10000000?
② (Bit C) Test: Result ≠ 00000000?
③ (Bit C) Test: Decimal value of most significant BCD Character greater than nine? (Not cleared if previously set.)
④ (Bit V) Test: Operand = 10000000 prior to execution?
⑤ (Bit V) Test: Operand = 01111111 prior to execution?
⑥ (Bit V) Test: Set equal to result of N⊕C after shift has occurred.

■ PROGRAM CONTROL OPERATIONS

Program Control operation can be subdivided into two categories: (1) Index Register/Stack Pointer instructions: (2) Jump and Branch operations.

● Index Register/Stack Pointer Operations

The instructions for direct operation on the MPU's Index Register and Stack Pointer are summarized in Table 5. Decrement (DEX, DES), increment (INX, INS), load (LDX, LDS), and store (STX, STS) instructions are provided for both. The Compare instruction, CPX, can be used to compare the Index Register to a 16-bit value and update the Condition Code Register accordingly.

The TSX instruction causes the Index Register to be loaded with the address of the last data byte put onto the "stack". The TXS instruction loads the Stack Pointer with a value equal to one less than the current contents of the Index Register. This causes the next byte to be pulled from the "stack" to come from the location indicated by the Index Register. The utility of these two instructions can be clarified by describing the "stack" concept relative to the HMCS 6800 system.

The "stack" can be thought of as a sequential list of data stored in the MPU's read/write memory. The Stack Pointer contains a 16-bit memory address that is used to access the list from one end on a last-in-first-out (LIFO) basis in contrast to the random access mode used by the MPU's other addressing modes.

The HD6800 MPU instruction set and interrupt structure allow extensive use of the stack concept for efficient handling of data movement, subroutines and interrupts. The instructions can be used to establish one or more "stacks" anywhere in read/write memory. Stack length is limited only by the amount of memory that is made available.

Operation of the Stack Pointer with the Push and Pull instructions is illustrated in Figs. 19 and 20. The Push instruction (PSHA) causes the contents of the indicated accumulator (A in this example) to be stored in memory at the location indicated by the Stack Pointer. The Stack Pointer is automatically decremented by one following the storage operation and is "pointing" to the next empty stack location.

The Pull instruction (PULA or PULB) causes the last byte stacked to be loaded into the appropriate accumulator. The Stack Pointer is automatically incremented by one just prior to the data transfer so that it will point to the last byte stacked rather than the next empty location. Note that the PULL instruction does not "remove" the data from memory; in the example, 1A is still in location (m+1) following execution of PULA. A subsequent PUSH instruction would overwrite than location with the new "pushed" data.

Execution of the Branch to Subroutine (BSR) and Jump to Subroutine (JSR) instructions cause a return address to be save on the stack as shown in Figs. 21 through 23. The stack is decremented after each byte of the return address is pushed onto the stack. For both of the these instructions, the return address is the memory location following the bytes of code that correspond to the BSR and JSR instruction. The code required for BSR or JSR may be either two or three bytes, depending on whether the JSR is in the indexed (two bytes) or the extended (three bytes) addressing mode. Before it is stacked, the Program Counter is automatically incremented the correct number of times to be pointing at the location of the next instruction. The Return from Subroutine instruction, RTS, causes the return address to be retrieved and loaded into the Program Counter as shown in Fig. 24.

There are several operations that cause the status of the MPU to be saved on the stack. The Software Interrupt (SWI) and Wait for Interrupt (WAI) instructions as well as the maskable ($\overline{IRQ}$) and non-maskable ($\overline{NMI}$) hardware interrupts all cause the MPU's internal registers (except for the Stack Pointer itself) to be stacked as shown in Fig. 25. MPU status is restored by the Return from interrupt, RTI, as shown in Fig. 26.

Table 5 Index Register and Stack Pointer Instructions

Operation	Mnemonic	Addressing Modes														Boolean/ Arithmetic Operation	Cond. Code Reg.					
		IMMED			DIRECT			INDEX			EXTND			IMPLIED				5	4	3	2	1	0
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#		H	I	N	Z	V	C
Compare Index Reg	CPX	8C	3	3	9C	4	2	AC	6	2	BC	5	3				$(X_H) - (M), (X_L) - (M+1)$	•	•	①	↕	②	•
Decrement Index Reg	DEX													09	4	1	$X - 1 \rightarrow X$	•	•	•	↕	•	•
Decrement Stack Pntr	DES													34	4	1	$SP - 1 \rightarrow SP$	•	•	•	•	•	•
Increment Index Reg	INX													08	4	1	$X + 1 \rightarrow X$	•	•	•	↕	•	•
Increment Stack Pntr	INS													31	4	1	$SP + 1 \rightarrow SP$	•	•	•	•	•	•
Load Index Reg	LDX	CE	3	3	DE	4	2	EE	6	2	FE	5	3				$M \rightarrow X_H, (M+1) \rightarrow X_L$	•	•	③	↕	R	•
Load Stack Pntr	LDS	8E	3	3	9E	4	2	AE	6	2	BE	5	3				$M \rightarrow SP_H, (M+1) \rightarrow SP_L$	•	•	③	↕	R	•
Store Index Reg	STX				DF	5	2	EF	7	2	FF	6	3				$X_H \rightarrow M, X_L \rightarrow (M+1)$	•	•	③	↕	R	•
Store Stack Pntr	STS				9F	5	2	AF	7	2	BF	6	3				$SP_H \rightarrow M, SP_L \rightarrow (M+1)$	•	•	③	↕	R	•
Index Reg → Stack Pntr	TXS													35	4	1	$X - 1 \rightarrow SP$	•	•	•	•	•	•
Stack Pntr → Index Reg	TSX													30	4	1	$SP + 1 \rightarrow X$	•	•	•	•	•	•

① (Bit N) Test: Sign bit of most significant (MS) byte of result = 1?
② (Bit V) Test: 2's complement overflow from subtraction of ms bytes?
③ (Bit N) Test: Result less than zero? (Bit 15 = 1)

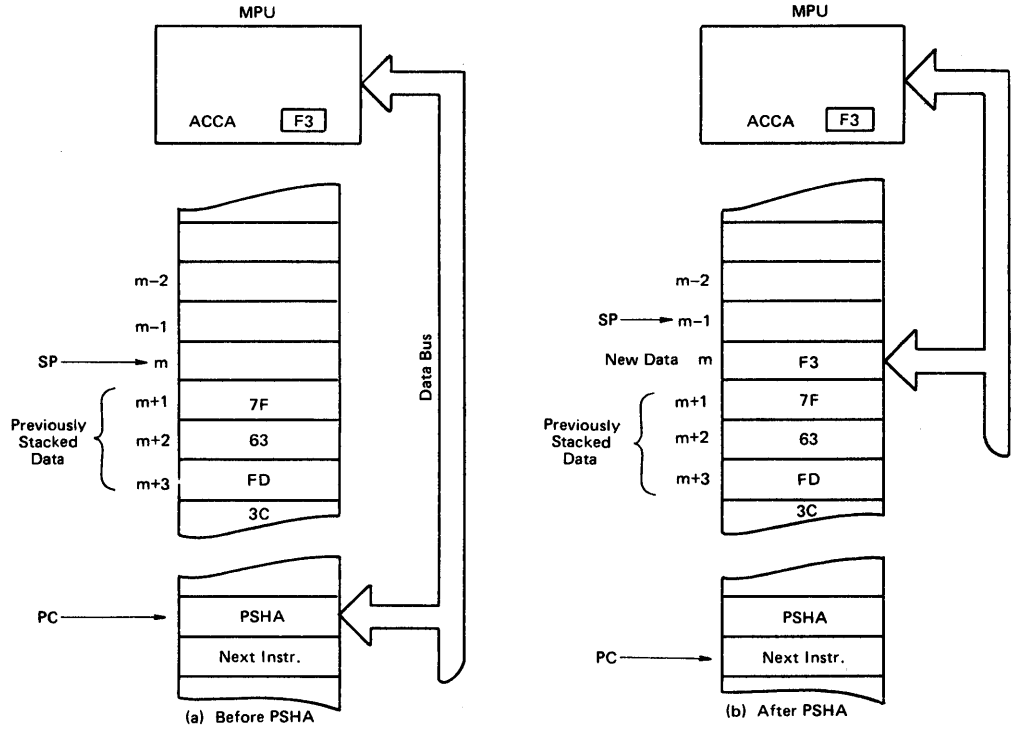


Figure 19 Stack Operation (Push Instruction)

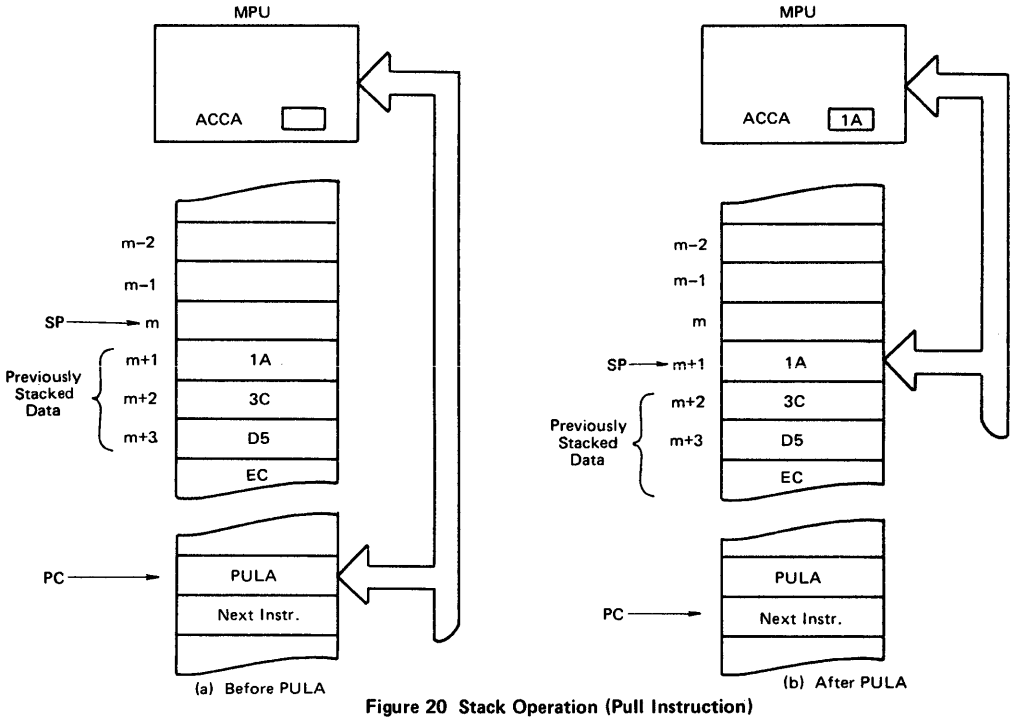


Figure 20 Stack Operation (Pull Instruction)

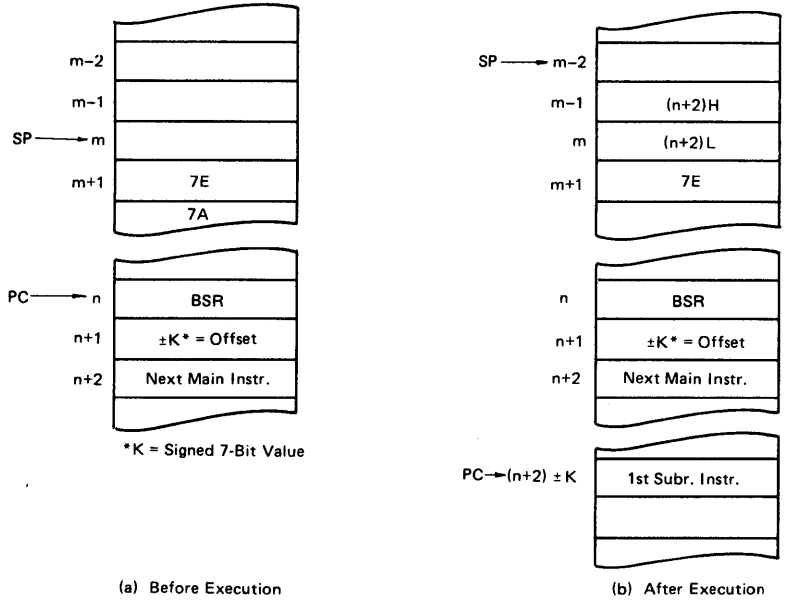


Figure 21 Program Flow for BSR

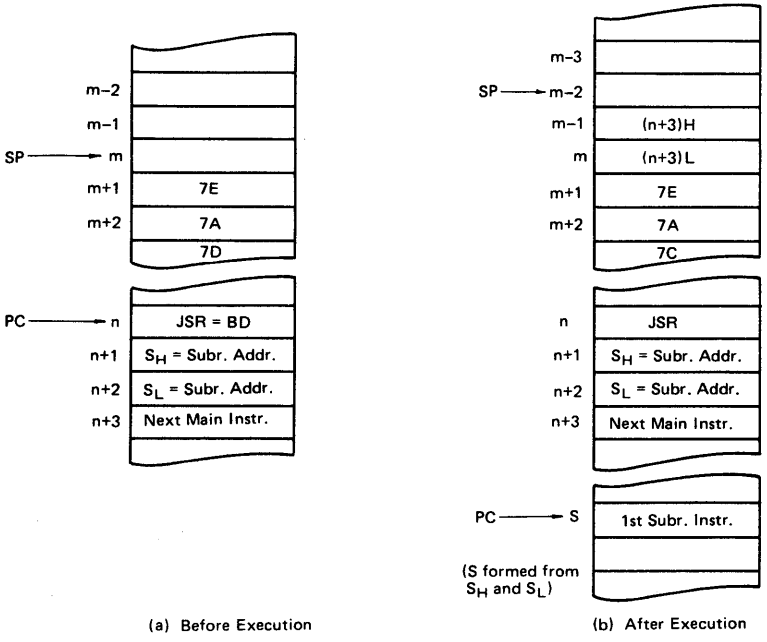


Figure 22 Program Flow for JSR (Extended)

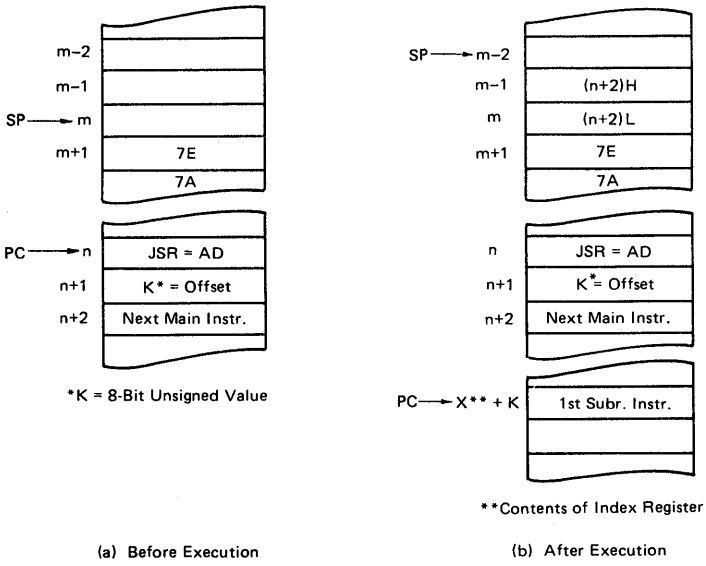


Figure 23 Program Flow for JSR (Indexed)

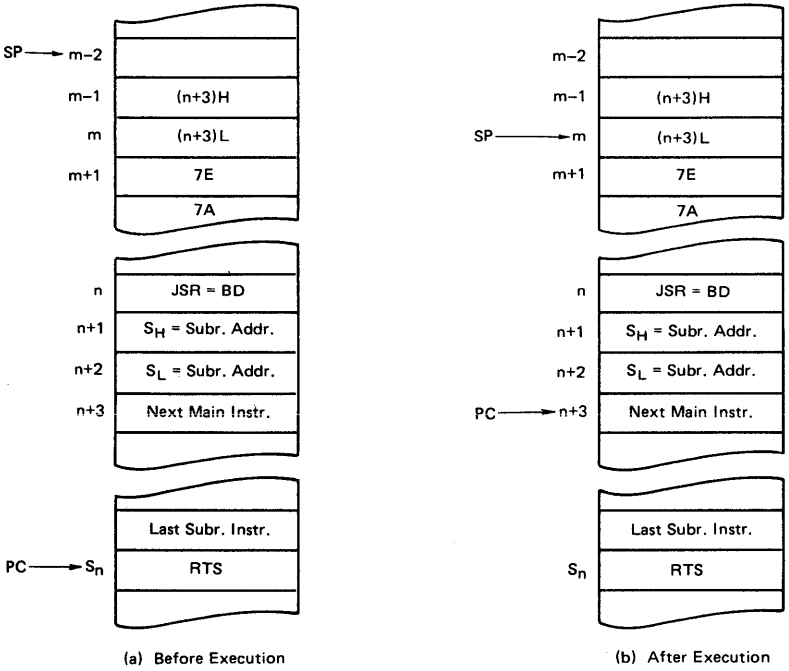


Figure 24 Program Flow for RTS

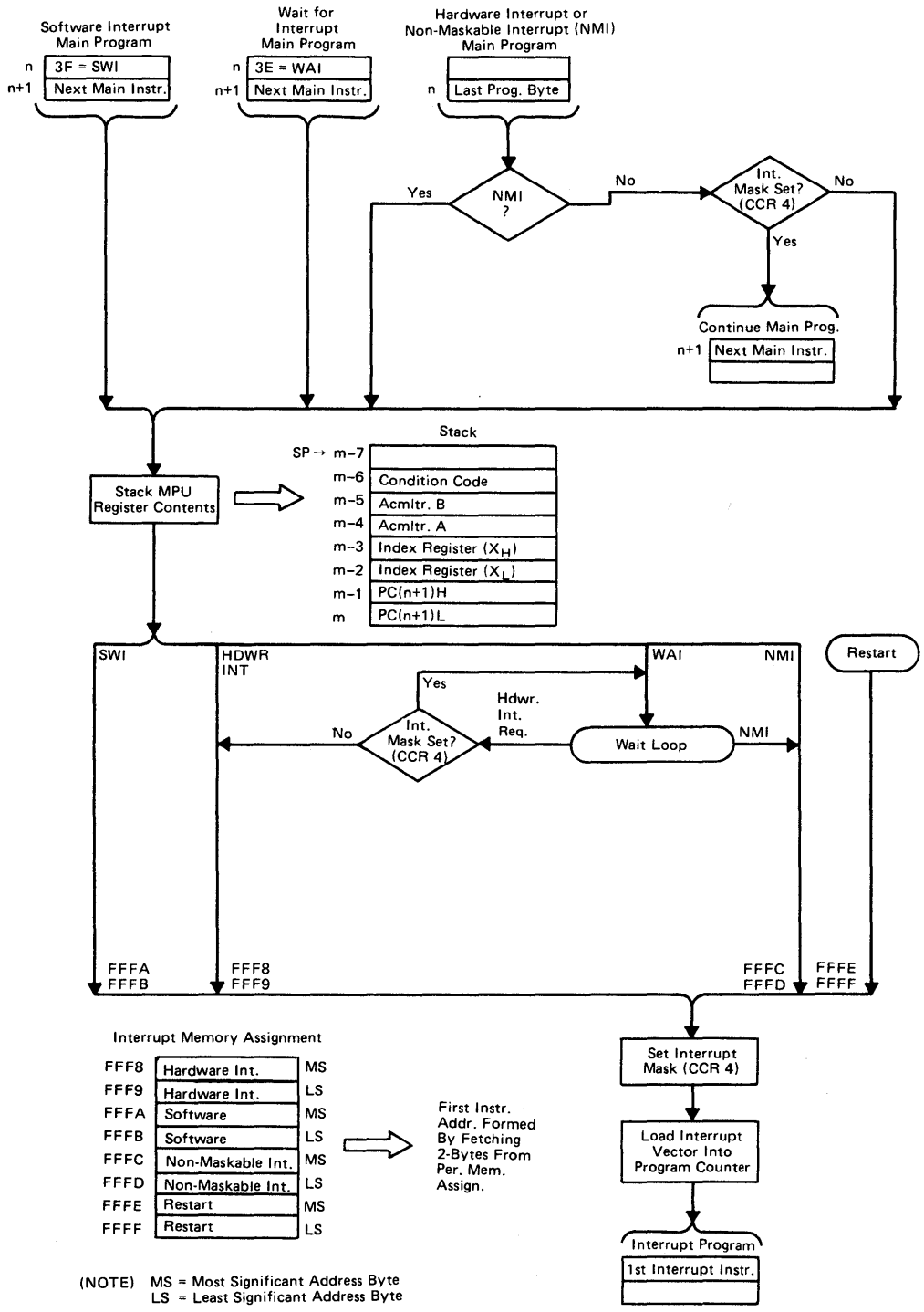


Figure 25 Program Flow for Interrupts

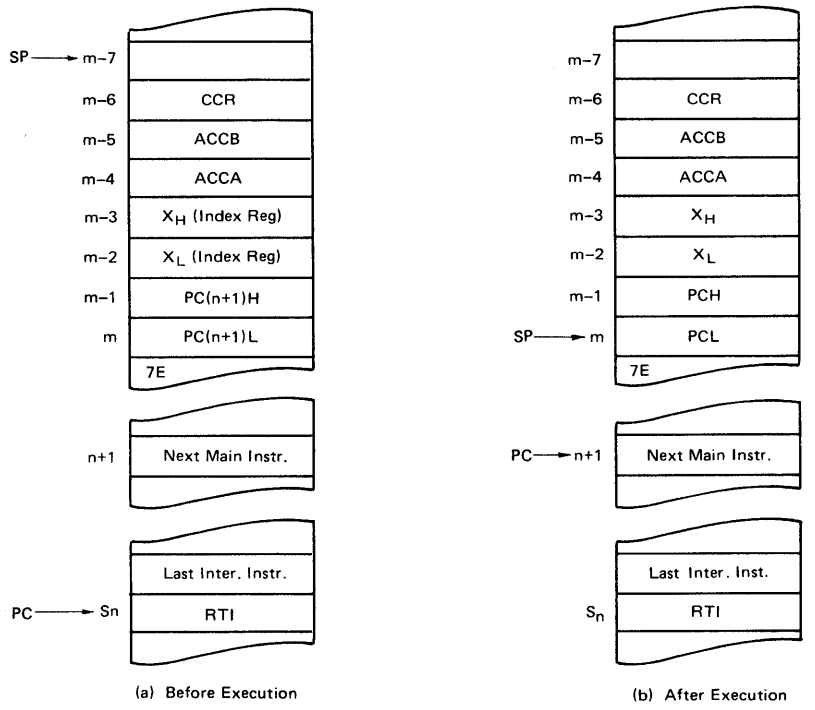


Figure 26 Program Flow for RTI

- **Jump and Branch Operation**

The Jump and Branch instructions are summarized in Table 6. These instructions are used to control the transfer of operation from one point to another in the control program.

The No Operation instruction, NOP, while included here, is a jump operation in a very limited sense. Its only effect is to increment the Program Counter by one. It is useful during program development as a "stand-in" for some other instruction that is to be determined during debug. It is also used for equalizing the execution time through alternate paths in a control program.

Execution of the Jump Instruction, JMP, and Branch Always, BRA, affects program flow as shown in Fig. 27. When the MPU encounters the Jump (Index) instruction, it adds the offset to the value in the Index Register and uses the result as the address of the next instruction to be executed. In the extended addressing mode, the address of the next instruction to be executed is fetched from the two locations immediately following the JMP instruction. The Branch Always (BRA) instruction is similar to the JMP (extended) instruction except that the relative addressing mode applies and the branch is limited to the range within −125 or +127 bytes of the branch instruction itself. The opcode for the BRA instruction requires one less byte than JMP (extended) but takes one more cycle to execute.

The effect on program flow for the Jump to Subroutine (JSR) and Branch to Subroutine (BSR) is shown in Figs. 21 through 23. Note that the Program Counter is properly incremented to be pointing at the correct return address before it is stacked. Operation of the Branch to Subroutine and Jump to Subroutine (extended) instruction is similar except for the range. The BSR instruction requires less opcode than JSR (2 bytes versus 3 bytes) and also executes one cycle faster than JSR. The Return from Subroutine, RTS, is used at the end of a subroutine to return to the main program as indicated in Fig. 24.

The effect of executing the Software Interrupt, SWI, and the Wait for Interrupt, WAI, and their relationship to the hardware interrupts is shown in Fig. 25. SWI causes the MPU contents to be stacked and then fetches the starting address of the interrupt routine from the memory locations that respond to the addresses FFFA and FFFB. Note that as in the case of the subroutine instructions, the Program Counter is incremented to point at the correct return address before being stacked. The Return from Interrupt instruction, RTI, (Fig. 26) is used at the end of an interrupt routine to restore control to the main program. The SWI instruction is useful for inserting break points in the control program, that is, it can be used to stop operation and put the MPU registers in memory where they can be examined. The WAI instruction is used to decrease the time required to service a hardware interrupt; it stacks the MPU contents and then waits for the interrupt to occur, effectively removing the stacking time from a hardware interrupt sequence.

Table 6 JUMP/BRANCH Instruction

Operation	Mnemonic	Addressing Modes												Branch Test	Cond. Code Reg.					
		RELATIVE			INDEX			EXTND			IMPLIED				5	4	3	2	1	0
		OP	~	#	OP	~	#	OP	~	#	OP	~	#		H	I	N	Z	V	C
Branch Always	BRA	20	4	2										None	•	•	•	•	•	•
Branch If Carry Clear	BCC	24	4	2										C = 0	•	•	•	•	•	•
Branch If Carry Set	BCS	25	4	2										C = 1	•	•	•	•	•	•
Branch If = Zero	BEQ	27	4	2										Z = 1	•	•	•	•	•	•
Branch If ≧ Zero	BGE	2C	4	2										N ⊕ V = 0	•	•	•	•	•	•
Branch If > Zero	BGT	2E	4	2										Z + (N ⊕ V) = 0	•	•	•	•	•	•
Branch If Higher	BHI	22	4	2										C + Z = 0	•	•	•	•	•	•
Branch If ≦ Zero	BLE	2F	4	2										Z + (N ⊕ V) = 1	•	•	•	•	•	•
Branch If Lower Or Same	BLS	23	4	2										C + Z = 1	•	•	•	•	•	•
Branch If < Zero	BLT	2D	4	2										N ⊕ V = 1	•	•	•	•	•	•
Branch If Minus	BMI	2B	4	2										N = 1	•	•	•	•	•	•
Branch If Not Equal Zero	BNE	26	4	2										Z = 0	•	•	•	•	•	•
Branch If Overflow Clear	BVC	28	4	2										V = 0	•	•	•	•	•	•
Branch If Overflow Set	BVS	29	4	2										V = 1	•	•	•	•	•	•
Branch If Plus	BPL	2A	4	2										N = 0	•	•	•	•	•	•
Branch To Subroutine	BSR	8D	8	2											•	•	•	•	•	•
Jump	JMP				6E	4	2	7E	3	3					•	•	•	•	•	•
Jump To Subroutine	JSR				AD	8	2	BD	9	3					•	•	•	•	•	•
No Operation	NOP										01	2	1	Advances Prog Cntr Only	•	•	•	•	•	•
Return From Interrupt	RTI										3B	10	1		———			①	———	
Return From Subroutine	RTS										39	5	1		•	•	•	•	•	•
Software Interrupt	SWI										3F	12	1		•	S	•	•	•	•
Wait for Interrupt	WAI										3E	9	1		•	②	•	•	•	•

① (All) Load Condition Code Register from Stack. (See Special Operations)
② (Bit I) Set when interrupt occurs. If previously set, a Non-Maskable interrupt is required to exit the wait state.

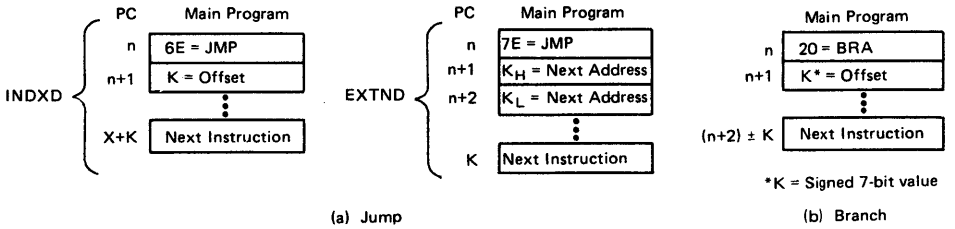


Figure 27 Program Flow for JUMP/BRANCH Instructions

```
BMI : N = 1 ;        BEQ : Z = 1 ;
BPL : N = 0 ;        BNE : Z = 0 ;

BVC : V = 0 ;        BCC : C = 0 ;
BVS : V = 1 ;        BCS : C = 1 ;

BHI : C + Z = 0 ;    BLT : N ⊕ V = 1 ;
BLS : C + Z = 1 ;    BGE : N ⊕ V = 0 ;

      BLE : Z + (N ⊕ V) = 1 ;
      BGT : Z + (N ⊕ V) = 0 ;
```

Figure 28 Conditional Branch Instructions

The conditional branch instructions, Fig. 28, consists of seven pairs of complementary instructions. They are used to test the results of the preceding operation and either continue with the next instruction in sequence (test fails) or cause a branch to another point in the program (test succeeds).

Four of the pairs are used for simple tests of status bits N, Z, V, and C:

1. Branch on Minus (BMI) and Branch On Plus (BPL) tests the sign bit, N, to determine if the previous result was negative or positive, respectively.
2. Branch On Equal (BEQ) and Branch On Not Equal (BNE) are used to test the zero status bit, Z, to determine whether or not the result of the previous operation was equal to "0". These two instructions are useful following a Compare (CMP) instruction to test for equality between an accumulator and the operand. They are also used following the Bit Test (BIT) to determine whether or not the same bit positions are set in an accumulator and the operand.

3. Branch On Overflow Clear (BVC) and Branch On Overflow Set (BVS) tests the state of the V bit to determine if the previous operation caused an arithmetic overflow.
4. Branch On Carry Clear (BCC) and Branch On Carry Set (BCS) tests the state of the C bit to determine if the previous operation caused a carry to occur. BCC and BCS are useful for testing relative magnitude when the values being tested are regarded as unsigned binary numbers, that is, the values are in the range "00" (lowest) of "FF" (highest). BCC following a comparison (CMP) will cause a branch if the (unsigned) value in the accumulator is higher than or the same as the value of the operand. Conversely, BCS will cause a branch if the accumulator value is lower than the operand.

The Fifth complementary pair, Branch On Higher (BHI) and Branch On Lower or Same (BLS) are in a sense complements to BCC and BCS. BHI tests for both C and Z = "0", if used following a CMP, it will cause a branch if the value in the accumulator is higher than the operand. Conversely, BLS will cause a branch if the unsigned binary value in the accumulator is lower than or the same as the operand.

The remaining two pairs are useful in testing results of operations in which the values are regarded as signed two's complement numbers. This differs from the unsigned binary case in the following sense: In unsigned, the orientation is higher or lower; in signed two's complement, the comparison is between larger or smaller where the range of values is between −128 and +127.

Branch On Less Than Zero (BLT) and Branch On Greater Than Or Equal Zero (BGE) test the status bits for N⊕V = "1" and N ⊕ V = "0", respectively. BLT will always cause a branch following an operation in which two negative numbers were added. In addition, it will cause a branch following a CMP in which the value in the accumulator was negative and the operand was positive. BLT will never cause a branch following a CMP in which the accumulator value was positive and the operand negative. BGE, the complement to BLT, will cause a branch following operations in which two positive values were added or in which the result was "0".

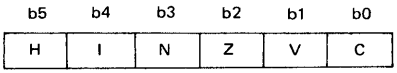
The last pair, Branch On Less Than Or Equal Zero (BLE) and Branch On Greater Than Zero (BGT) test the status bits for Z ⊕ (N + V) = "1" and Z ⊕(N + V) = "0", respectively, The action of BLE is identical to that for BLT except that a branch will also occur if the result of the previous result was "0". Conversely, BGT is similar to BGE except that no branch will occur following a "0" result.

■ CONDITION CODE REGISTER OPERATIONS

The Condition Code Register (CCR) is a 6-bit register within the MPU that is useful in controlling program flow during system operation. The bits are defined in Fig. 29.

The instructions shown in Table 7 are available to the user for direct manipulation of the CCR. In addition, the MPU automatically sets or clears the appropriate status bits as many of the other instructions on the condition code register was indicated as they were introduced.

Systems which require an interrupt window to be opened under program control should use a CLI-NOP-SEI sequence rather than CLI-SEI.



H = Half-carry; set whenever a carry from b3 to b4 of the result is generated by ADD, ABA, ADC; cleared if no b3 to b4 carry; not affected by other instructions.

I = Interrupt Mask; set by hardware of software interrupt or SEI instruction; cleared by CLI instruction. (Normally not used in arithmetic operations.) Restored to a "0" as a result of an RTI instruction if IM stored on the stacked is "0"

N = Negative; set if high order bit (b7) of result is set; cleared otherwise.

Z = Zero; set if result = "0"; cleared otherwise.

V = Overflow; set if there was arithmetic overflow as a result of the operation; cleared otherwise.

C = Carry; set if there was a carry from the most significant bit (b7) of the result; cleared otherwise.

Figure 29 Condition Code Register Bit Definition

■ ADDRESSING MODES

The MPU operates on 8-bit binary numbers presented to it via the Data Bus. A given number (byte) may represent either data or an instruction to be executed, depending on where it is encountered in the control program. The HD6800 MPU has 72 unique instructions, however, it recognizes and takes action on 197 of the 256 possibilities that can occur using an 8-bit word length. This larger number of instructions results from the fact that many of the executive instructions have more than one addressing mode.

Table 7 Condition Code Register Instructions

Operations	Mnemonic	Addressing Mode IMPLIED			Boolean Operation	Cond. Code Reg.					
						5	4	3	2	1	0
		OP	~	#		H	I	N	Z	V	C
Clear Carry	CLC	0C	2	1	0 → C	•	•	•	•	•	R
Clear Interrupt Mask	CLI	0E	2	1	0 → I	•	R	•	•	•	•
Clear Overflow	CLV	0A	2	1	0 → V	•	•	•	•	R	•
Set Carry	SEC	0D	2	1	1 → C	•	•	•	•	•	S
Set Interrupt Mask	SEI	0F	2	1	1 → I	•	S	•	•	•	•
Set Overflow	SEV	0B	2	1	1 → V	•	•	•	•	S	•
Acmltr A → CCR	TAP	06	2	1	A → CCR	①					
CCR → Acmltr A	TPA	07	2	1	CCR → A	•	•	•	•	•	•

R = Reset
S = Set
• = Not affected
① (ALL) Set according to the contents of Accumulator A.

These addressing modes refer to the manner in which the program causes the MPU to obtain its instructions and data. The programmer must have a method for addressing the MPU's internal registers and all of the external memory locations.

Selection of the desired addressing mode is made by the user as the source statements are written. Translation into appropriate opcode then depends on the method used. If manual translation is used, the addressing mode is inherent in the opcode. For example, the Immediate, Direct, Indexed, and Extended modes may all be used with the ADD instruction. The proper mode is determined by selecting (hexidecimal notation) 8B, 9B, AB, or BB, respectively.

The source statement format includes adequate information for the selection if an assembler program is used to generate the opcode. For instance, the Immediate mode is selected by the Assembler whenever it encounters the "#" symbol in the operand field. Similarly, an "X" in the operand field causes the Indexed mode to be selected. Only the Relative mode applies to the branch instructions, therefore, the mnemonic instruction itself is enough for the Assembler to determine addressing mode.

For the instructions that use both Direct and Extended modes, the Assembler selects the Direct mode if the operand value is in the range 0~255 and Extended otherwise. There are a number of instructions for which the Extended mode is valid but the Direct is not. For these instructions, the Assembler automatically selects the Extended mode even if the operand is in the 0~255 range. The addressing modes are summarized in Fig. 30.

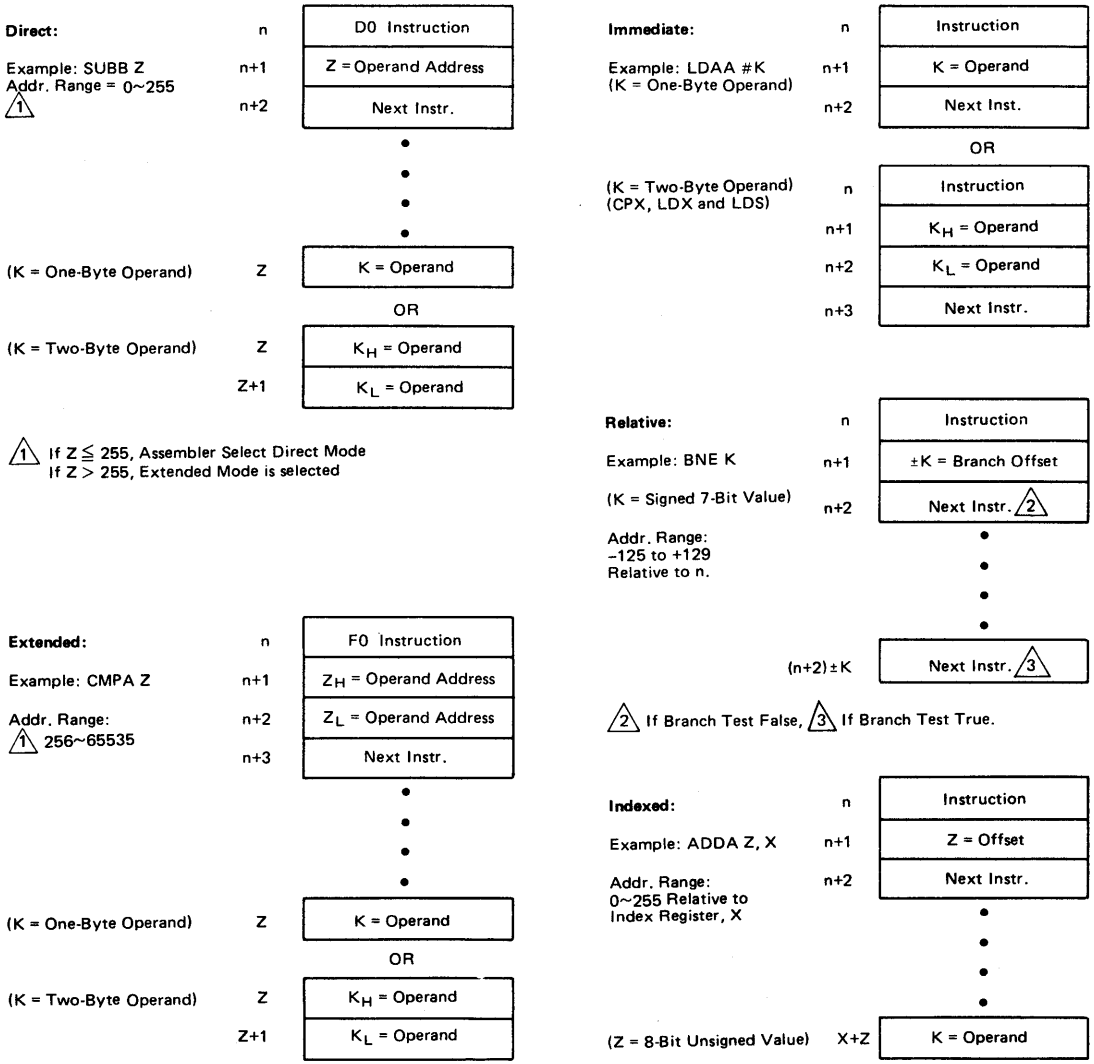


Figure 30 Addressing Mode Summary

- **Inherent (Includes "Accumulator Addressing" Mode)**

The successive fields in a statement are normally separated by one or more spaces. An exception to this rule occurs for instructions that use dual addressing in the operand field and for instructions that must distinguish between the two accumulators. In these cases, A and B are "operands" but the space between them and the operator may be omitted. This is commonly done, resulting in apparent four character mnemonics for those instructions.

The addition instruction, ADD, provides an example of dual addressing in the operand fields;

Operator	Operand	Comment
ADDA	MEM12	ADD CONTENTS OF MEM12 TO ACCA
or ADDB	MEM12	ADD CONTENTS OF MEM12 TO ACCB

The example used earlier for the test instruction, TST, also applies to the accumulators and uses the "accumulator addressing mode" to designate which of the two accumulators is being tested:

Operator	Comment
TSTB	TEST CONTENTS OF ACCB
or TSTA	TEST CONTENTS OF ACCA

A number of the instructions either alone or together with an accumulator operand contain all of the address information that is required, that is, "inherent" in the instruction, itself. For instance, the instruction ABA causes the MPU to add the contents of accumulators A and B together and place the result in accumulator A. The instruction INCB, another example of "accumulator addressing", causes the contents of accumulator B to be increased by one. Similarly, INX, increment the Index Register, causes the contents of the Index Register to be increased by one.

Program flow for instructions of this type is illustrated in Figures 31 and 32. In these figures, the general case is shown on the left and a specific example is shown on the right. Numerical examples are in decimal notation. Instructions of this type require only one byte of opcode. Cycle-by-cycle operation of the inherent mode is shown in Table 8.

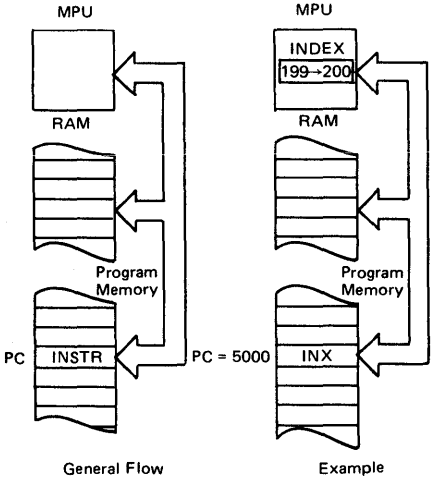


Figure 31 Inherent Addressing

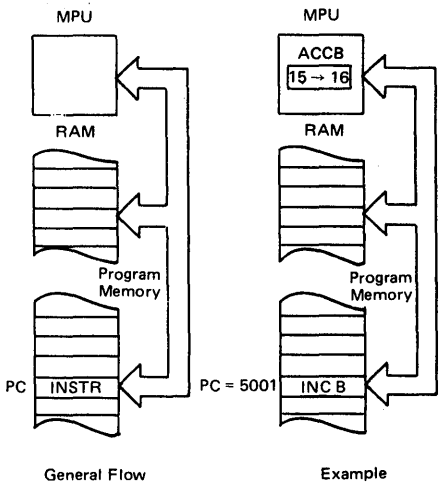


Figure 32 Accumulator Addressing

- **Immediate Addressing Mode**

In the Immediate addressing mode, the operand is the value that is to be operated on. For instance, the instruction

Operator	Operand	Comment
LDAA	#25	LOAD 25 INTO ACCA

causes the MPU to "immediately load accumulator A with the value 25"; no further address reference is required. The Immediate mode is selected by preceding the operand value with the "#" symbol. Program flow for this addressing mode is illustrated in Fig. 33.

The operand format allows either properly defined symbols or numerical values. Except for the instructions CPX, LDX, and LDS, the operand may be any value in the range 0 ~ 255. Since Compare Index Register (CPX), Load Index Register (LDX), Load Stack Pointer (LDS), require 16-bit values, the immediate mode for these three instructions requie two-byte operands.

Table 9 shows the cycle-by-cycle operation for the immediate addressing mode.

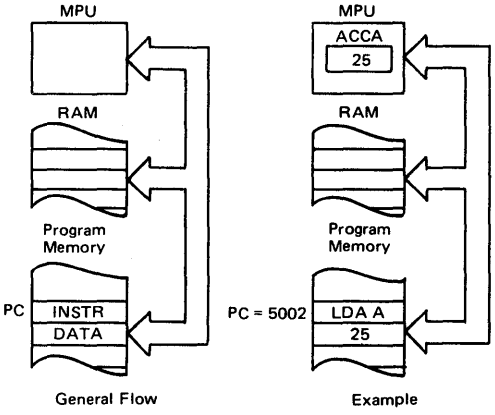


Figure 33 Immediate Addressing Mode

Table 8 Inherent Mode Cycle by Cycle Operation

Address Mode and Instructions	Cycle	Cycle #	VMA Line	Address Bus	R/$\overline{W}$ Line	Data Bus
ABA DAA SEC ASL DEC SEI ASR INC SEV CBA LSR TAB CLC NEG TAP CLI NOP TBA CLR ROL TPA CLV ROR TST COM SBA	2	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Op Code of Next Instruction
DES DEX INS INX	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Op Code of Next Instruction
		3	0	Previous Register Contents	1	Irrelevant Data (NOTE 1)
		4	0	New Register Contents	1	Irrelevant Data (NOTE 1)
PSH	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Op Code of Next Instruction
		3	1	Stack Pointer	0	Accumulator Data
		4	0	Stack Pointer − 1	1	Accumulator Data
PUL	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Op Code of Next Instruction
		3	0	Stack Pointer	1	Irrelevant Data (NOTE 1)
		4	1	Stack Pointer + 1	1	Operand Data from Stack
TSX	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Op Code of Next Instruction
		3	0	Stack Pointer	1	Irrelevant Data (NOTE 1)
		4	0	New Index Register	1	Irrelevant Data (NOTE 1)
TXS	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Op Code of Next Instruction
		3	0	Index Register	1	Irrelevant Data
		4	0	New Stack Pointer	1	Irrelevant Data
RTS	5	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Irrelevant Data (NOTE 2)
		3	0	Stack Pointer	1	Irrelevant Data (NOTE 1)
		4	1	Stack Pointer + 1	1	Address of Next Instruction (High Order Byte)
		5	1	Stack Pointer + 2	1	Address of Next Instruction (Low Order Byte)
WAI	9	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Op Code of Next Instruction
		3	1	Stack Pointer	0	Return Address (Low Order Byte)
		4	1	Stack Pointer − 1	0	Return Address (High Order Byte)
		5	1	Stack Pointer − 2	0	Index Register (Low Order Byte)
		6	1	Stack Pointer − 3	0	Index Register (High Order Byte)
		7	1	Stack Pointer − 4	0	Contents of Accumulator A
		8	1	Stack Pointer − 5	0	Contents of Accumulator B
		9	1	Stack Pointer − 6 (NOTE 3)	1	Contents of Cond. Code Register
RTI	10	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Irrelevant Data (NOTE 2)
		3	0	Stack Pointer	1	Irrelevant Data (NOTE 1)
		4	1	Stack Pointer + 1	1	Contents of Cond. Code Register from Stack
		5	1	Stack Pointer + 2	1	Contents of Accumulator B from Stack
		6	1	Stack Pointer + 3	1	Contents of Accumulator A from Stack
		7	1	Stack Pointer + 4	1	Index Register from Stack (High Order Byte)
		8	1	Stack Pointer + 5	1	Index Register from Stack (Low Order Byte)
		9	1	Stack Pointer + 6	1	Next Instruction Address from Stack (High Order Byte)
		10	1	Stack Pointer + 7	1	Next Instruction Address from Stack (Low Order Byte)
SWI	12	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Irrelevant Data (NOTE 1)
		3	1	Stack Pointer	0	Return Address (Low Order Byte)
		4	1	Stack Pointer − 1	0	Return Address (High Order Byte)
		5	1	Stack Pointer − 2	0	Index Register (Low Order Byte)
		6	1	Stack Pointer − 3	0	Index Register (High Order Byte)
		7	1	Stack Pointer − 4	0	Contents of Accumulator A
		8	1	Stack Pointer − 5	0	Contents of Accumulator B
		9	1	Stack Pointer − 6	0	Contents of Cond. Code Register
		10	0	Stack Pointer − 7	1	Irrelevant Data (NOTE 1)
		11	1	Vector Address FFFA (Hex)	1	Address of Subroutine (High Order Byte)
		12	1	Vector Address FFFB (Hex)	1	Address of Subroutine (Low Order Byte)

NOTE 1. If device which is addressed during this cycle uses VMA, then the Data Bus will go to the high impedance three-state condition. Depending on bus capacitance, data from the previous cycle may be retained on the Data Bus.
NOTE 2. Data is ignored by the MPU.
NOTE 3. While the MPU is waiting for the interrupt, Bus Available will go "High" indicating the following states of the control lines: VMA is "Low"; Address Bus, R/$\overline{W}$, and Data Bus are all in the high impedance state.

Table 9 Immediate Mode Cycle by Cycle Operation

Address Mode and Instructions	Cycle	Cycle #	VMA Line	Address Bus	R/$\overline{W}$ Line	Data Bus
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	2	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Operand Data
CPX	3	1	1	Op Code Address	1	Op Code
LDS		2	1	Op Code Address + 1	1	Operand Data (High Order Byte)
LDX		3	1	Op Code Address + 2	1	Operand Data (Low Order Byte)

- **Direct and Extended Addressing Modes**

In the Direct and Extended modes of addressing, the operand field of the source statement is the address of the value that is to be operated on. The Direct and Extended modes differ only in the range of memory locations to which they can direct the MPU. Direct addressing generates a single 8-bit operand and, hence, can address only memory locations 0 ~ 255; a two byte operand is generated for Extended addressing, enabling the MPU to reach the remaining memory locations, 256 ~ 65535. An example of Direct addressing and its effect on program flow is illustrated in Fig. 34.

Table 10 shows the cycle-by-cycle operations of this mode.

The MPU, after encountering the opcode for the instrution LDAA (Direct) at memory location 5004 (Program Counter = 5004), looks in the next location, 5005, for the address of the operand. It then sets the program counter equal to the value found there (100 in the example) and fetches the operand, in this case a value to be loaded into accumulator A, from that location. For instructions requiring a two-byte operand such as LDX (Load the Index Register), the operand bytes would be retrieved from locations 100 and 101.

Extended addressing, Fig. 35, is similar except that a two-byte address is obtained from locations 5007 and 5008 after the LDAB (Extended) opcode shows up in location 5006. Extended addressing can be thought of as the "standard" addressing mode, that is, it is a method of reaching anyplace in memory. Direct addressing, since only one address byte is required, provides a faster method of processing data and generates fewer bytes of control code. In most applications, the direct addressing range, memory locations 0 ~ 255, are reserved for RAM. They are used for data buffering and temporary storage of system variables, the area in which faster addressing is of most value, Cycle-by-cycle operation is shown in Table 11 for Extended Addressing.

Table 10 Direct Mode Cycle by Cycle Operation

Address Mode and Instructions	Cycle	Cycle #	VMA Line	Address Bus	R/$\overline{W}$ Line	Data Bus
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	3	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand
		3	1	Address of Operand	1	Operand Data
CPX LDS LDX	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand
		3	1	Address of Operand	1	Operand Data (High Order Byte)
		4	1	Operand Address + 1	1	Operand Data (Low Order Byte)
STA	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Destination Address
		3	0	Destination Address	1	Irrelevant Data (NOTE 1)
		4	1	Destination Address	0	Data from Accumulator
STS STX	5	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand
		3	0	Address of Operand	1	Irrelevant Data (NOTE 1)
		4	1	Address of Operand	0	Register Data (High Order Byte)
		5	1	Address of Operand + 1	0	Register Data (Low Order Byte)

NOTE 1. If device which is address during this cycle uses VMA, then the Data Bus will go to the high impedance three-state condition. Depending on bus capacitance, data from the previous cycle may be retained on the Data Bus.

Table 11 Extended Mode Cycle by Cycle

Address Mode and Instructions	Cycle	Cycle #	VMA Line	Address Bus	R/$\overline{W}$ Line	Data Bus
STS STX	6	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand (High Order Byte)
		3	1	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	0	Address of Operand	1	Irrelevant Data (NOTE 1)
		5	1	Address of Operand	0	Operand Data (High Order Byte)
		6	1	Address of Operand + 1	0	Operand Data (Low Order Byte)
JSR	9	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Subroutine (High Order Byte)
		3	1	Op Code Address + 2	1	Address of Subroutine (Low Order Byte)
		4	1	Subroutine Starting Address	1	Op Code of Next Instruction
		5	1	Stack Pointer	0	Return Address (Low Order Byte)
		6	1	Stack Pointer − 1	0	Return Address (High Order Byte)
		7	0	Stack Pointer − 2	1	Irrelevant Data (NOTE 1)
		8	0	Op Code Address + 2	1	Irrelevant Data (NOTE 1)
		9	1	Op Code Address + 2	1	Address of Subroutine (Low Order Byte)
JMP	3	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Jump Address (High Order Byte)
		3	1	Op Code Address + 2	1	Jump Address (Low Order Byte)
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand (High Order Byte)
		3	1	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	1	Address of Operand	1	Operand Data
CPX LDS LDX	5	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand (High Order Byte)
		3	1	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	1	Address of Operand	1	Operand Data (High Order Byte)
		5	1	Address of Operand + 1	1	Operand Data (Low Order Byte)
STA A STA B	5	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Destination Address (High Order Byte)
		3	1	Op Code Address + 2	1	Destination Address (Low Order Byte)
		4	0	Operand Destination Address	1	Irrelevant Data (NOTE 1)
		5	1	Operand Destination Address	0	Data from Accumulator
ASL LSR ASR NEG CLR ROL COM ROR DEC TST INC	6	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand (High Order Byte)
		3	1	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	1	Address of Operand	1	Current Operand Data
		5	0	Address of Operand	1	Irrelevant Data (NOTE 1)
		6	1/0 (NOTE 2)	Address of Operand	0	New Operand Data (NOTE 2)

NOTE 1. If device which is addressed during this cycle uses VMA, then the Data Bus will go to the high impedance three-state condition. Depending on bus capacitance, data from the previous cycle may be retained on the Data Bus.
NOTE 2. For TST, VMA = 0 and Operand data does not change.

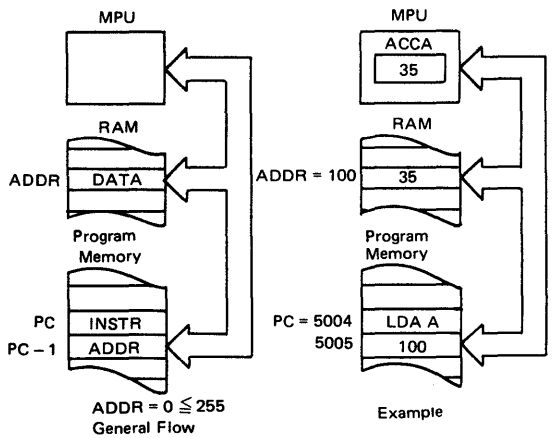


Figure 34 Direct Addressing Mode

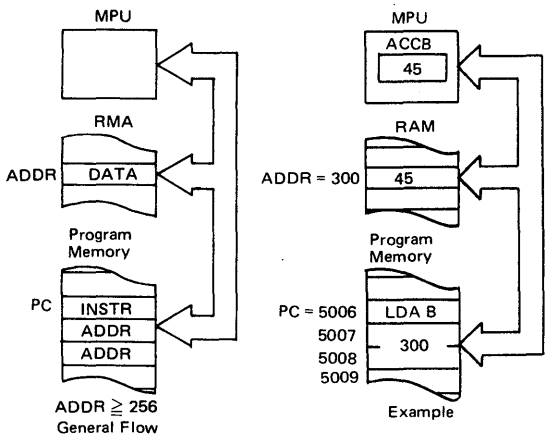


Figure 35 Extended Addressing Mode

- **Relative Address Mode**

In both the Direct and Extended modes, the address obtained by the MPU is an absolute numerical address. The Relative addressing mode, implemented for the MPU's branch instructions, specifies a memory location relative to the Program Counter's current location. Branch instructions generate two bytes of machine code, one for the instruction opcode and one for the "relative" address (see Fig. 36). Since it is desirable to be able to branch in either direction, the 8-bit address byte is interpreted as a signed 7-bit value; the 8th bit of the operand is treated as a sign bit, "0" = plus and "1" = minus. The remaining seven bits represent the numerical value. This result in a relative addressing range of ±127 with respect to the location of the branch instruction itself. However, the branch range is computed with respect to the next instruction that would be executed if the branch conditions are not satisfied. Since two byte are generated, the next instruction is located at PC+2. If, D is defined as the address of the branch destination, the range is then;

$$(PC+2) - 128 \leqq D \leqq (PC+2) + 127$$

or $$PC - 126 \leqq D \leqq PC + 129$$

that is, the destination of the branch instruction must be within -126 to +129 memory locations of the branch instruction itself. For transferring control beyond this range, the unconditional jump (JMP), jump to subroutine (JSR), and return from subroutine (RTS) are used.

In Fig. 36, when the MPU encounters the opcode for BEQ (Branch if result of last instruction was zero), it tests the Zero bit in the Condition Code Register. If that bit is "0", indicating a non-zero result, the MPU continues execution with the next instruction (in location 5010 in Fig. 36). If the previous result was zero, the branch condition is satisfied and the MPU adds the offset, 15 in this case, to PC+2 and branches to location 5025 for the next instruction.

The branch instructions allow the programmer to efficiently direct the MPU to one point or another in the control program depending on the outcome of test results. Since the control program is normally in read-only memory and cannot be changed, the relative address used in execution of branch instructions is a constant numerical value. Cycle-by-cycle operation is shown in Table 12 for relative addressing.

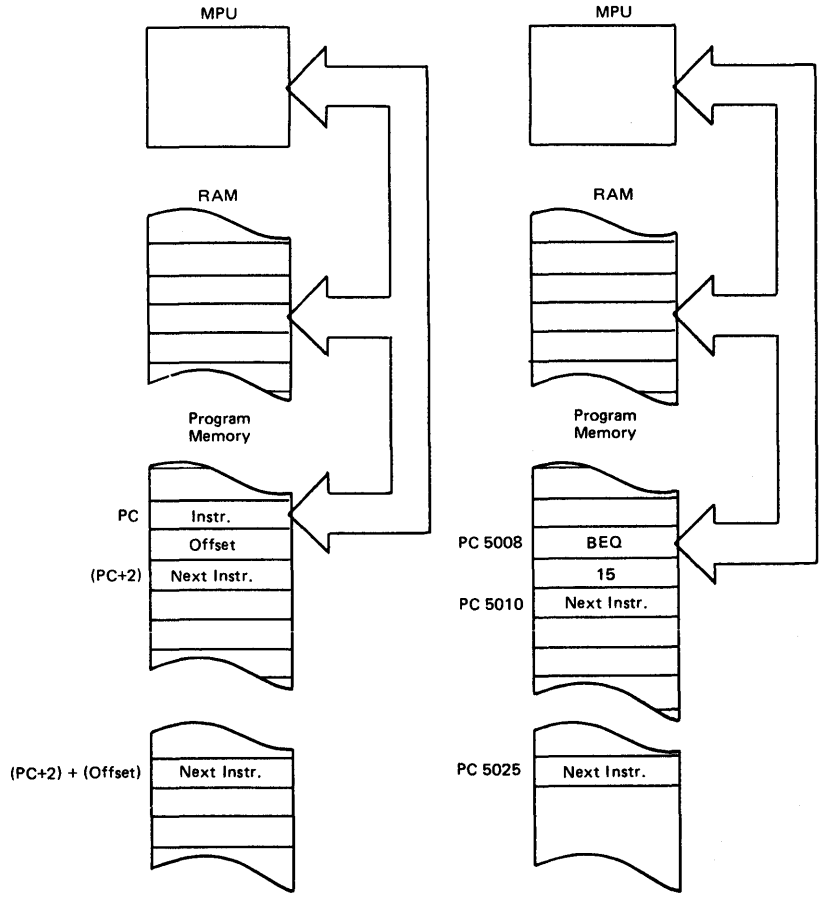


Figure 36 Relative Addressing Mode

Table 12 Relative Mode Cycle-by-Cycle Operation

Address Mode and Instructions	Cycle	Cycle #	VMA Line	Address Bus	R/$\overline{W}$ Line	Data Bus
BCC BHI BNE BCS BLE BPL BEQ BLS BRA BGE BLT BVC BGT BMI BVS	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Branch Offset
		3	0	Op Code Address + 2	1	Irrelevant Data (NOTE 1)
		4	0	Branch Address	1	Irrelevant Data (NOTE 1)
BSR	8	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Branch Offset
		3	0	Return Address of Main Program	1	Irrelevant Data (NOTE 1)
		4	1	Stack Pointer	0	Return Address (Low Order Byte)
		5	1	Stack Pointer − 1	0	Return Address (High Order Byte)
		6	0	Stack Pointer − 2	1	Irrelevant Data (NOTE 1)
		7	0	Return Address of Main Program	1	Irrelevant Data (NOTE 1)
		8	0	Subroutine Address	1	Irrelevant Data (NOTE 1)

NOTE 1. If device which is addressed during this cycle uses VMA, then the Data Bus will go to the high impedance three-state condition. Depending on bus capacitance, data from the previous cycle may be retained on the Data Bus.

- **Indexed Addressing Mode**

With Indexed addressing the numerical address is variable and depend on the current contents of the Index Register. A source statement such as

Operator	Operand	Comment
STAA	X	PUT A IN INDEXED LOCATION

causes the MPU to store the contents of accumulator A in the memory location specified by the contents of the Index Register (recall that the label X is reserved to designate the Index Register). Since there are instructions for manipulating X during program execution (LDX, INX, DEX, etc.), the Indexed addressing mode provides a dynamic "on the fly" way to modify program activity.

The operand field can also contain a numerical value that will be automatically added to X during execution. This format is illustrated in Fig. 37.

When the MPU encounters the LDAB (Indexed) opcode in location 5006, it looks in the next memory location for the value to be added to X (5 in the example) and calculates the required address by adding 5 to the present Index Register value of 400. In the operand format, the offset may be represented by a label or a numerical value in the range 0 ~ 255 as in the example. In the earlier example, STAA X, the operand is equivalent to 0, X , that is, the "0" may be omitted when the desired address is equal to X. Table 13 shows the cycle-by-cycle operation for the Indexed Mode of Addressing.

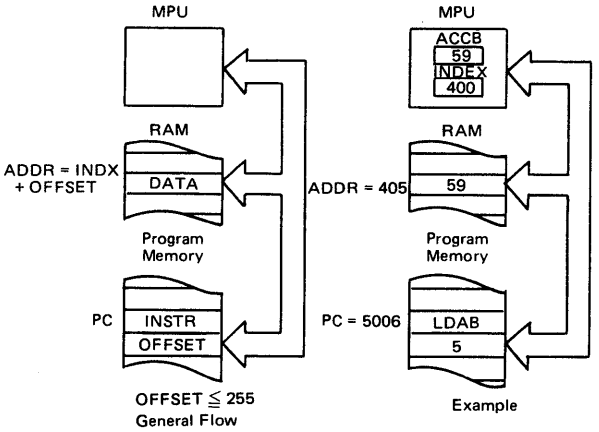


Figure 37 Indexed Addressing Mode

Table 13 Indexed Mode Cycle by Cycle

Address Mode and Instructions	Cycle	Cycle #	VMA Line	Address Bus	R/$\overline{W}$ Line	Data Bus
JMP	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (NOTE 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (NOTE 1)
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	5	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (NOTE 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (NOTE 1)
		5	1	Index Register Plus Offset	1	Operand Data
CPX LDS LDX	6	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (NOTE 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (NOTE 1)
		5	1	Index Register Plus Offset	1	Operand Data (High Order Byte)
		6	1	Index Register Plus Offset + 1	1	Operand Data (Low Order Byte)
STA	6	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (NOTE 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (NOTE 1)
		5	0	Index Register Plus Offset	1	Irrelevant Data (NOTE 1)
		6	1	Index Register Plus Offset	0	Operand Data
ASL LSR ASR NEG CLR ROL COM ROR DEC TST INC	7	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (NOTE 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (NOTE 1)
		5	1	Index Register Plus Offset	1	Current Operand Data
		6	0	Index Register Plus Offset	1	Irrelevant Data (NOTE 1)
		7	1/0 (NOTE 2)	Index Register Plus Offset	0	New Operand Data (NOTE 2)
STS STX	7	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (NOTE 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (NOTE 1)
		5	0	Index Register Plus Offset	1	Irrelevant Data (NOTE 1)
		6	1	Index Register Plus Offset	0	Operand Data (High Order Byte)
		7	1	Index Register Plus Offset + 1	0	Operand Data (Low Oder Byte)
JSR	8	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (NOTE 1)
		4	1	Stack Pointer	0	Return Address (Low Order Byte)
		5	1	Stack Pointer − 1	0	Return Address (High Order Byte)
		6	0	Stack Pointer − 2	1	Irrelevant Data (NOTE 1)
		7	0	Index Register	1	Irrelevant Data (NOTE 1)
		8	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (NOTE 1)

NOTE 1. If Device which is addressed during this cycle uses VMA, then the Data Bus will go to the high impedance three-state condition. Depending on bus capacitance, data from the previous cycle may be retained on the Data Bus.
NOTE 2. For TST, VMA = 0 and Operand data does not change.

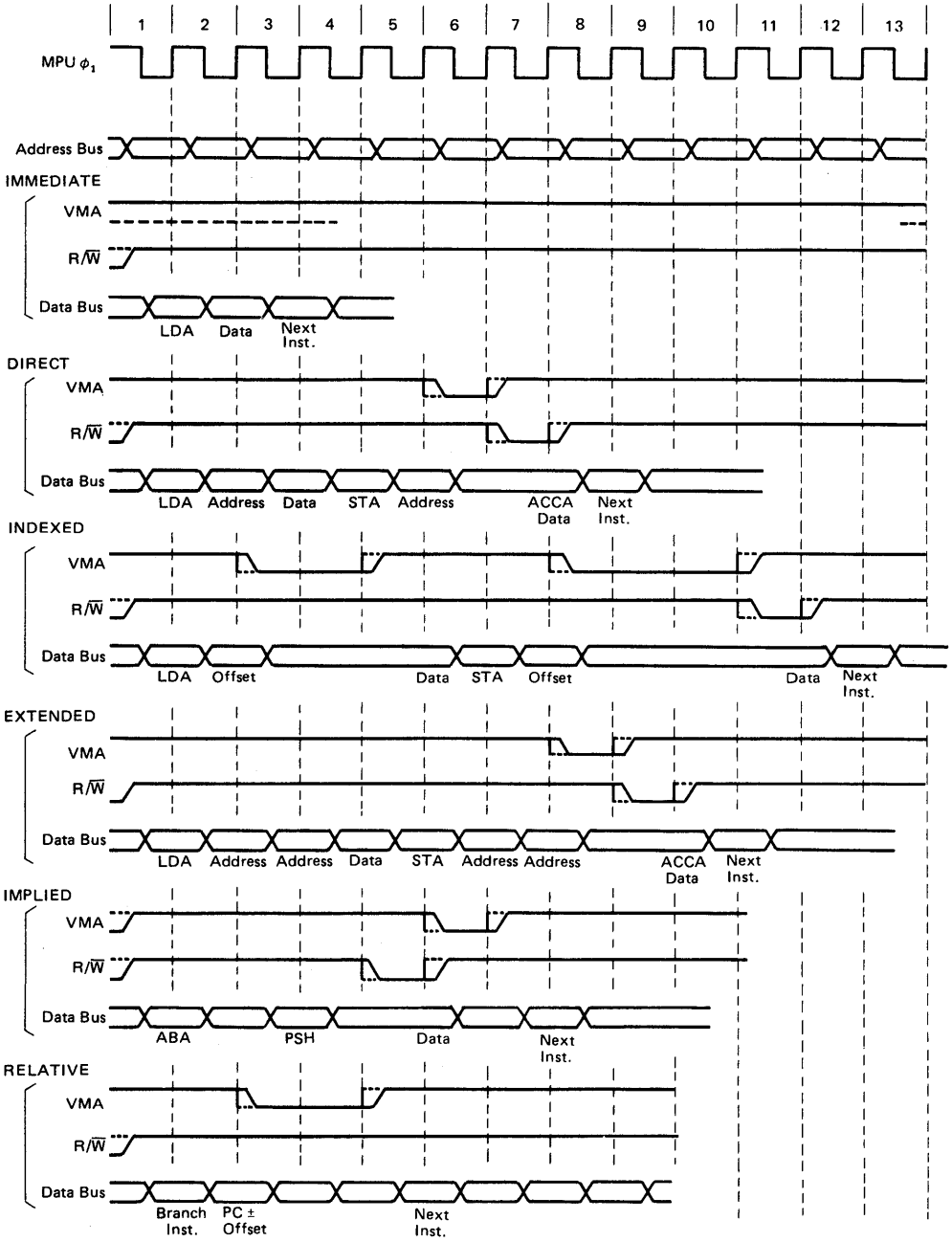


Figure 38 Example of Excution Timing in Each Addressing Mode

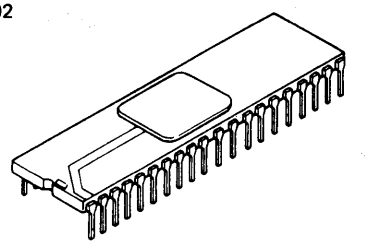
HD6802

MPU (Microprocessor with Clock and RAM)

The HD6802 is a monolithic 8-bit microprocessor that contains all the registers and accumulators of the present HD6800 plus an internal clock oscillator and driver on the same chip. In addition, the HD6802 has 128 bytes of RAM on the chip located at hex addresses 0000 to 007F. The first 32 bytes of RAM, at hex addresses 0000 to 001F, may be retained in a low power mode by utilizing V_{CC} standby, thus facilitating memory retention during a power-down situation.

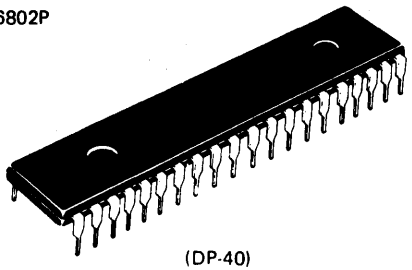
The HD6802 is completely software compatible with the HD6800 as well as the entire HMCS6800 family of parts. Hence, the HD6802 is expandable to 65k words.

HD6802



(DC-40)

HD6802P

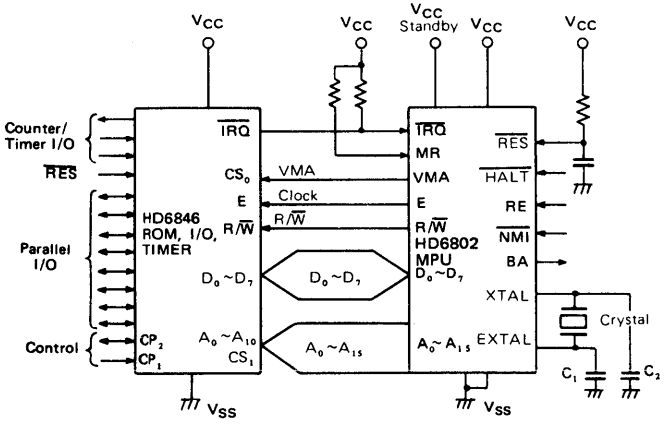


(DP-40)

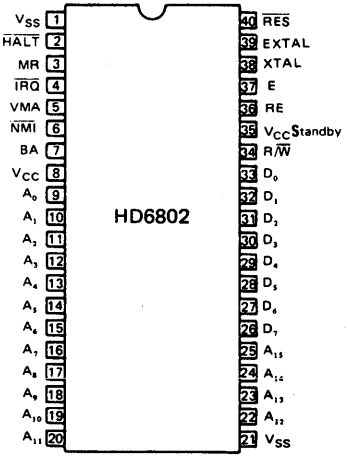
■ FEATURES

- On-Chip Clock Circuit
- 128 × 8 Bit On-Chip RAM
- 32 Bytes of RAM are Retainable
- Software-Compatible with the HD6800
- Expandable to 65k words
- Standard TTL-Compatible Inputs and Outputs
- 8 Bit Word Size
- 16 Bit Memory Addressing
- Interrupt Capability
- Compatible with MC6802

■ BLOCK DIAGRAM



■ PIN ARRANGEMENT



(Top View)

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC} * V_{CC} Standby*	-0.3 ~ +7.0	V
Input Voltage	V_{in}*	-0.3 ~ +7.0	V
Operating Temperature	T_{opr}	-20 ~ +75	°C
Storage Temperature	T_{stg}	-55 ~ +150	°C

* With respect to V_{SS} (SYSTEM GND)

(NOTE) Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ RECOMMENDED OPERATING CONDITIONS

Item	Symbol		min	typ	max	Unit
Supply Voltage	V_{CC} * V_{CC} Standby*		4.75	5.0	5.25	V
Input Voltage	V_{IL}*		-0.3	–	0.8	V
	V_{IH}*	Except $\overline{RES}$	2.0	–	V_{CC}	V
		$\overline{RES}$	4.25	–	V_{CC}	V
Operation Temperature	T_{opr}		-20	25	75	°C

* With respect to V_{SS} (SYSTEM GND)

■ ELECTRICAL CHARACTERISTICS

● DC CHARACTERISTICS (V_{CC}=5.0V±5%, V_{CC} Standby=5.0V±5%, V_{SS}=0V, Ta=-20~+75°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ**	max	Unit
Input "High" Voltage	Except $\overline{RES}$	V_{IH}		2.0	–	V_{CC}	V
	$\overline{RES}$			4.25	–	V_{CC}	
Input "Low" Voltage	Except $\overline{RES}$	*** V_{IL}		-0.3	–	0.8	V
	$\overline{RES}$			-0.3	–	0.8	
Output "High" Voltage	D_0~D_7, E	V_{OH}	I_{OH} = -205μA	2.4	–	–	V
	A_0~A_{15}, R/$\overline{W}$, VMA		I_{OH} = -145μA	2.4	–	–	
	BA		I_{OH} = -100μA	2.4	–	–	
Output "Low" Voltage		V_{OL}	I_{OL} = 1.6mA	–	–	0.4	V
Three State (Off State) Input Current	D_0~D_7	I_{TSI}	V_{in} = 0.4~2.4V	-10	–	10	μA
Input Leakage Current	Except D_0~D_7 ****	I_{in}	V_{in} = 0~5.25V	-2.5	–	2.5	μA
Power Dissipation		P_D*		–	0.6	1.2	W
Input Capacitance	D_0~D_7	C_{in}	V_{in}=0V, T_a=25°C, f=1.0MHz	–	10	12.5	pF
	Except D_0~D_7			–	6.5	10	
Output Capacitance	A_0~A_{15}, R/$\overline{W}$, BA, VMA, E	C_{out}	V_{in}=0V, T_a=25°C, f=1.0MHz	–	–	12	pF

* In power-down mode, maximum power dissipation is less than 42mW.
** T_a=25°C, V_{CC}=5V
*** As $\overline{RES}$ input has histeresis character, applied voltage up to 2.4V is regarded as "Low" level when it goes up from 0V.
**** Does not include EXTAL and XTAL, which are crystal inputs.

• AC CHARACTERISTICS (V_{CC}=5.0V±5%, V_{CC} Standby=5.0V±5%, V_{SS}=0V, Ta=-20~+75°C, unless otherwise noted.)

1. CLOCK TIMING CHARACTERISTICS

Item		Symbol	Test Condition	min	typ	max	Unit
Frequency of Operation	Input Clock ÷ 4	f		0.1	–	1.0	MHz
	Crystal Frequency	f_{XTAL}		1.0	–	4.0	
Cycle Time		t_{cyc}	Fig. 2, Fig. 3	1.0	–	10	μs
Clock Pulse Width	"High" Level	$PW_{\phi H}$	at 2.4V (Fig. 2, Fig. 3)	450	–	4500	ns
	"Low" Level	$PW_{\phi L}$	at 0.8V (Fig. 2, Fig. 3)				
Clock Fall Time		t_{ϕ}	0.8V ~ 2.4V(Fig.2,Fig.3)	–	–	25	ns

2. READ/WRITE TIMING

Item	Symbol	Test Condition	min	typ*	max	Unit
Address Delay	t_{AD}	Fig. 2, Fig. 3, Fig. 6	–	–	270	ns
Peripheral Read Access Time	t_{acc}	Fig. 2	–	–	530	ns
Data Setup Time (Read)	t_{DSR}	Fig. 2	100	–	–	ns
Input Data Hold Time	t_H	Fig. 2	10	–	–	ns
Output Data Hold Time	t_H	Fig. 3	20	–	–	ns
Address Hold Time (Address, R/$\overline{W}$, VMA)	t_{AH}	Fig. 2, Fig. 3	10	–	–	ns
Data Delay Time (Write)	t_{DDW}	Fig. 3	–	165	225	ns
Bus Available Delay	t_{BA}	Fig. 4, Fig. 5, Fig. 7, Fig. 8	–	–	250	ns
Processor Controls Processor Control Setup Time	t_{PCS}	Fig. 4~Fig. 7, Fig. 12	200	–	–	ns
Processor Control Rise and Fall Time (Measured at 0.8V and 2.0V)	t_{PCr} t_{PCf}	Fig. 4~Fig. 7, Fig. 12, Fig. 13, Fig. 16	–	–	100	ns

*Ta = 25°C, V_{CC} = 5V

3. POWER DOWN SEQUENCE TIMING, POWER UP RESET TIMING AND MEMORY READY TIMING

Item	Symbol	Test Condition	min	typ	max	Unit
RAM Enable Reset Time (1)	t_{RE1}	Fig. 13	150	–	–	ns
RAM Enable Reset Time (2)	t_{RE2}	Fig. 13	E-3 cycles	–	–	
Reset Release Time	t_{LRES}	Fig. 12	20*	–	–	ms
RAM Enable Reset Time (3)	t_{RE3}	Fig. 12	0	–	–	ns
Memory Ready Setup Time	t_{SMR}	Fig. 16	300	–	–	ns
Memory Ready Hold Time	t_{HMR}	Fig. 16	0	–	200	ns

*t_{RES} = 20 msec min. for S type, 50 msec min. for R type.

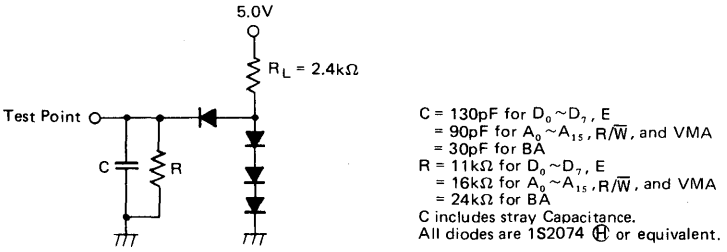


Figure 1 Bus Timing Test Load

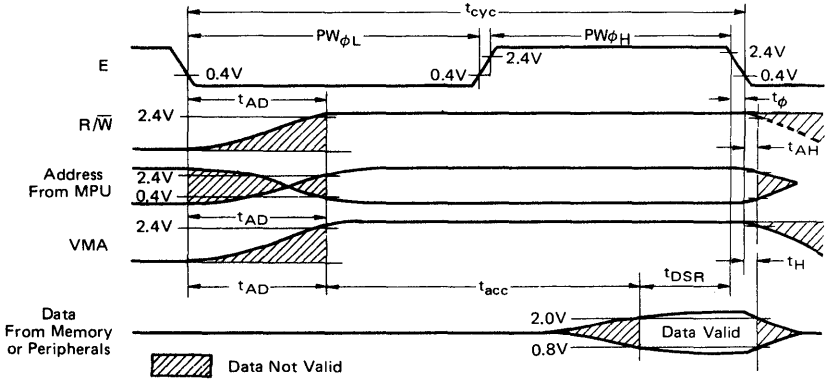


Figure 2 Read Data from Memory or Peripherals

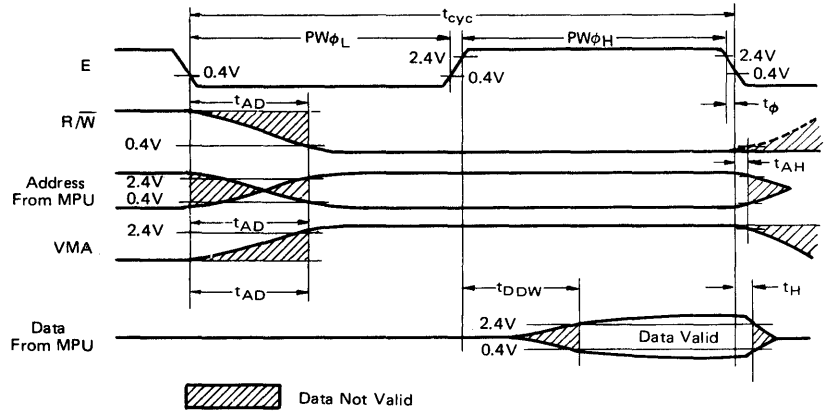


Figure 3 Write Data in Memory or Peripherals

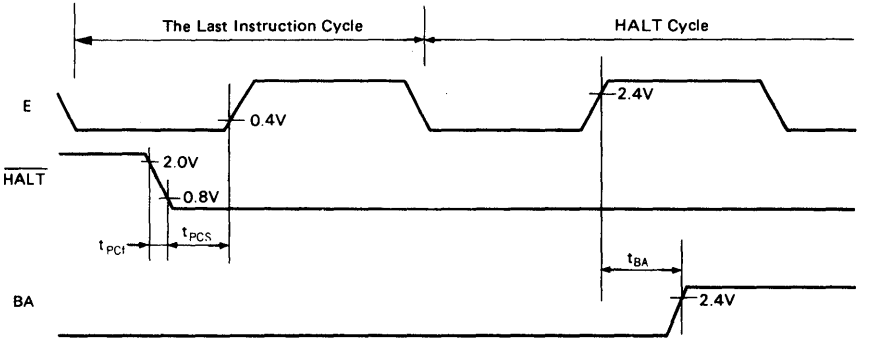


Figure 4 Timing of $\overline{\text{HALT}}$ and BA

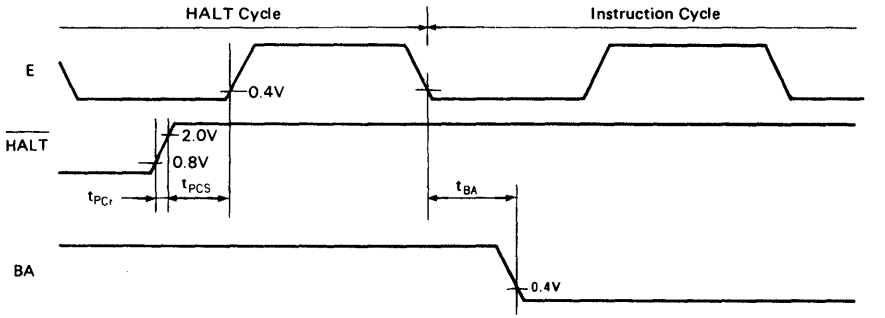


Figure 5 Timing of $\overline{\text{HALT}}$ and BA

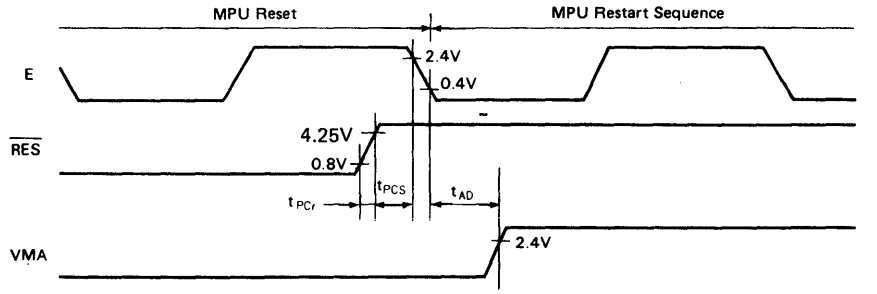


Figure 6 $\overline{\text{RES}}$ and MPU Restart Sequence

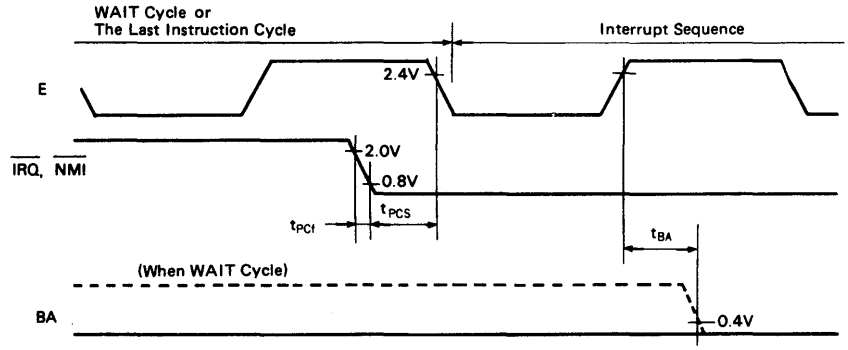


Figure 7 $\overline{IRQ}$ and $\overline{NMI}$ Interrupt Timing

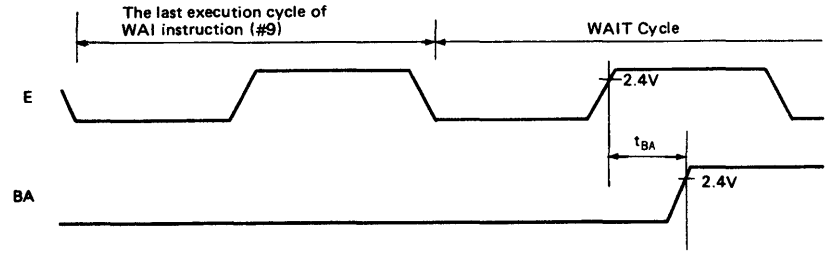


Figure 8 WAI Instruction and BA Timing

■ MPU REGISTERS

A general block diagram of the HD6802 is shown in Fig. 9. As shown, the number and configuration of the registers are the same as for the HD6800. The 128 x 8 bit RAM has been added to the basic MPU. The first 32 bytes may be operated in a low power mode via a V_{CC} standby. These 32 bytes can be retained during power-up and power-down conditions via the RE signal.

The MPU has three 16-bit registers and three 8-bit registers available for use by the programmer (Fig. 10).

- **Program Counter (PC)**

The program counter is a two byte (16-bit) register that points to the current program address.

- **Stack Pointer (SP)**

The stack pointer is a two byte (16-bit) register that contains the address of the next available location in an external push-down/pop-up stack. This stack is normally a random access Read/Write memory that may have any location (address) that is convenient. In those applications that require storage of information in the stack when power is lost, the stack must be non-volatile.

- **Index Register (IX)**

The index register is a two byte register that is used to store data or a sixteen bit memory address for the Indexed mode of memory addressing.

- **Accumulators (ACCA, ACCB)**

The MPU contains two 8-bit accumulators that are used to hold operands and results from an arithmetic logic unit(ALU).

- **Condition Code Register (CCR)**

The condition code register indicates the results of an Arithmetic Logic Unit operation: Negative(N), Zero(Z), Overflow(V), Carry from bit7(C), and half carry from bit3(H). These bits of the Condition Code Register are used as testable conditions for the conditional branch instructions. Bit 4 is the interrupt mask bit(I). The used bits of the Condition Code Register (B6 and B7) are ones.

Fig. 11 shows the order of saving the microprocessor status within the stack.

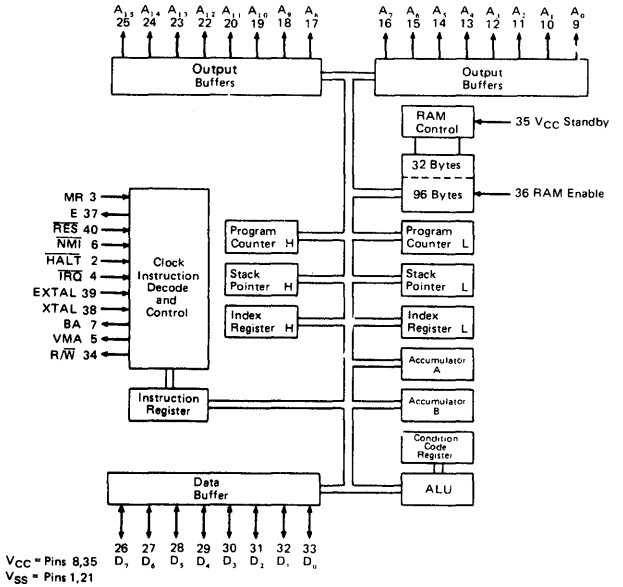


Figure 9 Expanded Block Diagram

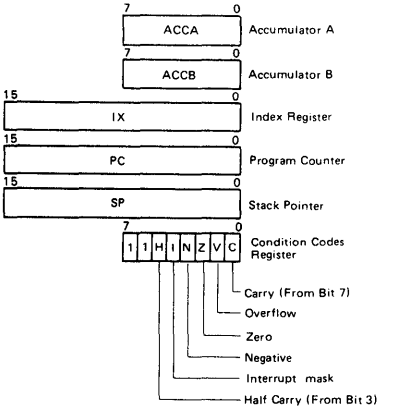


Figure 10 Programming Model of The Microprocessing Unit

SP = Stack Pointer
CC = Condition Codes (Also called the Processor Status Byte)
ACCB = Accumulator B
ACCA = Accumulator A
IXH = Index Register, Higher Order 8 Bits
IXL = Index Register, Lower Order 8 Bits
PCH = Program Counter, Higher Order 8 Bits
PCL = Program Counter, Lower Order 8 Bits

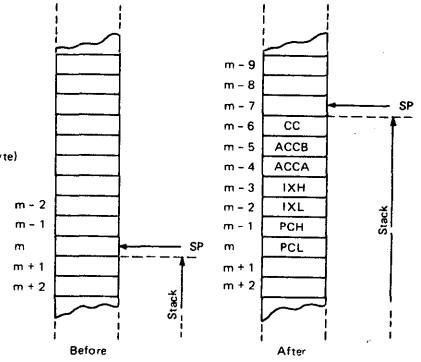


Figure 11 Saving The Status of The Microprocessor in The Stack

■ HD6802 MPU SIGNAL DESCRIPTION

Proper operation of the MPU requires that certain control and timing signals be provided to accomplish specific functions and that other signal lines be monitored to determine the state of the processor. These control and timing signals for the HD6802 are similar to those of the HD6800 except that TSC, DBE, ϕ_1, ϕ_2 input, and two unused pins have been eliminated, and the following signal and timing lines have been added.

RAM Enable (RE)
Crystal Connections EXTAL and XTAL
Memory Ready(MR)
V_{CC} Standby
Enable ϕ_2 Output(E)

The following is a summary of the HD6802 MPU signals:

- **Address Bus ($A_0 \sim A_{15}$)**

Sixteen pins are used for the address bus. The outputs are capable of driving one standard TTL load and 90pF.

- **Data Bus ($D_0 \sim D_7$)**

Eight pins are used for the data bus. It is bidirectional, transferring data to and from the memory and peripheral devices. It also has three-state output buffers capable of driving one standard TTL load and 130pF.

Data Bus will be in the output mode when the internal RAM is accessed. This prohibits external data entering the MPU. It should be noted that the internal RAM is fully decoded from $0000 to $007F. External RAM at $0000 to $007F must be disabled when internal RAM is accessed.

- **HALT**

When this input is in the "Low" state, all activity in the machine will be halted: This input is level sensitive.

In the halt mode, the machine will stop at the end of an instruction. Bus Available will be at a "High" state. Valid Memory Address will be at a "Low" state. The address bus will display the address of the next instruction.

To insure single instruction operation, transition of the $\overline{HALT}$ line must not occur during the last 250ns of E and the $\overline{HALT}$ line must go "High" for one Clock cycle.

$\overline{HALT}$ should be tied "High" if not used. This is good engineering design practice in general and necessary to insure proper operation of the part.

- **Read/Write (R/$\overline{W}$)**

This TTL compatible output signals the peripherals and memory devices whether the MPU is in a Read ("High") or Write ("Low") state. The normal standby state of this signal is Read ("High"). When the processor is halted, it will be in the logical one state ("High").

This output is capable of driving one standard TTL load and 90pF.

- **Valid Memory Address (VMA)**

This output indicates to peripheral devices that there is a valid address on the address bus. In normal operation, this signal should be utilized for enabling peripheral interfaces such as the PIA and ACIA. This signal is not three-state. One standard TTL load and 90pF may be directly driven by this active high signal.

- **Bus Available (BA)**

The Bus Available signal will normally be in the "Low" state. When activated, it will go to the "High" state indicating that the microprocessor has stopped and that the address bus is available (but not in a three-state condition). This will occur if the $\overline{HALT}$ line is in the "Low" state or the processor is in the wait state as a result of the execution of a WAI instruction. At such time, all three-state output drivers will go to their off state and other outputs to their normally inactive level.

The processor is removed from the wait state by the occurrence of a maskable (mask bit I=0) or nonmaskable interrupt. This output is capable of driving one standard TTL load and 30pF.

- **Interrupt Request ($\overline{IRQ}$)**

This level sensitive input requests that an interrupt sequence be generated within the machine. The processor will wait, until it completes the current instruction that is being executed before it recognizes the request. At that time, if the interrupt mask bit in the Condition Code Register is not set, the machine will begin an interrupt sequence. The index Register, Program Counter, Accumulators, and Condition Code Register are stored away on the stack. Next the MPU will respond to the interrupt request by setting the interrupt mask bit high so that no further interrupts may occur. At the end of the cycle, a 16-bit address will be loaded that points to a vectoring address which is located in memory locations FFF8 and FFF9. An address loaded at these locations causes the MPU to branch to an interrupt routine in memory.

The $\overline{HALT}$ line must be in the "High" state for interrupts to be serviced. Interrupts will be latched internally while $\overline{HALT}$ is "Low".

A 3kΩ external register to V_{CC} should be used for wire-OR and optimum control of interrupts.

- **Reset ($\overline{RES}$)**

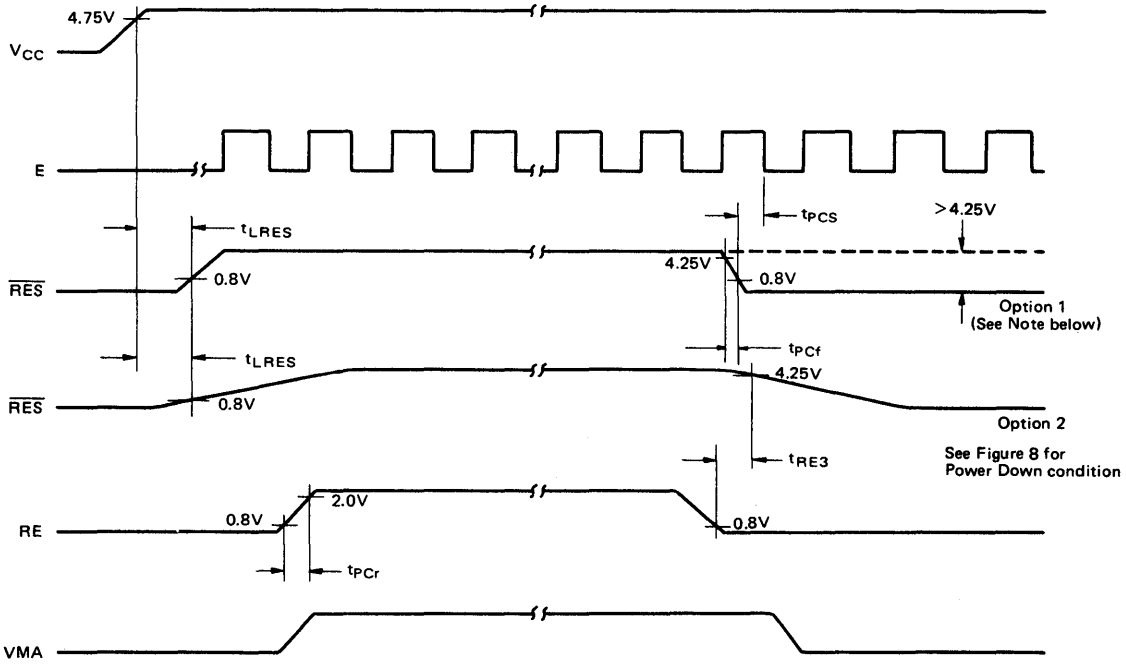
This input is used to reset and start the MPU from a power-down condition, resulting from a power failure or an initial start-up of the processor. When this line is "Low", the MPU is inactive and the information in the registers will be lost. If a "High" level is detected on the input, this will signal the MPU to begin the restart sequence. This will start execution of a routine to initialize the processor from its reset condition. All the higher order address lines will be forced "High". For the restart, the last two(FFFE, FFFF) locations in memory will be used to load the program that is addressed by the program counter. During the restart routine, the interrupt mask bit is set and must be reset before the MPU can be interrupted by $\overline{IRQ}$. Power-up and reset timing and power-down sequences are shown in Fig. 12 and Fig. 13 respectively.

- **Non-Maskable Interrupt ($\overline{NMI}$)**

A low-going edge on this input requests that a non-mask-interrupt sequence be generated within the processor. As with the $\overline{IRQ}$ signal, the processor will complete the current instruction that is being executed before it recognizes the $\overline{NMI}$ signal. The interrupt mask bit in the Condition Code Register has no effect on $\overline{NMI}$.

The Index Register, Program Counter, Accumulators, and Condition Code Register are stored away on the stack. At the end of the cycle, a 16-bit address will be loaded that points to a vectoring address which is located in memory locations FFFC and FFFD. An address loaded at these locations causes the MPU to branch to a non-maskable interrupt routine in memory. A 3kΩ external resistor to V_{CC} should be used for wire-OR and optimum control of interrupts.

Inputs $\overline{IRQ}$ and $\overline{NMI}$ are hardware interrupt lines that are sampled when E is "High" and will start the interrupt routine on a "Low" E following the completion of an instruction. $\overline{IRQ}$ and $\overline{NMI}$ should be tied "High" if not used. This is good engineering design practice in general and necessary to insure proper operation of the part. Fig. 14 is a flowchart describing the major decision paths and interrupt vectors of the microprocessor. Table 1 gives the memory map for interrupt vectors.



(NOTE) If option 1 is chosen, $\overline{RES}$ and RE pins can be tied together.

Figure 12 Power-up and Reset Timing

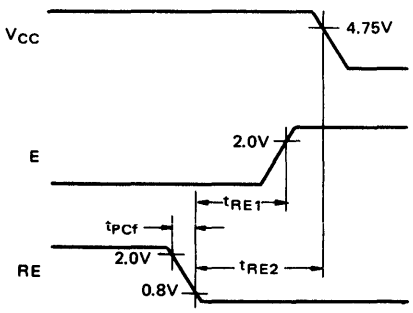


Figure 13 Power-down Sequence

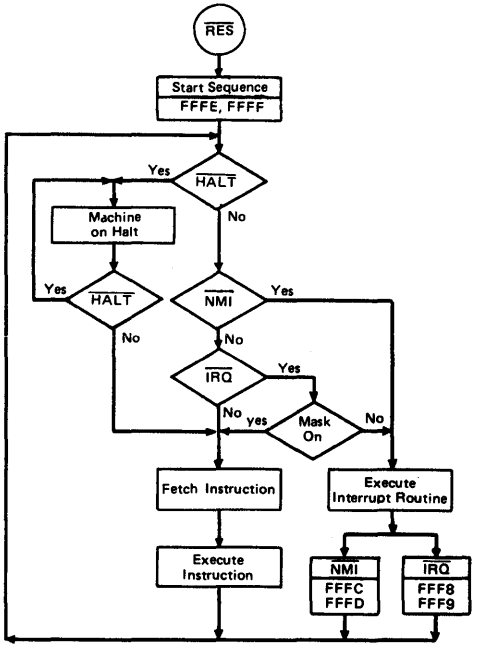


Figure 14 MPU Flow Chart

Table 1 Memory Map for Interrupt Vectors

Vector MS	Vector LS	Description	
FFFE	FFFF	Restart	($\overline{RES}$)
FFFC	FFFD	Non-Maskable Interrupt	($\overline{NMI}$)
FFFA	FFFB	Software Interrupt	(SWI)
FFF8	FFF9	Interrupt Request	($\overline{IRQ}$)

- **RAM Enable (RE)**

A TTL-compatible RAM enable input controls the on-chip RAM of the HD6802. When placed in the "High" state, the on-chip memory is enabled to respond to the MPU controls. In the "Low" state, RAM is disabled. This pin may also be utilized to disable reading and writing the on-chip RAM during a power-down situation. RAM enable must be "Low" three cycles before V_{CC} goes below 4.75V during power-down.

RE should be tied to the correct "High" or "Low" state if not used. This is good engineering design practice in general and necessary to insure proper operation of the part.

- **EXTAL and XTAL**

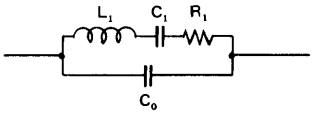
The HD6802 has an internal oscillator that may be crystal controlled. These connections are for a parallel resonant fundamental crystal (AT cut). A divide-by-four circuit has been added to the HD6802 so that a 4MHz crystal may be used in lieu of a 1MHz crystal for a more cost-effective system. Pin39 of the HD6802 may be driven externally by a TTL input signal if a separate clock is required. Pin38 is to be left open in this mode.

An RC network is not directly usable as a frequency source on pins 38 and 39. An RC network type TTL or CMOS oscillator will work well as long as the TTL or CMOS output drives the HD6802.

If an external clock is used, it may not be halted for more than 4.5μs. The HD6802 is a dynamic part except for the internal RAM, and requires the external clock to retain information.

Conditions for Crystal (4 MHz)

- AT Cut Parallel resonant
- C_0 = 7 pF max.
- R_1 = 80Ω max.



Crystal Equivalent Circuit

Recommended Oscillator (4MHz)

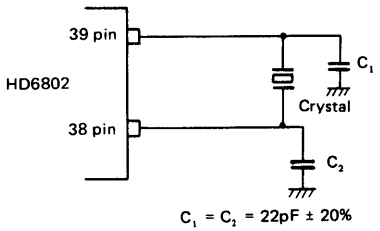


Figure 15 Crystal Oscillator

When using the crystal, see the note for Board Design of the Oscillation Circuit in HD6802.

- **Memory Ready (MR)**

MR is a TTL compatible input control signal which allows stretching of E. When MR is "High", E will be in normal operation. When MR is "Low", E may be stretched integral multiples of half periods, thus allowing interface to slow memories. Memory Ready timing is shown in Fig. 16.

MR should be tied "High" if not used. This is good engineering design practice in general and necessary to insure proper operation of the part. A maximum stretch is 4.5μs.

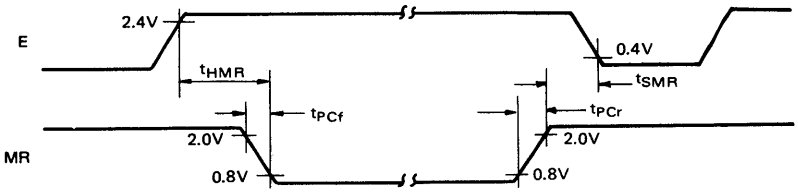


Figure 16 Memory Ready Control Function

- **Enable (E)**

This pin supplies the clock for the MPU and the rest of the system. This is a single phase, TTL compatible clock. This clock may be conditioned by a Memory Ready Signal. This is equivalent to ϕ_2 on the HD6800.

- **V_{CC} Standby**

This pin supplies the dc voltage to the first 32 bytes of RAM as well as the RAM Enable (RE) control logic. Thus retention of data in this portion of the RAM on a power up, power-down, or standby condition is guaranteed at the range of 4.0 V to 5.25 V.

Maximum current drain at 5.25V is 8mA.

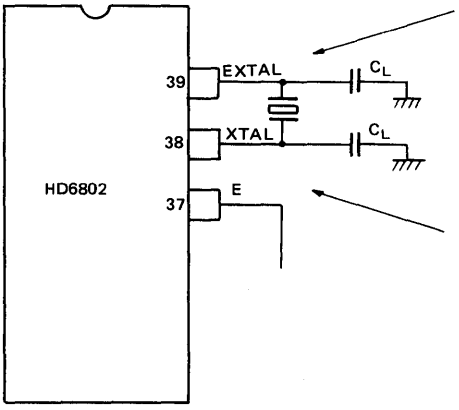
■ MPU INSTRUCTION SET

The HD6802 has a set of 72 different instructions. Included are binary and decimal arithmetic, logical, shift, rotate, load, store, conditional or unconditional branch, interrupt and stack manipulation instructions.

This instruction set is the same as that for the 6800MPU(HD6800 etc.) and is not explained again in this data sheet.

■ NOTE FOR BOARD DESIGN OF THE OSCILLATION CIRCUIT IN HD6802

In designing the board, the following notes should be taken when the crystal oscillator is used.

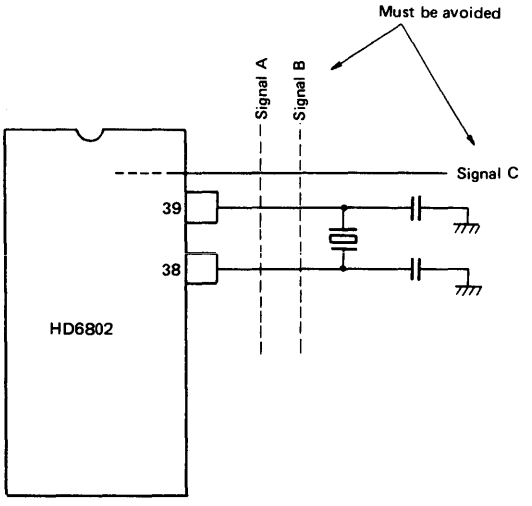


Crystal oscillator and load capacity C_L must be placed near the LSI as much as possible.

Normal oscillation may be disturbed when external noise is induced to pin 38 and 39.

Pin 38 signal line should be wired apart from pin 37 signal line as much as possible. Don't wire them in parallel, or normal oscillation may be disturbed when E signal is feedbacked to XTAL.

The following design must be avoided.



A signal line or a power source line must not cross or go near the oscillation circuit line as shown in the left figure to prevent the induction from these lines and perform the correct oscillation. The resistance among XTAL, EXTAL and other pins should be over 10MΩ.

Figure 17 Note for Board Design of the Oscillation Circuit

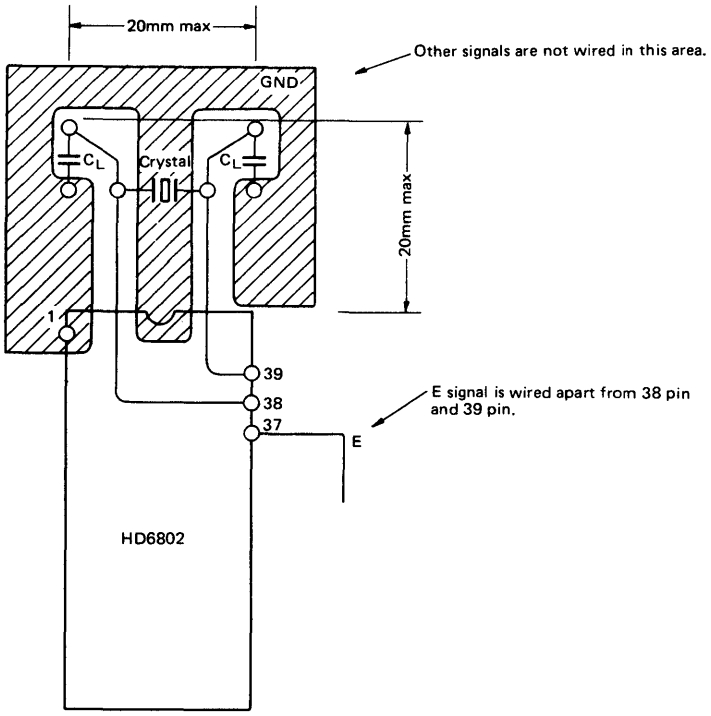


Figure 18 Example of Board Design Using the Crystal Oscillator

HD6802W

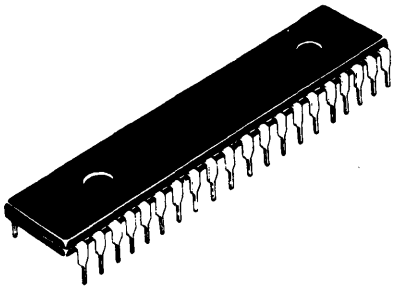
MPU (Microprocessor with Clock and RAM)

HD6802W is the enhanced version of HD6802 which contains MPU, clock and 256 bytes RAM. Internal RAM has been extended from 128 to 256 bytes to increase the capacity of system read/write memory for handling temporary data and manipulating the stack.

The internal RAM is located at hex addresses 0000 to 00FF. The first 32 bytes of RAM, at hex addresses 0000 to 001F, may be retained in a low power mode by utilizing V_{CC} standby, thus facilitating memory retention during a power-down situation.

The HD6802W is completely software compatible with the HD6800 as well as the entire HMCS6800 family of parts. Hence, the HD6802W is expandable to 65k words.

HD6802WP

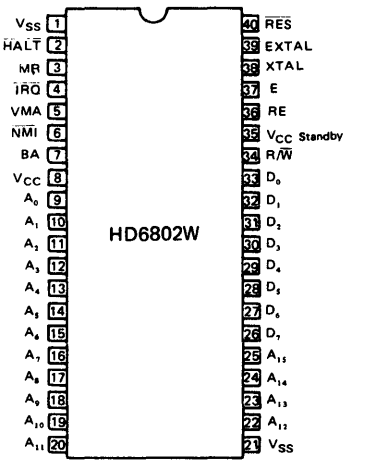


(DP-40)

■ FEATURES

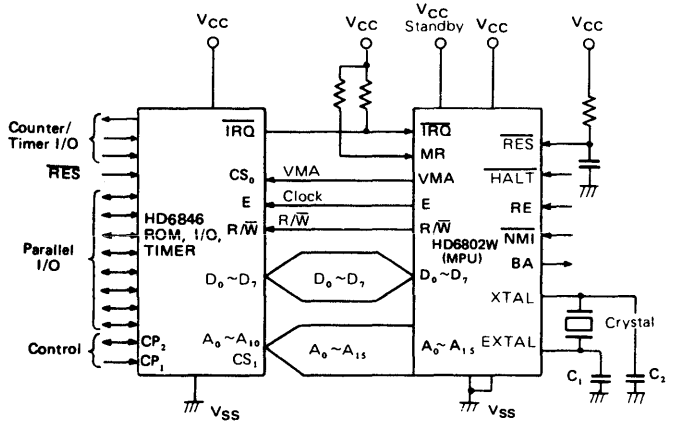
- On-Chip Clock Circuit
- 256 × 8 Bit On-Chip RAM
- 32 Bytes of RAM are Retainable
- Software-Compatible with the HD6800, HD6802
- Expandable to 65k words
- Standard TTL-Compatible Inputs and Outputs
- 8 Bit Word Size
- 16 Bit Memory Addressing
- Interrupt Capability

■ PIN ARRANGEMENT



(Top View)

■ BLOCK DIAGRAM



A expanded block diagram of the HD6802W is shown in Fig. 1. As shown, the number and configuration of the registers are the same as the HD6802 except that the internal RAM has been extended to 256 bytes.

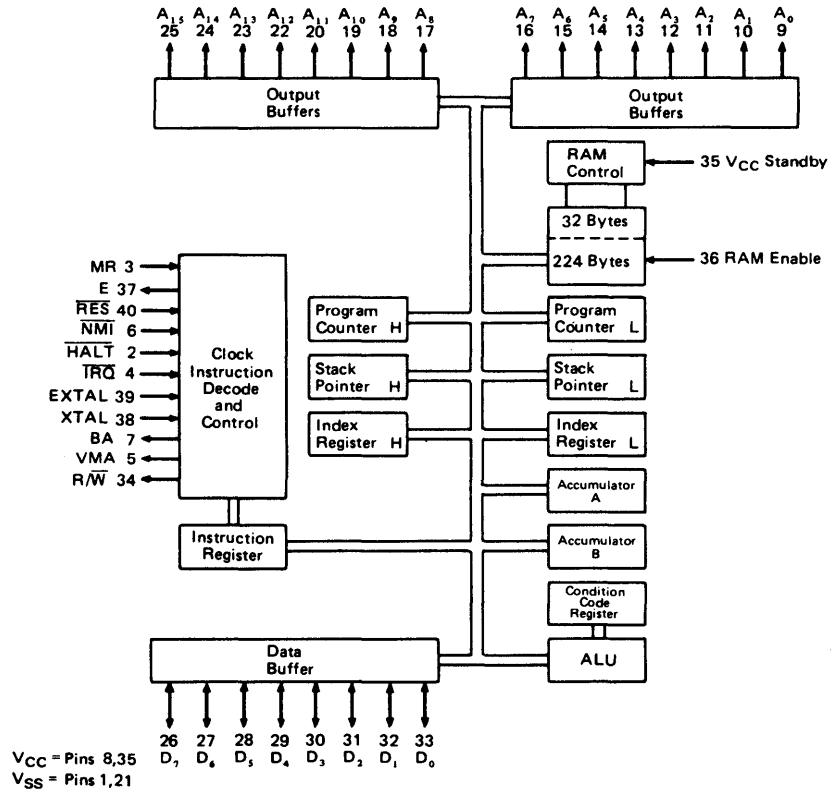


Figure 1 Expanded Block Diagram

Address Map of RAM is shown is Fig. 2.

The HD6802W has 256 bytes of RAM on the chip located at hex addresses 0000 to 00FF. The first 32 bytes of RAM, at hex addresses 0000 to 001F, may be retained in a low power mode by utilizing V_{CC} standby and setting RAM Enable Signal "Low" level, thus facilitating memory retention during a power-down situation.

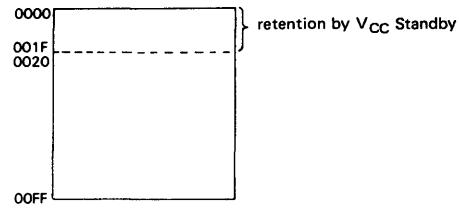


Figure 2 Address Map of HD6802W

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC} * V_{CC} Standby*	-0.3 ~ +7.0	V
Input Voltage	V_{in}*	-0.3 ~ +7.0	V
Operating Temperature	T_{opr}	-20 ~ +75	°C
Storage Temperature	T_{stg}	-55 ~ +150	°C

* With respect to V_{SS} (SYSTEM GND)

(NOTE) Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ RECOMMENDED OPERATING CONDITIONS

Item	Symbol		min	typ	max	Unit
Supply Voltage	V_{CC} *		4.75	5.0	5.25	V
	V_{CC} Standby*		4.0			
Input Voltage	V_{IL}*		-0.3	–	0.8	V
	V_{IH}*	Except $\overline{RES}$	2.0	–	V_{CC}	V
		$\overline{RES}$	V_{CC} -0.75	–	V_{CC}	
Operation Temperature	T_{opr}		-20	25	75	°C

* With respect to V_{SS} (SYSTEM GND)

■ ELECTRICAL CHARACTERISTICS

• DC CHARACTERISTICS (V_{CC}=5.0V±5%, V_{CC} Standby=5.0V±5%, V_{SS}=0V, Ta=-20~+75°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ*	max	Unit
Input "High" Voltage	Except $\overline{RES}$	V_{IH}		2.0	–	V_{CC}	V
	$\overline{RES}$			V_{CC}-0.75	–	V_{CC}	
Input "Low" Voltage	Except $\overline{RES}$	V_{IL}**		-0.3	–	0.8	V
	$\overline{RES}$			-0.3	–	0.8	
Output "High" Voltage	D_0~D_7, E	V_{OH}	I_{OH} = -205μA	2.4	–	–	V
	A_0~A_{15}, R/$\overline{W}$, VMA		I_{OH} = -145μA	2.4	–	–	
	BA		I_{OH} = -100μA	2.4	–	–	
Output "Low" Voltage		V_{OL}	I_{OL} = 1.6mA	–	–	0.4	V
Three State (Off State) Input Current	D_0~D_7	I_{TSI}	V_{in} = 0.4~2.4V	-10	–	10	μA
Input Leakage Current	Except D_0~D_7	I_{in} ***	V_{in} = 0~5.25V	-2.5	–	2.5	μA
Power Dissipation		P_D ****		–	0.7	1.2	W
Input Capacitance	D_0~D_7	C_{in}	V_{in}=0V, T_a=25°C, f=1.0MHz	–	10	12.5	pF
	Except D_0~D_7			–	6.5	10	
Output Capacitance	A_0~A_{15}, R/$\overline{W}$, BA, VMA	C_{out}	V_{in}=0V, T_a=25°C, f=1.0MHz	–	–	12	pF

* T_a=25° C, V_{CC} =5V
** As $\overline{RES}$ input has histeresis character, applied voltage up to 2.4V is regarded as "Low" level when it goes up from 0V.
*** Does not include EXTAL and XTAL, which are crystal inputs.
**** In power-down mode, maximum power dissipation is less than 42mW.

● **AC CHARACTERISTICS (V_{CC}=5.0V±5%, V_{CC} Standby=5.0V±5%, V_{SS}=0V, Ta=−20~+75°C, unless otherwise noted.)**

1. CLOCK TIMING CHARACTERISTICS

Item		Symbol	Test Condition	min	typ	max	Unit
Frequency of Operation	Input Clock ÷ 4	f		0.1	–	1.0	MHz
	Crystal Frequency	f_{XTAL}		1.0	–	4.0	
Cycle Time		t_{cyc}	Fig. 4, Fig. 5	1.0	–	10	μs
Clock Pulse Width	"High" Level	$PW_{\phi H}$	at 2.4V (Fig. 4, Fig. 5)	450	–	4500	ns
	"Low" Level	$PW_{\phi L}$	at 0.8V (Fig. 4, Fig. 5)				
Clock Fall Time		t_{ϕ}	0.8V ~ 2.4V (Fig.4,Fig.5)	–	–	25	ns

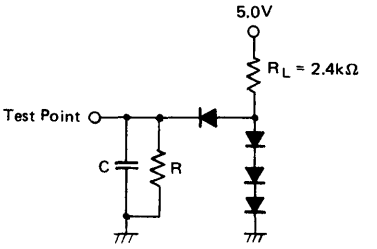
2. READ/WRITE TIMING

Item	Symbol	Test Condition	min	typ*	max	Unit
Address Delay	t_{AD}	Fig. 4, Fig. 5, Fig. 8	–	–	270	ns
Peripheral Read Access Time	t_{acc}	Fig. 4	–	–	530	ns
Data Setup Time (Read)	t_{DSR}	Fig. 4	100	–	–	ns
Input Data Hold Time	t_H	Fig. 4	10	–	–	ns
Output Data Hold Time	t_H	Fig. 5	20	–	–	ns
Address Hold Time (Address, R/$\overline{W}$, VMA)	t_{AH}	Fig. 4, Fig. 5	10	–	–	ns
Data Delay Time (Write)	t_{DDW}	Fig. 5	–	165	225	ns
Bus Available Delay	t_{BA}	Fig. 6, Fig. 7, Fig. 9, Fig. 10	–	–	250	ns
Processor Controls Processor Control Setup Time	t_{PCS}	Fig. 6 ~ Fig. 9, Fig. 11	200	–	–	ns
Processor Control Rise and Fall Time (Measured at 0.8V and 2.0V)	t_{PCr}, t_{PCf}	Fig. 6 ~ Fig. 9, Fig. 11, Fig. 12, Fig. 14	–	–	100	ns

* Ta = 25°C, V_{CC} = 5V

3. POWER DOWN SEQUENCE TIMING, POWER UP RESET TIMING AND MEMORY READY TIMING

Item	Symbol	Test Condition	min	typ	max	Unit
RAM Enable Reset Time (1)	t_{RE1}	Fig. 12	150	–	–	ns
RAM Enable Reset Time (2)	t_{RE2}	Fig. 12	E-3 cycles	–	–	
Reset Release Time	t_{LRES}	Fig. 11	20	–	–	ms
RAM Enable Reset Time (3)	t_{RE3}	Fig. 11	0	–	–	ns
Memory Ready Setup Time	t_{SMR}	Fig. 14	300	–	–	ns
Memory Ready Hold Time	t_{HMR}	Fig. 14	0	–	200	ns



C = 130pF for D_0~D_7, E
= 90pF for A_0~A_{15}, R/$\overline{W}$, and VMA
= 30pF for BA
R = 11kΩ for D_0~D_7, E
= 16kΩ for A_0~A_{15}, R/$\overline{W}$, and VMA
= 24kΩ for BA
C includes stray Capacitance.
All diodes are 1S2074Ⓗ or equivalent.

Figure 3 Bus Timing Test Load

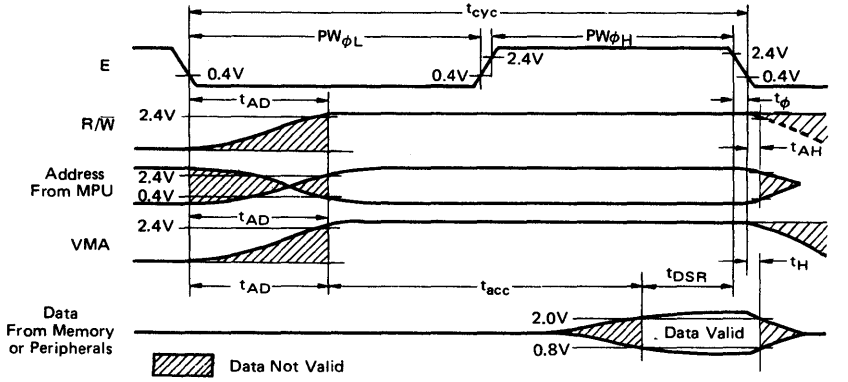


Figure 4 Read Data from Memory or Peripherals

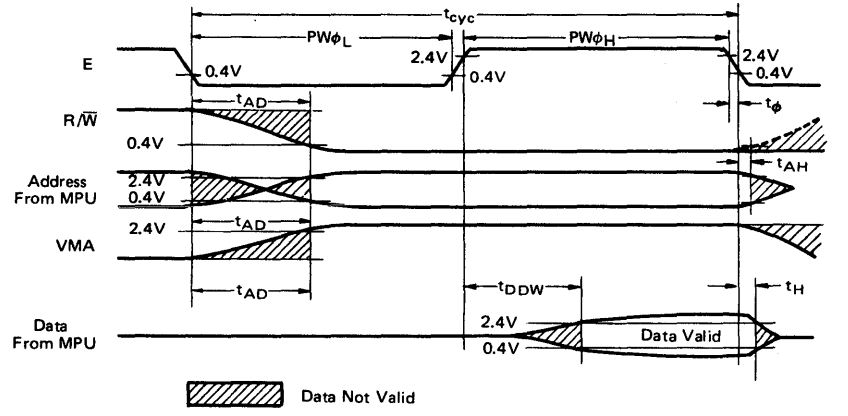


Figure 5 Write Data in Memory or Peripherals

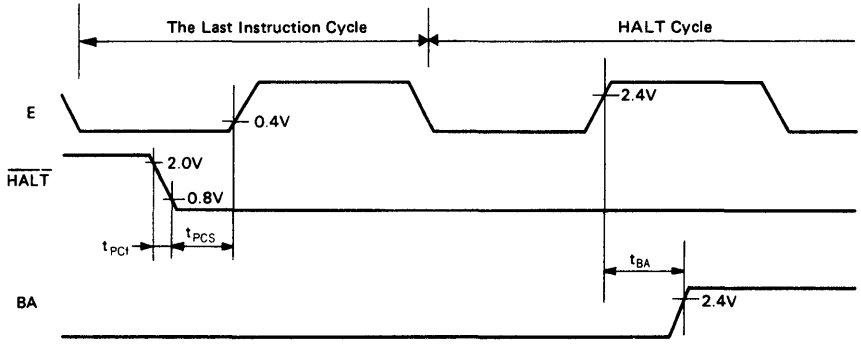


Figure 6 Timing of $\overline{\text{HALT}}$ and BA

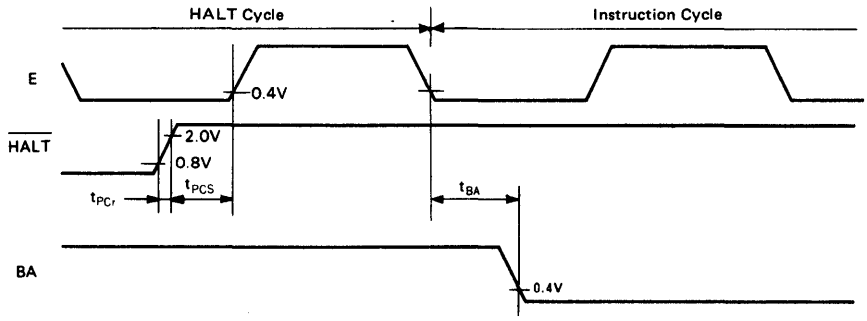


Figure 7 Timing of $\overline{\text{HALT}}$ and BA

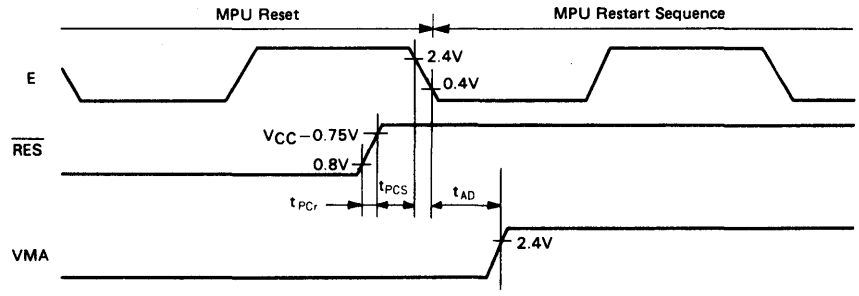


Figure 8 $\overline{\text{RES}}$ and MPU Restart Sequence

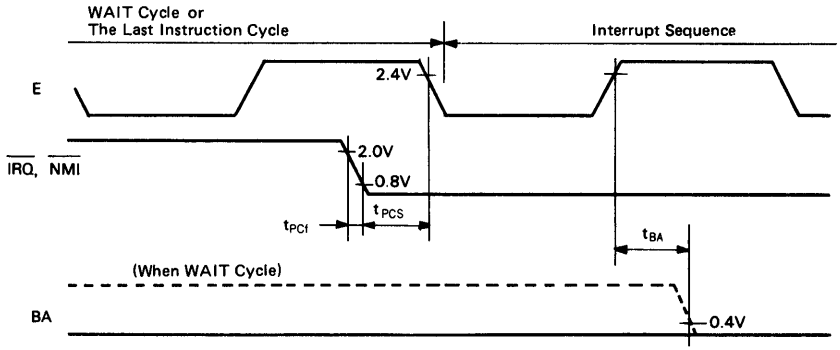


Figure 9 $\overline{IRQ}$ and $\overline{NMI}$ Interrupt Timing

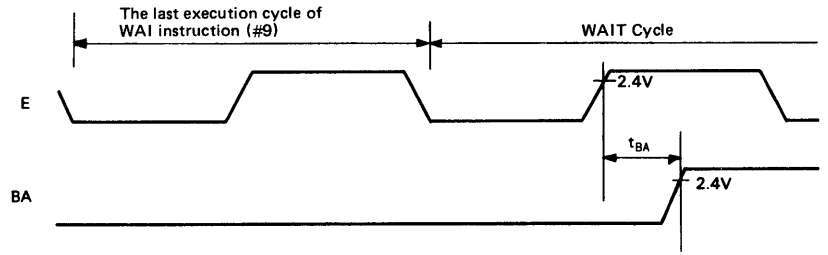


Figure 10 WAI Instruction and BA Timing

■ HD6802W MPU SIGNAL DESCRIPTION

- **Address Bus ($A_0 \sim A_{15}$)**

Sixteen pins are used for the address bus. The outputs are capable of driving one standard TTL load and 90pF.

- **Data Bus ($D_0 \sim D_7$)**

Eight pins are used for the data bus. It is bidirectional, transferring data to and from the memory and peripheral devices. It also has three-state output buffers capable of driving one standard TTL load and 130pF.

Data Bus will be in the output mode when the internal RAM is accessed. This prohibits external data entering the MPU. It should be noted that the internal RAM is fully decoded from $0000 to $00FF. External RAM at $0000 to $00FF must be disabled when internal RAM is accessed.

- **$\overline{HALT}$**

When this input is in the "Low" state, all activity in the machine will be halted: This input is level sensitive.

In the halt mode, the machine will stop at the end of an instruction. Bus Available will be at a "High" state. Valid Memory Address will be at a "Low" state. The address bus will display the address of the next instruction.

To insure single instruction operation, transition of the $\overline{HALT}$ line must not occur during the last t_{PCS} of E and the $\overline{HALT}$ line must go "High" for one Clock cycle.

$\overline{HALT}$ should be tied "High" if not used. This is good engineering design practice in general and necessary to insure proper operation of the part.

- **Read/Write (R/$\overline{W}$)**

This TTL compatible output signals the peripherals and memory devices whether the MPU is in a Read ("High") or Write ("Low") state. The normal standby state of this signal is Read ("High"). When the processor is halted, it will be in the logical one state ("High").

This output is capable of driving one standard TTL load and 90pF.

- **Valid Memory Address (VMA)**

This output indicates to peripheral devices that there is a valid address on the address bus. In normal operation, this signal should be utilized for enabling peripheral interfaces such as the PIA and ACIA. This signal is not three-state. One standard TTL load and 90pF may be directly driven by this active high signal.

- **Bus Available (BA)**

The Bus Available signal will normally be in the "Low" state. When activated, it will go to the "High" state indicating that the microprocessor has stopped and that the address bus is available (but not in a three-state condition). This will occur if the $\overline{HALT}$ line is in the "Low" state or the processor is in the wait state as a result of the execution of a WAI instruction. At such time, all three-state output drivers will go to their off state and other outputs to their normally inactive level.

The processor is removed from the wait state by the occurrence of a maskable (mask bit I=0) or nonmaskable interrupt. This output is capable of driving one standard TTL load and 30pF.

- **Interrupt Request ($\overline{IRQ}$)**

This level sensitive input requests that an interrupt sequence be generated within the machine. The processor will wait, until it completes the current instruction that is being executed before it recognizes the request. At that time, if the interrupt mask bit in the Condition Code Register is not set, the machine will begin an interrupt sequence. The index Register, Program Counter, Accumulators, and Condition Code Register are stored away on the stack. Next the MPU will respond to the interrupt request by setting the interrupt mask bit high so that no further interrupts may occur. At the end of the cycle, a 16-bit address will be loaded that points to a vectoring address which is located in memory locations FFF8 and FFF9. An address loaded at these locations causes the MPU to branch to an interrupt routine in memory.

The $\overline{HALT}$ line must be in the "High" state for interrupts to be serviced. Interrupts will be latched internally while $\overline{HALT}$ is "Low".

A 3kΩ external register to V_{CC} should be used for wire-OR and optimum control of interrupts.

- **Reset ($\overline{RES}$)**

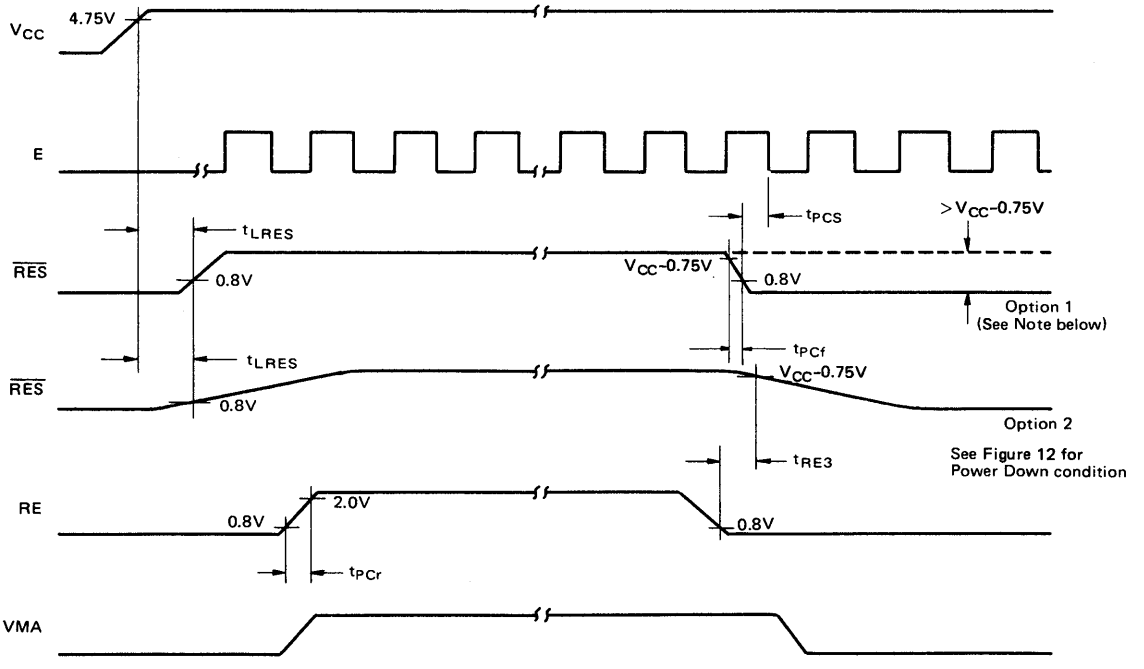
This input is used to reset and start the MPU from a power-down condition, resulting from a power failure or an initial start-up of the processor. When this line is "Low", the MPU is inactive and the information in the registers will be lost. If a "High" level is detected on the input, this will signal the MPU to begin the restart sequence. This will start execution of a routine to initialize the processor from its reset condition. All the higher order address lines will be forced "High". For the restart, the last two(FFFE, FFFF) locations in memory will be used to load the program that is addressed by the program counter. During the restart routine, the interrupt mask bit is set and must be reset before the MPU can be interrupted by $\overline{IRQ}$. Power-up and reset timing and power-down sequences are shown in Fig. 11 and Fig. 12 respectively.

- **Non-Maskable Interrupt ($\overline{NMI}$)**

A low-going edge on this input requests that a non-mask-interrupt sequence be generated within the processor. As with the $\overline{IRQ}$ signal, the processor will complete the current instruction that is being executed before it recognizes the $\overline{NMI}$ signal. The interrupt mask bit in the Condition Code Register has no effect on $\overline{NMI}$.

The Index Register, Program Counter, Accumulators, and Condition Code Register are stored away on the stack. At the end of the cycle, a 16-bit address will be loaded that points to a vectoring address which is located in memory locations FFFC and FFFD. An address loaded at these locations causes the MPU to branch to a non-maskable interrupt routine in memory. A 3kΩ external resistor to V_{CC} should be used for wire-OR and optimum control of interrupts.

Inputs $\overline{IRQ}$ and $\overline{NMI}$ are hardware interrupt lines that are sampled when E is "High" and will start the interrupt routine on a "Low" E following the completion of an instruction. $\overline{IRQ}$ and $\overline{NMI}$ should be tied "High" if not used. This is good engineering design practice in general and necessary to insure proper operation of the part. Fig. 13 is a flowchart describing the major decision paths and interrupt vectors of the microprocessor. Table 1 gives the memory map for interrupt vectors.



(NOTE) If option 1 is chosen, $\overline{RES}$ and RE pins can be tied together.

Figure 11 Power-up and Reset Timing

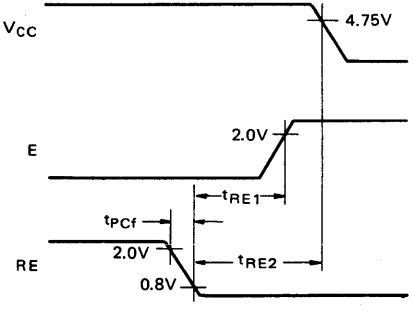


Figure 12 Power-down Sequence

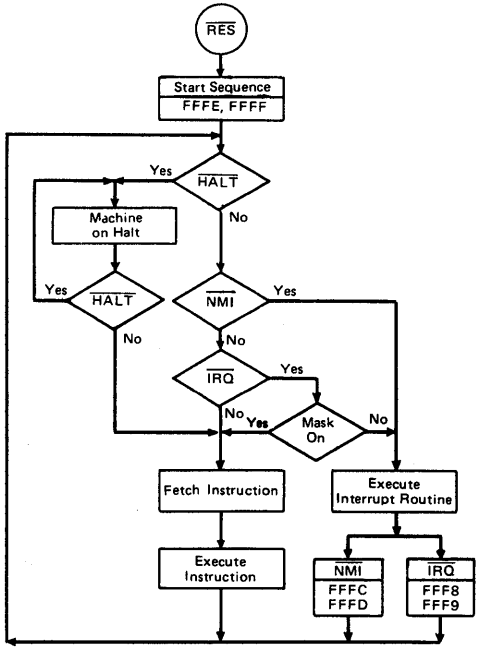


Figure 13 MPU Flow Chart

Table 1 Memory Map for Interrupt Vectors

Vector MS	Vector LS	Description
FFFE	FFFF	Restart ($\overline{RES}$)
FFFC	FFFD	Non-Maskable Interrupt ($\overline{NMI}$)
FFFA	FFFB	Software Interrupt (SWI)
FFF8	FFF9	Interrupt Request ($\overline{IRQ}$)

- **RAM Enable (RE)**

A TTL-compatible RAM enable input controls the on-chip RAM of the HD6802W. When placed in the "High" state, the on-chip memory is enabled to respond to the MPU controls. In the "Low" state, RAM is disabled. This pin may also be utilized to disable reading and writing the on-chip RAM during a power-down situation. RAM enable must be "Low" three cycles before V_{CC} goes below 4.75V during power-down.

RE should be tied to the correct "High" or "Low" state if not used. This is good engineering design practice in general and necessary to insure proper operation of the part.

- **EXTAL and XTAL**

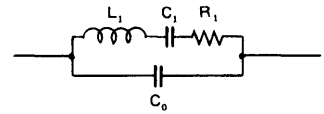
The HD6802W has an internal oscillator that may be crystal controlled. These connections are for a parallel resonant fundamental crystal (AT cut). A divide-by-four circuit has been added to the HD6802W so that a 4MHz crystal may be used in lieu of a 1MHz crystal for a more cost-effective system. Pin39 of the HD6802W may be driven externally by a TTL input signal if a separate clock is required. Pin38 is to be left open in this mode.

An RC network is not directly usable as a frequency source on pins 38 and 39. An RC network type TTL or CMOS oscillator will work well as long as the TTL or CMOS output drives the HD6802W.

If an external clock is used, it may not be halted for more than 4.5μs. The HD6802W is a dynamic part except for the internal RAM, and requires the external clock to retain information.

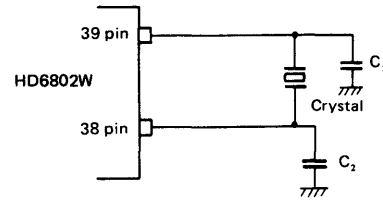
Conditions for Crystal (4 MHz)

- AT Cut Parallel resonant
- C_0 = 7 pF max.
- R_1 = 80Ω max.



Crystal Equivalent Circuit

Recommended Oscillator (4MHz)



$C_1 = C_2$ = 22pF ±20%

When using the crystal, see the note for Board Design of the Oscillation Circuit in HD6802W.

- **Memory Ready (MR)**

MR is a TTL compatible input control signal which allows stretching of E. When MR is "High", E will be in normal operation. When MR is "Low", E may be stretched integral multiples of half periods, thus allowing interface to slow memories. Memory Ready timing is shown in Fig. 14.

MR should be tied "High" if not used. This is good engineering design practice in general and necessary to insure proper operation of the part. A maximum stretch is 4.5μs.

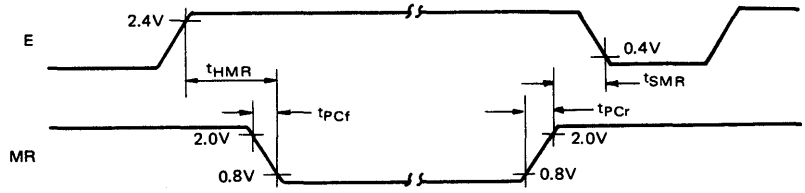


Figure 14 Memory Ready Control Function

- **Enable (E)**

This pin supplies the clock for the MPU and the rest of the system. This is a single phase, TTL compatible clock. This clock may be conditioned by a Memory Ready Signal. This is equivalent to ϕ_2 on the HD6800.

- **V_{CC} Standby**

This pin supplies the dc voltage to the first 32 bytes of RAM as well as the RAM Enable (RE) control logic. Thus retention of data in this portion of the RAM on a power-up, power-down, or standby condition is guaranteed at the range of 4.0 V to 5.25 V.

Maximum current drain at 5.25V is 8mA.

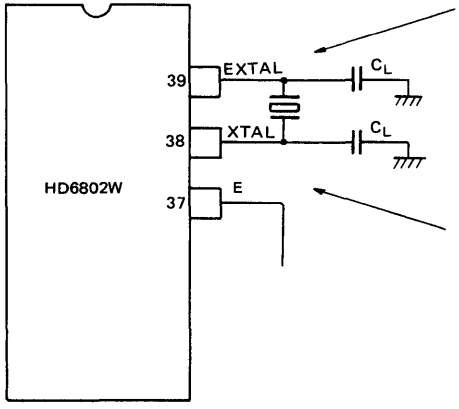
■ MPU INSTRUCTION SET

The HD6802W has a set of 72 different instructions. Included are binary and decimal arithmetic, logical, shift, rotate, load, store, conditional or unconditional branch, interrupt and stack manipulation instructions.

This instruction set is the same as that for the 6800MPU (HD6800 etc.) and is not explained again in this data sheet.

■ NOTE FOR BOARD DESIGN OF THE OSCILLATION CIRCUIT IN HD6802W

In designing the board, the following notes should be taken when the crystal oscillator is used.

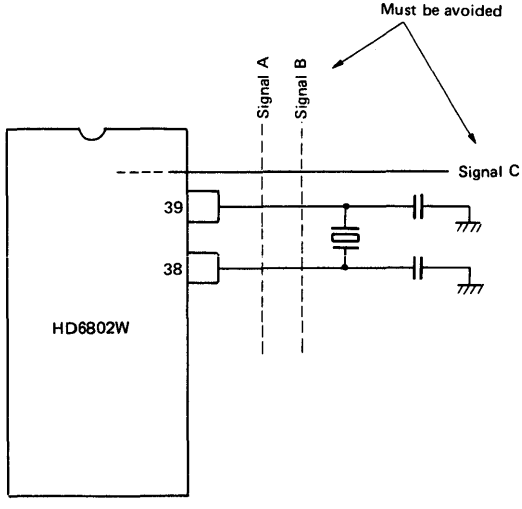


Crystal oscillator and load capacity C_L must be placed near the LSI as much as possible.

Normal oscillation may be disturbed when external noise is induced to pin 38 and 39.

Pin 38 signal line should be wired apart from pin 37 signal line as much as possible. Don't wire them in parallel, or normal oscillation may be disturbed when E signal is feedbacked to XTAL.

The following design must be avoided.



A signal line or a power source line must not cross or go near the oscillation circuit line as shown in the left figure to prevent the induction from these lines and perform the correct oscillation. The resistance among XTAL, EXTAL and other pins should be over 10MΩ.

Figure 15 Note for Board Design of the Oscillation Circuit

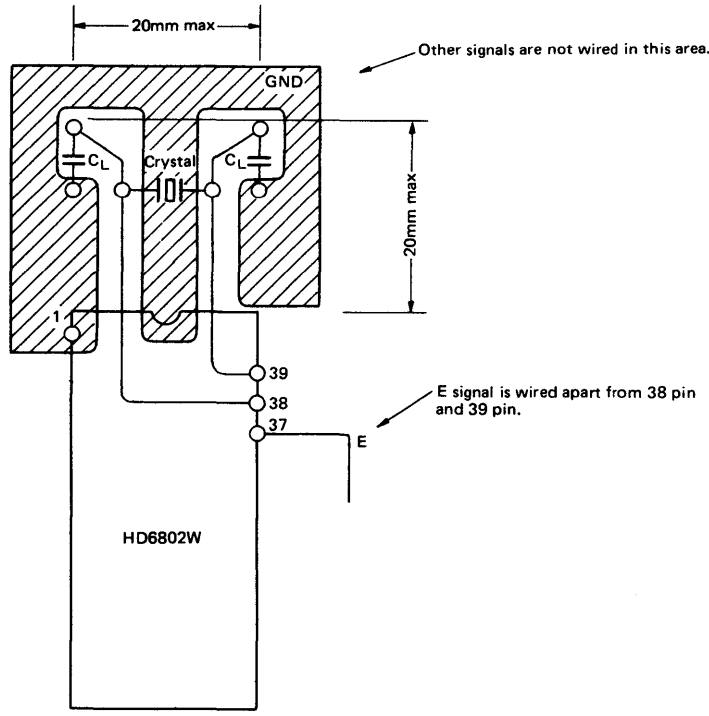


Figure 16 Example of Board Design Using the Crystal Oscillator

HD6809, HD68A09, HD68B09— MPU (Micro Processing Unit)

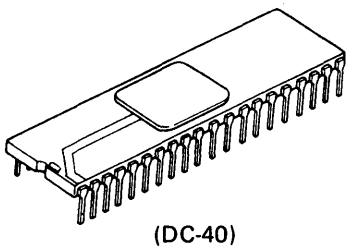
The HD6809 is a revolutionary high performance 8-bit microprocessor which supports modern programming techniques such as position independence, reentrancy, and modular programming.

This third-generation addition to the HMCS6800 family has major architectural improvements which include additional registers, instructions and addressing modes.

The basic instructions of any computer are greatly enhanced by the presence of powerful addressing modes. The HD6809 has the most complete set of addressing modes available on any 8-bit microprocessor today.

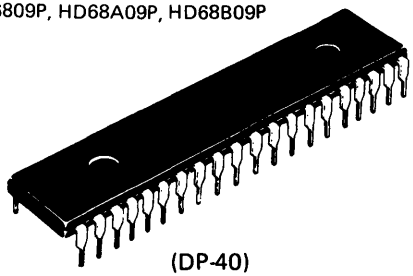
The HD6809 has hardware and software features which make it an ideal processor for higher level language execution or standard controller applications. The specification of the HD68B09 is preliminary.

HD6809, HD68A09, HD68B09



(DC-40)

HD6809P, HD68A09P, HD68B09P



(DP-40)

HD6800 COMPATIBLE

- Hardware – Interfaces with All HMCS6800 Peripherals
- Software – Upward Source Code Compatible Instruction Set and Addressing Modes

■ ARCHITECTURAL FEATURES

- Two 16-bit Index Registers
- Two 16-bit Indexable Stack Pointers
- Two 8-bit Accumulators can be Concatenated to Form One 16-Bit Accumulator
- Direct Page Register Allows Direct Addressing Throughout Memory

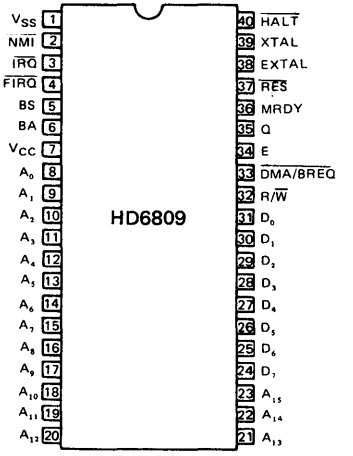
■ HARDWARE FEATURES

- On Chip Oscillator
- $\overline{\text{DMA/BREQ}}$ Allows DMA Operation or Memory Refresh
- Fast Interrupt Request Input Stacks Only Condition Code Register and Program Counter
- MRDY Input Extends Data Access Times for Use With Slow Memory
- Interrupt Acknowledge Output Allows Vectoring By Devices
- SYNC Acknowledge Output Allows for Synchronization to External Event
- Single Bus-Cycle RESET
- Single 5-Volt Supply Operation
- NMI Blocked After RESET Until After First Load of Stack Pointer
- Early Address Valid Allows Use With Slower Memories
- Early Write-Data for Dynamic Memories
- Compatible with MC6809, MC68A09 and MC68B09

■ SOFTWARE FEATURES

- 10 Addressing Modes
  - HMCS6800 Upward Compatible Addressing Modes
  - Direct Addressing Anywhere in Memory Map
  - Long Relative Branches
  - Program Counter Relative
  - True Indirect Addressing
  - Expanded Indexed Addressing:

■ PIN ARRANGEMENT

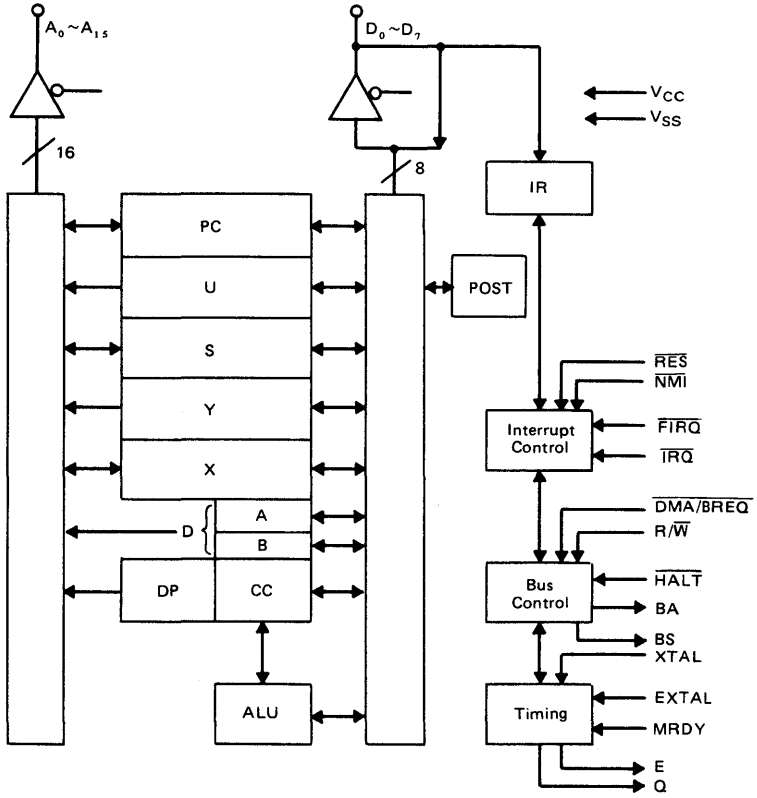


(Top View)

0, 5, 8, or 16-bit Constant Offsets
8, or 16-bit Accumulator Offsets
Auto-Increment/Decrement by 1 or 2

- Improved Stack Manipulation
- 1464 Instructions with Unique Addressing Modes
- 8 x 8 Unsigned Multiply
- 16-bit Arithmetic
- Transfer/Exchange All Registers
- Push/Pull Any Registers or Any Set of Registers
- Load Effective Address

■ BLOCK DIAGRAM



■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC}*	-0.3 ~ +7.0	V
Input Voltage	V_{in}*	-0.3 ~ +7.0	V
Operating Temperature	T_{opr}	-20 ~ +75	°C
Storage Temperature	T_{stg}	-55 ~ +150	°C

* With respect to V_{SS} (SYSTEM GND)

(NOTE) Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ RECOMMENDED OPERATING CONDITIONS

Item	Symbol		min	typ	max	Unit
Supply Voltage	V_{CC}*		4.75	5.0	5.25	V
Input Voltage	V_{IL}*		-0.3	–	0.8	V
	V_{IH}*	Logic	2.2	–	V_{cc}*	V
		$\overline{RES}$	4.0	–	V_{cc}*	
Operating Temperature	T_{opr}		-20	25	75	°C

* With respect to V_{SS} (SYSTEM GND)

■ ELECTRICAL CHARACTERISTICS

• DC CHARACTERISTICS (V_{CC}=5V±5%, V_{SS} = 0V, Ta = -20~+75°C, unless otherwise noted.)

Item		Symbol	Test Condition	HD6809			HD68A09			HD68B09			Unit
				min	typ*	max	min	typ*	max	min	typ*	max	
Input "High" Voltage	Except $\overline{RES}$	V_{IH}		2.2	–	V_{CC}	2.2	–	V_{CC}	2.2	–	V_{CC}	V
	$\overline{RES}$			4.0	–	V_{CC}	4.0	–	V_{CC}	4.0	–	V_{CC}	
Input "Low" Voltage		V_{IL}		-0.3	–	0.8	-0.3	–	0.8	-0.3	–	0.8	V
Input Leakage Current	Except EXTAL, XTAL	Iin	Vin=0~5.25V, V_{CC}=max	-2.5	–	2.5	-2.5	–	2.5	-2.5	–	2.5	μA
Three State (Off State) Input Current	D_0~D_7	I_{TSI}	Vin=0.4~2.4V, V_{CC}=max	-10	–	10	-10	–	10	-10	–	10	μA
	A_0~A_{15}, R/$\overline{W}$			-100	–	100	-100	–	100	-100	–	100	
Output "High" Voltage	D_0~D_7	V_{OH}	I_{LOAD}=-205μA, V_{CC}=min	2.4	–	–	2.4	–	–	2.4	–	–	V
	A_0~A_{15}, R/$\overline{W}$, Q, E		I_{LOAD}=-145μA, V_{CC}=min	2.4	–	–	2.4	–	–	2.4	–	–	
	BA, BS		I_{LOAD}=-100μA, V_{CC}=min	2.4	–	–	2.4	–	–	2.4	–	–	
Output "Low" Voltage		V_{OL}	I_{LOAD}=2mA	–	–	0.5	–	–	0.5	–	–	0.5	V
Power Dissipation		P_D		–	–	1.0	–	–	1.0	–	–	1.0	W
Input Capacitance	D_0~D_7	Cin	Vin=0V, Ta=25°C, f=1MHz	–	10	15	–	10	15	–	10	15	pF
	Except D_0~D_7			–	7	10	–	7	10	–	7	10	
Output Capacitance	A_0~A_{15}, R/$\overline{W}$, BA, BS	Cout		–	–	12	–	–	12	–	–	12	pF

*Ta=25°C, V_{CC}=5V

- **AC CHARACTERISTICS** (V_{CC}=5V±5%, V_{SS} = 0V, Ta = -20~+75°C, unless otherwise noted.)

1. CLOCK TIMING

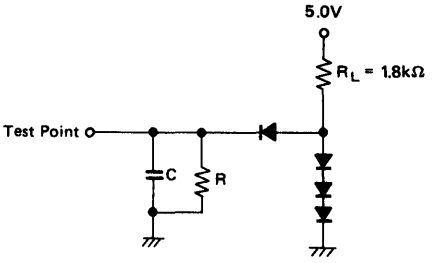
Item	Symbol	Test Condition	HD6809			HD68A09			HD68B09			Unit
			min	typ	max	min	typ	max	min	typ	max	
Frequency of Operation (Crystal or External Input)	f_{XTAL}	Fig. 2, Fig. 3	0.4	–	4	0.4	–	6	0.4	–	8	MHz
Cycle Time	t_{cyc}		1000	–	10000	667	–	10000	500	–	10000	ns
Total Up Time	t_{UT}		975	–	–	640	–	–	480	–	–	ns
Processor Clock "High"	t_{PWEH}		450	–	–	280	–	–	220	–	–	ns
Processor Clock "Low"	t_{PWEL}		450	–	–	295	–	–	210	–	–	ns
E Rise and Fall Time	t_{Er}, t_{Ef}		–	–	25	–	–	25	–	–	20	ns
E_{Low} to Q_{High} Time	t_{AVS}		–	–	250	–	–	165	–	–	125	ns
Q Clock "High"	t_{PWQH}		450	–	–	280	–	–	220	–	–	ns
Q Rise and Fall Time	t_{Qr}, t_{Qf}		–	–	25	–	–	25	–	–	20	ns
Q_{Low} to E Falling	t_{QE}		200	–	–	133	–	–	100	–	–	ns

2. BUS TIMING

Item	Symbol	Test Condition	HD6809			HD68A09			HD68B09			Unit
			min	typ	max	min	typ	max	min	typ	max	
Address Delay	t_{AD}	Fig. 2, Fig. 3	–	–	200	–	–	140	–	–	110	ns
Address Valid to Q_{High}	t_{AQ}		50	–	–	25	–	–	15	–	–	ns
Peripheral Read Access Time ($t_{UT}-t_{AD}-t_{DSR}=t_{ACC}$)	t_{ACC}		695	–	–	440	–	–	330	–	–	ns
Data Set Up Time (Read)	t_{DSR}		80	–	–	60	–	–	40	–	–	ns
Input Data Hold Time	t_{DHR}		10	–	–	10	–	–	10	–	–	ns
Address Hold Time (A_0~A_{15}, R/$\overline{W}$)	t_{AH}	Fig. 2, Fig. 3 Ta=0~+75°C	20	–	–	20	–	–	20	–	–	ns
		Fig. 2, Fig. 3 Ta=-20~0°C	10	–	–	10	–	–	10	–	–	ns
Data Delay Time (Write)	t_{DDW}	Fig. 3	–	–	225	–	–	180	–	–	145	ns
Output Hold Time	t_{DHW}	Fig. 3 Ta=0~+75°C	30	–	–	30	–	–	30	–	–	ns
		Fig. 3 Ta=-20~0°C	20	–	–	20	–	–	20	–	–	ns

3. PROCESSOR CONTROL TIMING

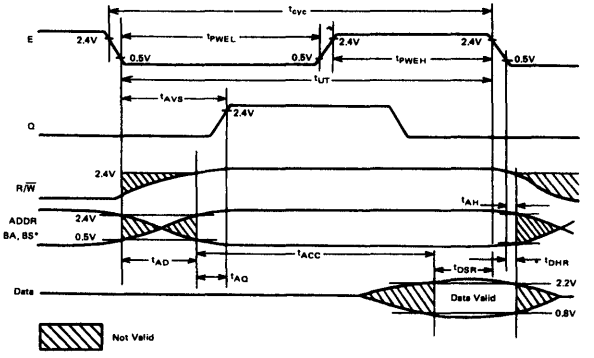
Item	Symbol	Test Condition	HD6809			HD68A09			HD68B09			Unit
			min	typ	max	min	typ	max	min	typ	max	
MRDY Set Up Time	t_{PCSM}	Fig. 6~Fig. 10 Fig. 14, Fig. 15	125	–	–	125	–	–	125	–	–	ns
Interrupts Set Up Time	t_{PCS}		200	–	–	140	–	–	110	–	–	ns
$\overline{HALT}$ Set Up Time	t_{PCSH}		200	–	–	140	–	–	110	–	–	ns
$\overline{RES}$ Set Up Time	t_{PCSR}		200	–	–	140	–	–	110	–	–	ns
DMA/$\overline{BREQ}$ Set Up Time	t_{PCSD}		125	–	–	125	–	–	125	–	–	ns
Processor Control Rise and Fall Time	t_{PCr}, t_{PCf}		–	–	100	–	–	100	–	–	100	ns
Crystal Oscillator Start Time	t_{RC}		–	–	50	–	–	30	–	–	30	ms



- C = 30pF (BA, BS)
  130pF ($D_0 \sim D_7$, E, Q)
  90pF ($A_0 \sim A_{15}$, R/$\overline{W}$)
- R = 11kΩ ($D_0 \sim D_7$)
  16kΩ ($A_0 \sim A_{15}$, E, Q, R/$\overline{W}$)
  24kΩ (BA, BS)

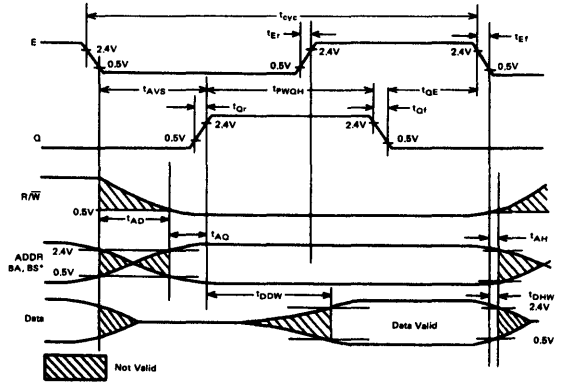
All diodes are 1S2074Ⓗ or equivalent.
C includes Stray Capacitance.

Figure 1 Bus Timing Test Load



*Hold time for BA, BS not specified.

Figure 2 Read Data from Memory or Peripherals



*Hold time for BA, BS not specified.

Figure 3 Write Data to Memory or Peripherals

■ PROGRAMMING MODEL

As shown in Figure 4, the HD6809 adds three registers to the set available in the HD6800. The added registers include a Direct Page Register, the User Stack pointer and a second Index Register.

• Accumulators (A, B, D)

The A and B registers are general purpose accumulators which are used for arithmetic calculations and manipulation of data.

Certain instructions concatenate the A and B registers to form a single 16-bit accumulator. This is referred to as the D register, and is formed with the A register as the most significant byte.

• Direct Page Register (DP)

The Direct Page Register of the HD6809 serves to enhance the Direct Addressing Mode. The content of this register appears at the higher address outputs ($A_8 \sim A_{15}$) during Direct Addressing Instruction execution. This allows the direct mode to be used at any place in memory, under program control. To ensure HD6800 compatibility, all bits of this register are cleared during Processor Reset.

- **Index Registers (X, Y)**

The Index Registers are used in indexed mode of addressing. The 16-bit address in this register takes part in the calculation of effective addresses. This address may be used to point to data directly or may be modified by an optional constant or register offset. During some indexed modes, the contents of the index register are incremented and decremented to point to the next item of tabular type data. All four pointer registers (X, Y, U, S) may be used as index registers.

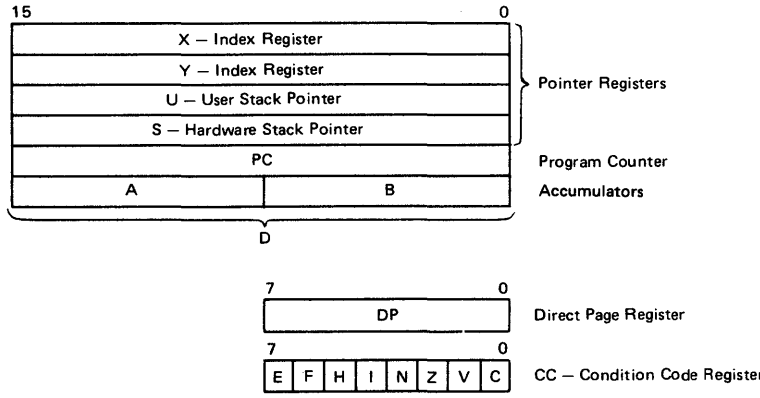


Figure 4 Programming Model of The Microprocessing Unit

- **Stack Pointer (U, S)**

The Hardware Stack Pointer (S) is used automatically by the processor during subroutine calls and interrupts. The stack pointers of the HD6809 point to the top of the stack, in contrast to the HD6800 stack pointer, which pointed to the next free location on the stack. The User Stack Pointer (U) is controlled exclusively by the programmer thus allowing arguments to be passed to and from subroutines with ease. Both Stack Pointers have the same indexed mode addressing capabilities as the X and Y registers, but also support Push and Pull instructions. This allows the HD6809 to be used efficiently as a stack processor, greatly enhancing its ability to support higher level languages and modular programming.

- **Program Counter**

The Program Counter is used by the processor to point to the address of the next instruction to be executed by the processor. Relative Addressing is provided allowing the Program Counter to be used like an index register in some situations.

- **Condition Code Register**

The Condition Code Register defines the State of the Processor at any given time. See Fig. 5.

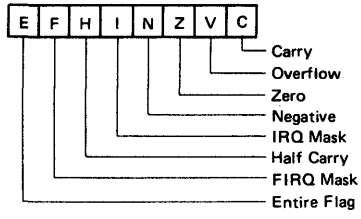


Figure 5 Condition Code Register Format

■ CONDITION CODE REGISTER DESCRIPTION

- **Bit 0 (C)**

Bit 0 is the carry flag, and is usually the carry from the binary ALU. C is also used to represent a 'borrow' from subtract like instructions (CMP, NEG, SUB, SBC) and is the complement of the carry from the binary ALU.

- **Bit 1 (V)**

Bit 1 is the overflow flag, and is set to a one by an operation which causes a signed two's complement arithmetic overflow. This overflow is detected in an operation in which the carry from the MSB in the ALU does not match the carry from the MSB-1.

- **Bit 2 (Z)**

Bit 2 is the zero flag, and is set to a one if the result of the previous operation was identically zero.

- **Bit 3 (N)**

Bit 3 is the negative flag, which contains exactly the value of the MSB of the result of the preceding operation. Thus, a negative two's-complement result will leave N set to a one.

- **Bit 4 (I)**

Bit 4 is the $\overline{IRQ}$ mask bit. The processor will not recognize interrupts from the $\overline{IRQ}$ line if this bit is set to a one. $\overline{NMI}$, $\overline{FIRQ}$, $\overline{IRQ}$, $\overline{RES}$, and SWI all are set I to a one; SWI2 and SWI3 do not affect I.

- **Bit 5 (H)**

Bit 5 is the half-carry bit, and is used to indicate a carry from bit 3 in the ALU as a result of an 8-bit addition only (ADC or ADD). This bit is used by the DAA instruction to perform a BCD decimal add adjust operation. The state of this flag is

undefined in all subtract-like instructions.

- **Bit 6 (F)**

Bit 6 is the $\overline{FIRQ}$ mask bit. The processor will not recognize interrupts from the $\overline{FIRQ}$ line if this bit is a one. $\overline{NMI}$, $\overline{FIRQ}$, SWI, and $\overline{RES}$ all set F to a one. $\overline{IRQ}$, SWI2 and SWI3 do not affect F.

- **Bit 7 (E)**

Bit 7 is the entire flag, and when set to a one indicates that the complete machine state (all the registers) was stacked, as opposed to the subset state (PC and CC). The E bit of the stacked CC is used on a return from interrupt (RTI) to determine the extent of the unstacking. Therefore, the current E left in the Condition Code Register represents past action.

■ SIGNAL DESCRIPTION

- **Power (V_{SS}, V_{CC})**

Two pins are used to supply power to the part: V_{SS} is ground or 0 volts, while V_{CC} is +5.0V ±5%.

- **Address Bus (A_0~A_{15})**

Sixteen pins are used to output address information from the MPU onto the Address Bus. When the processor does not require the bus for a data transfer, it will output address $FFFF_{16}$, R/$\overline{W}$ = "High", and BS = "Low"; this is a "dummy access" or $\overline{VMA}$ cycle. Addresses are valid on the rising edge of Q (see Figs. 2 and 3). All address bus drivers are made high impedance when output Bus Availalbe (BA) is "High". Each pin will drive one Schottky TTL load or four LS TTL loads, and typically 90 pF.

- **Data Bus (D_0~D_7)**

These eight pins provide communication with the system bi-directional data bus. Each pin will drive one Schottky TTL load or four LS TTL loads, and typically 130 pF.

- **Read/Write (R/$\overline{W}$)**

This signal indicates the direction of data transfer on the data bus. A "Low" indicates that the MPU is writing data onto the data bus. R/$\overline{W}$ is made high impedance when BA is "High". R/$\overline{W}$ is valid on the rising edge of Q. Refer to Figs. 2 and 3.

- **Reset ($\overline{RES}$)**

A "Low" level on this Schmitt-trigger input for greater than one bus cycle will reset the MPU, as shown in Fig. 6. The Reset vectors are fetched from locations $FFFE_{16}$ and $FFFF_{16}$ (Table 1) when Interrupt Acknowledge is true, ($\overline{BA}$ · BS=1). During initial power-on, the Reset line should be held "Low" until the clock oscillator is fully operational. See Fig. 7.

Because the HD6809 Reset pin has a Schmitt-trigger input with a threshold voltage higher than that of standard peripherals, a simple R/C network may be used to reset the entire system. This higher threshold voltage ensures that all peripherals are out of the reset state before the Processor.

Table 1 Memory Map for Interrupt Vectors

Memory Map For Vector Locations		Interrupt Vector Description
MS	LS	
FFFE	FFFF	$\overline{RES}$
FFFC	FFFD	$\overline{NMI}$
FFFA	FFFB	$\overline{SWI}$
FFF8	FFF9	$\overline{IRQ}$
FFF6	FFF7	$\overline{FIRQ}$
FFF4	FFF5	SWI2
FFF2	FFF3	SWI3
FFF0	FFF1	Reserved

- **$\overline{HALT}$**

A "Low" level on this input pin will cause the MPU to stop running at the end of the present instruction and remain halted indefinitely without loss of data. When halted, the BA output is driven "High" indicating the buses are high impedance. BS is also "High" which indicates the processor is in the Halt or Bus Grant state. While halted, the MPU will not respond to external real-time requests ($\overline{FIRQ}$, $\overline{IRQ}$) although $\overline{DMA/BREQ}$ will always be accepted, and $\overline{NMI}$ or $\overline{RES}$ will be latched for later response. During the Halt state Q and E continue to run normally. If the MPU is not running ($\overline{RES}$, $\overline{DMA/BREQ}$), a halted state (BA·BS=1) can be achieved by pulling $\overline{HALT}$ "Low" while $\overline{RES}$ is still "Low". If $\overline{DAM/BREQ}$ and $\overline{HALT}$ are both pulled "Low", the processor will reach the last cycle of the instruction (by reverse cycle stealing) where the machine will then become halted. See Figs. 8 and 16.

- **Bus Available, Bus Status (BA, BS)**

The BA output is an indication of an internal control signal which makes the MOS buses of the MPU high impedance. This signal does not imply that the bus will be available for more than one cycle. When BA goes "Low", an additional dead cycle will elapse before the MPU acquires the bus.

The BS output signal, when decoded with BA, represents the MPU state (valid with leading edge of Q).

Table 2 MPU State Definition

BA	BS	MPU State
0	0	Normal (Running)
0	1	Interrupt or RESET Acknowledge
1	0	SYNC Acknowledge
1	1	HALT or Bus Grant

Interrupt Acknowledge is indicated during both cycles of a hardware-vector-fetch ($\overline{RES}$, $\overline{NMI}$, $\overline{FIRQ}$, $\overline{IRQ}$, SWI, SWI2, SWI3). This signal, plus decoding of the lower four address lines, can provide the user with an indication of which interrupt level is being serviced and allow vectoring by device. See Table 1.

Sync Acknowledge is indicated while the MPU is waiting for external synchronization on an interrupt line.

Halt/Bus Grant is true when the HD6809 is in a Halt or Bus Grant condition.

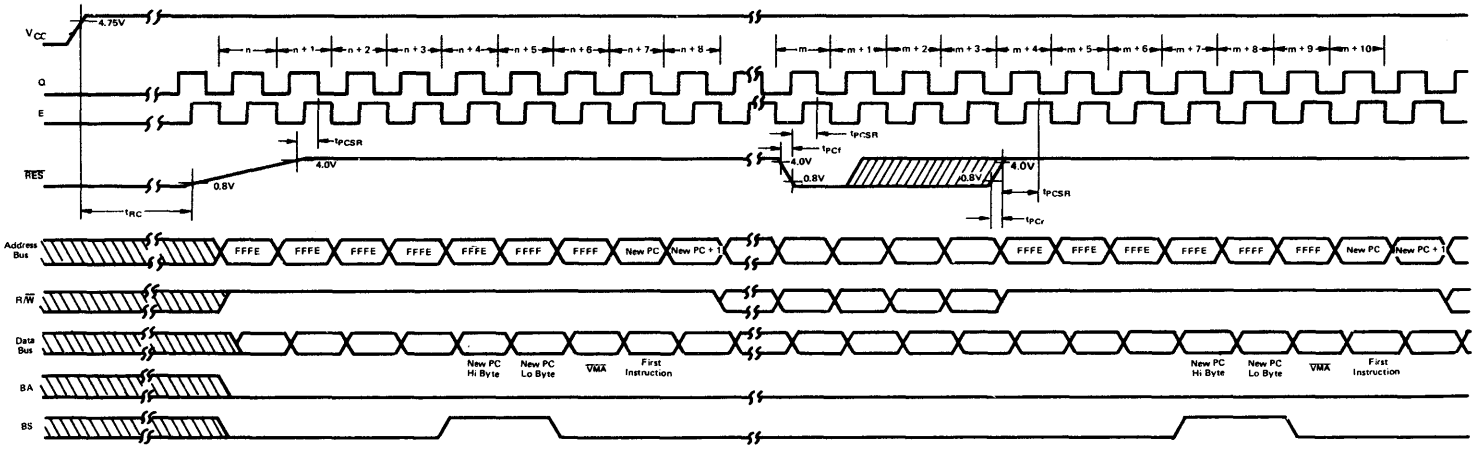
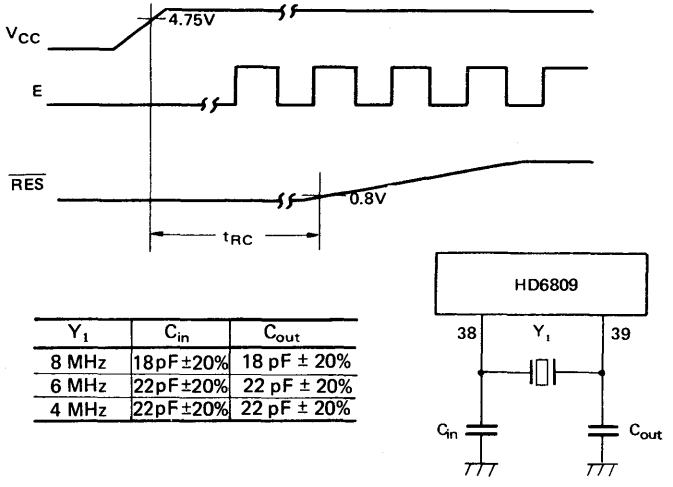


Figure 6 $\overline{RES}$ Timing



Y_1	C_{in}	C_{out}
8 MHz	18pF±20%	18 pF ± 20%
6 MHz	22pF±20%	22 pF ± 20%
4 MHz	22pF±20%	22 pF ± 20%

Figure 7 Crystal Connections and Oscillator Start Up

- **Non Maskable Interrupt ($\overline{NMI}$)***

A negative edge on this input requests that a non-maskable interrupt sequence be generated. A non-maskable interrupt cannot be inhibited by the program, and also has a higher priority than $\overline{FIRQ}$, $\overline{IRQ}$ or software interrupts. During recognition of an $\overline{NMI}$, the entire machine state is saved on the hardware stack. After reset, an $\overline{NMI}$ will not be recognized until the first program load of the Hardware Stack Pointer (S). The pulse width of $\overline{NMI}$ "Low" must be at least one E cycle. If the $\overline{NMI}$ input does not meet the minimum set up with respect to Q, the interrupt will not be recognized until the next cycle. See Fig. 9.

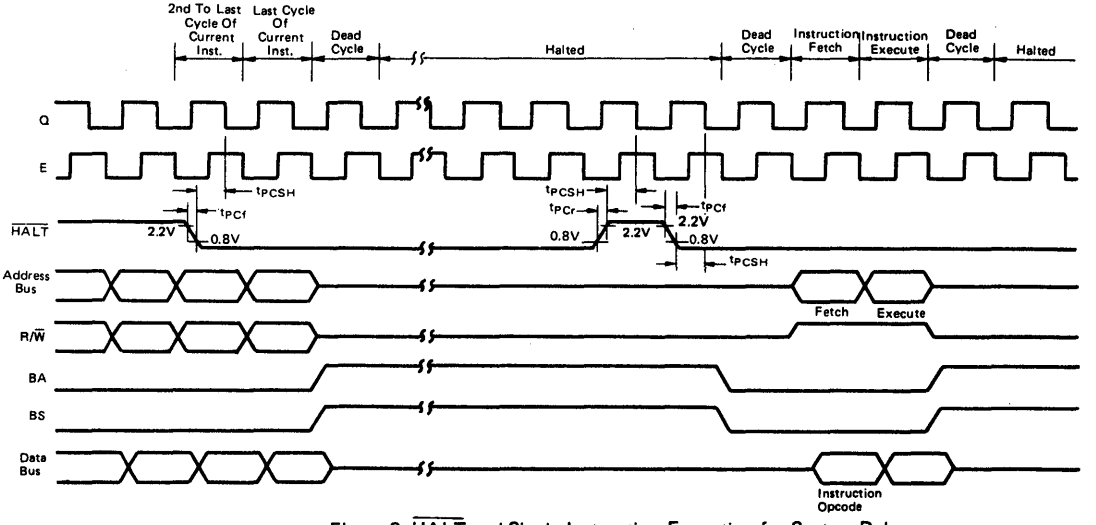


Figure 8 $\overline{HALT}$ and Single Instruction Execution for System Debug

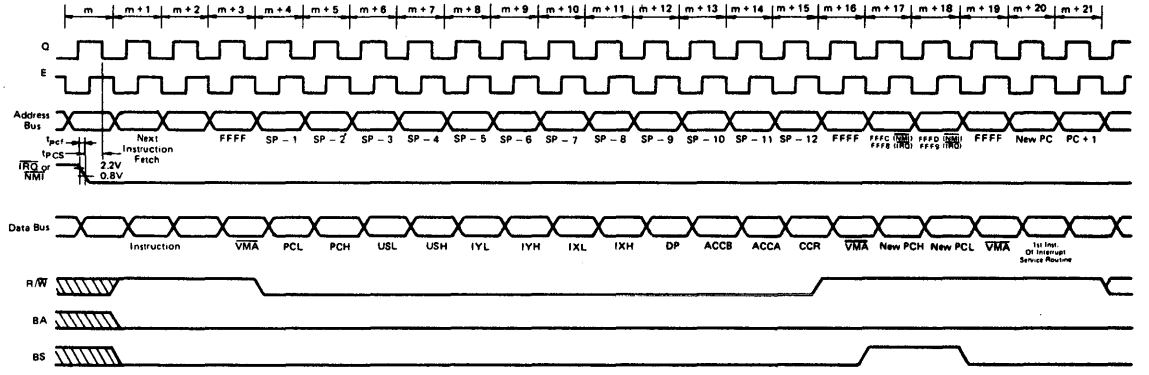


Figure 9 $\overline{IRQ}$ and $\overline{NMI}$ Interrupt Timing

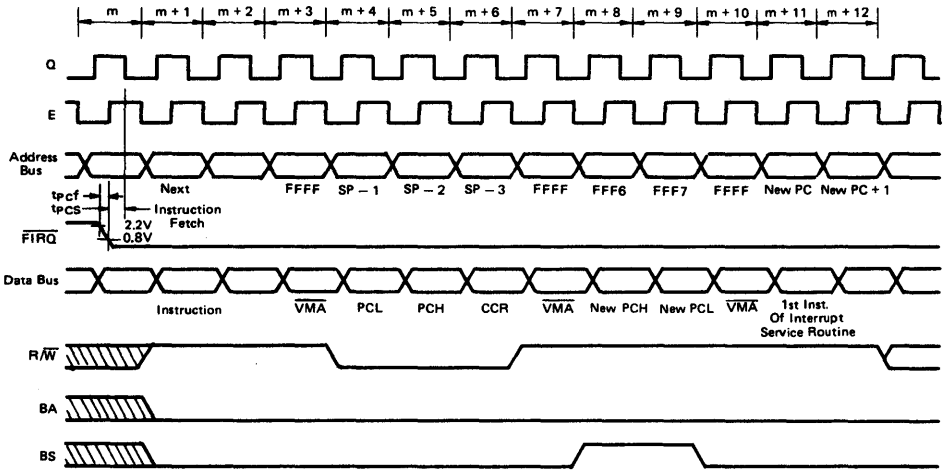


Figure 10 $\overline{\text{FIRQ}}$ Interrupt Timing

- **Fast-Interrupt Request ($\overline{\text{FIRQ}}$)***

A "Low" level on this input pin will initiate a fast interrupt sequence provided its mask bit (F) in the CC is clear. This sequence has priority over the standard Interrupt Request ($\overline{\text{IRQ}}$), and is fast in the sense that it stacks only the contents of the condition code register and the program counter. The interrupt service routine should clear the source of the interrupt before doing an RTI. See Fig. 10.

- **Interrupt Request ($\overline{\text{IRQ}}$)***

A "Low" level input on this pin will initiate an interrupt Request sequence provided the mask bit (I) in the CC is clear. Since $\overline{\text{IRQ}}$ stacks the entire machine state it provides a slower response to interrupts than $\overline{\text{FIRQ}}$. $\overline{\text{IRQ}}$ also has a lower priority than $\overline{\text{FIRQ}}$. Again, the interrupt service routine should clear the source of the interrupt before doing an RTI. See Fig. 9.

* $\overline{\text{NMI}}$, $\overline{\text{FIRQ}}$ and $\overline{\text{IRQ}}$ requests are latched by the falling edge of every Q, except during cycle stealing operations (e.g., DMA) where only $\overline{\text{NMI}}$ is latched. From this point, a delay of at least one bus cycle will occur before the interrupt is serviced by MPU.

- **XTAL, EXTAL**

These inputs are used to connect the on-chip oscillator to an external parallel-resonant crystal. Alternately, the pin EXTAL may be used as a TTL level input for external timing by grounding XTAL. The crystal or external frequency is four times the bus frequency. See Fig. 7. Proper RF layout techniques should be observed in the layout of printed circuit boards.

<NOTE FOR BOARD DESIGN OF THE OSCILLATION CIRCUIT>

In designing the board, the following notes should be taken when the crystal oscillator is used.

1) Crystal oscillator and load capacity Cin, Cout must be placed near the LSI as much as possible.

[Normal oscillation may be disturbed when external noise is induced to pin 38 and 39.]

2) Pin 38 and 39 signal line should be wired apart from other signal line as much as possible. Don't wire them in parallel.

[Normal oscillation may be disturbed when E or Q signal is feedbacked to pin 38 and 39.]

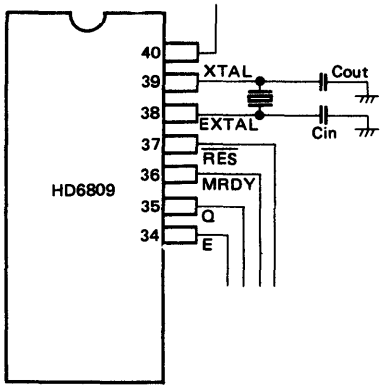


Figure 11 Board Design of the Oscillation Circuit.

<THE FOLLOWING DESIGN MUST BE AVOIDED>

A signal line or a power source line must not cross or go near the oscillation circuit line as shown in Fig. 12 to prevent the induction from these lines and perform the correct oscillation. The resistance among XTAL, EXTAL and other pins should be over 10MΩ.

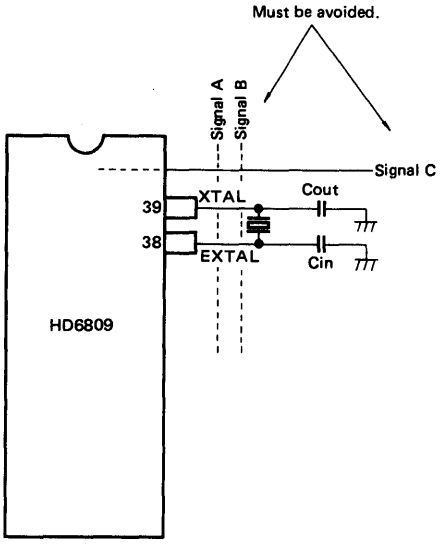


Figure 12 Example of Normal Oscillation may be Disturbed.

- **E, Q**

E is similar to the HD6800 bus timing signal ϕ_2; Q is a quadrature clock signal which leads E. Q has no parallel on the HD6800. Addresses from the MPU will be valid with the leading edge of Q. Data is latched on the falling edge of E. Timing for E and Q is shown in Fig. 13.

- **MRDY**

This input control signal allows stretching of E and Q to extend data-access time. E and Q operate normally while MRDY is "High". When MRDY is "Low", E and Q may be stretched in integral multiples of quarter (1/4) bus cycles, thus allowing interface to slow memories, as shown in Fig. 14. A maximum

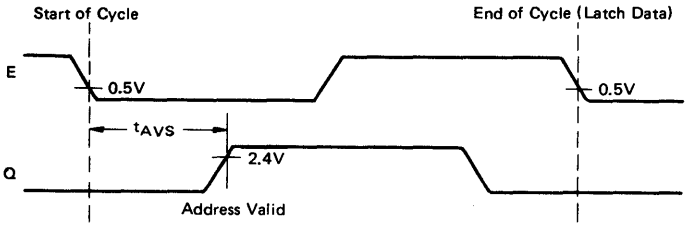


Figure 13 E/Q Relationship

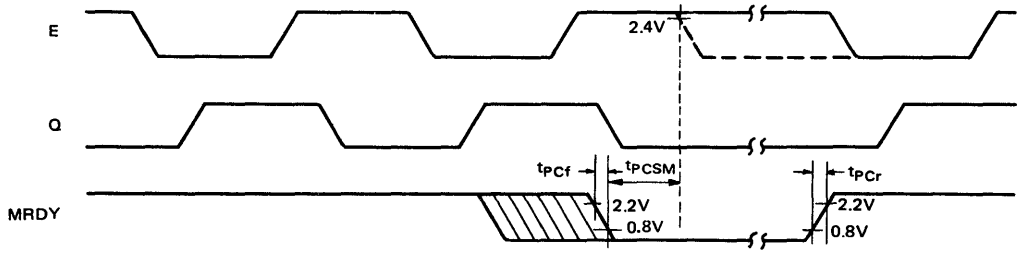


Figure 14 MRDY Timing

stretch is 10 microseconds. During nonvalid memory access ($\overline{VMA}$ cycles) MRDY has no effect on stretching E and Q; this inhibits slowing the processor during "don't care" bus accesses. MRDY may also be used to stretch clocks (for slow memory) when bus control has been transferred to an external device (through the use of $\overline{HALT}$ and $\overline{DMA/BREQ}$).

- **$\overline{DMA/BREQ}$**

The $\overline{DMA/BREQ}$ input provides a method of suspending execution and acquiring the MPU bus for another use, as shown in Fig. 15. Typical uses include DMA and dynamic memory refresh.

Transition of $\overline{DMA/BREQ}$ should occur during Q. A "Low" level on this pin will stop instruction execution at the end of the current cycle. The MPU will acknowledge $\overline{DMA/BREQ}$ by setting BA and BS to "High" level. The requesting device will now have up to 15 bus cycles before the MPU retrieves the bus for self-refresh. Self-refresh requires one bus cycle with a leading and trailing dead cycle. See Fig. 16.

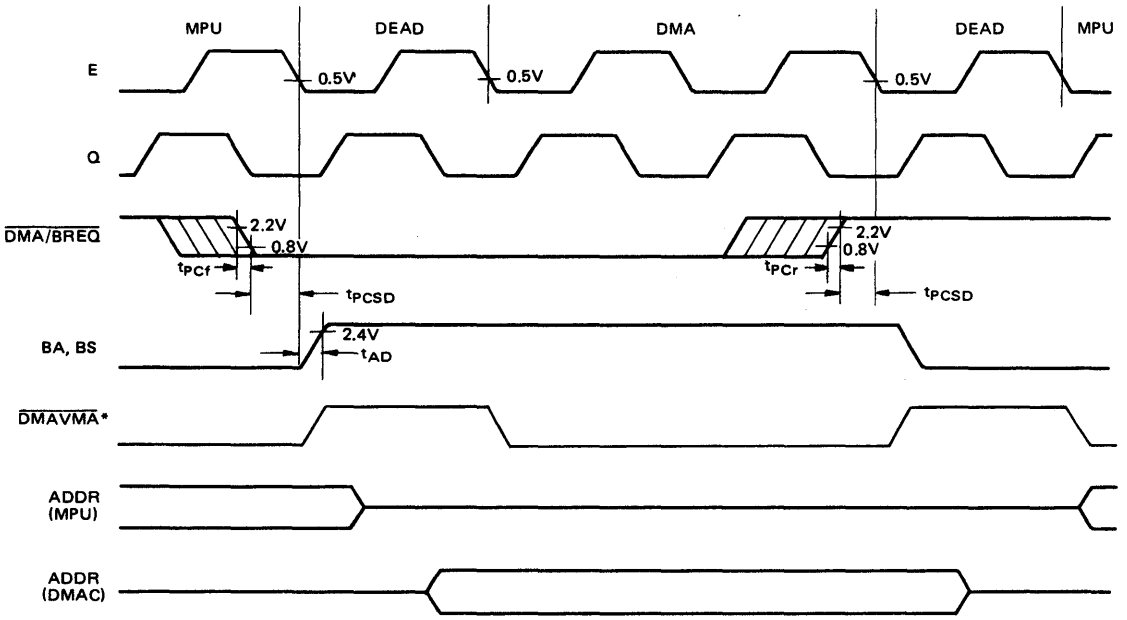
Typically, the DMA controller will request to use the bus by asserting $\overline{DMA/BREQ}$ pin "Low" on the leading edge of E. When the MPU replies by setting BA and BS to a one, that cycle will be a dead cycle used to transfer bus mastership to the DMA controller.

False memory accesses may be prevented during and dead cycles by developing a system $\overline{DMAVMA}$ signal which is "Low" in any cycle when BA has changed.

When BA goes "Low" (either as a result of $\overline{DMA/BREQ}$ = "High" or MPU self-refresh), the DMA device should be taken off the bus. Another dead cycle will elapse before the MPU accesses memory, to allow transfer of bus mastership without contention.

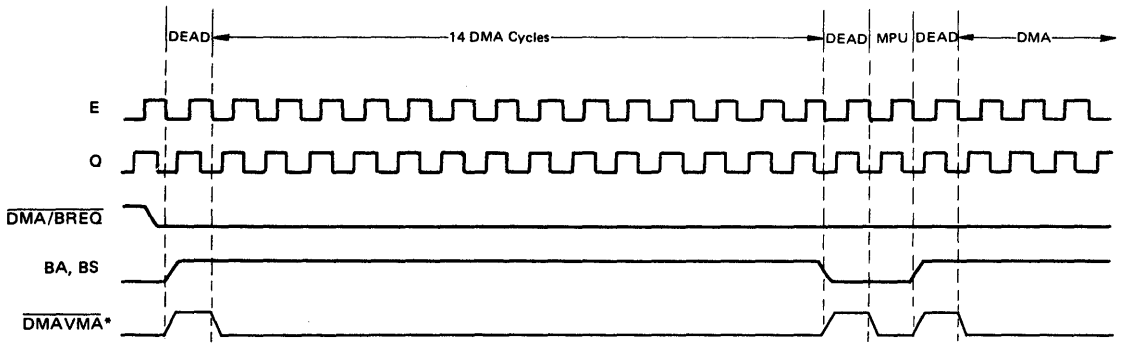
■ MPU OPERATION

During normal operation, the MPU fetches an instruction from memory and then executes the requested function. This



*$\overline{DMAVMA}$ is a signal which is developed externally, but is a system requirement for DMA.

Figure 15 Typical DMA Timing (<14 Cycles)



*$\overline{DMAVMA}$ is a signal which is developed externally, but is a system requirement for DMA.

Figure 16 Auto – Refresh DMA Timing
(Reverse Cycle Stealing)

sequence begins at $\overline{RES}$ and is repeated indefinitely unless altered by a special instruction or hardware occurrence. Software instructions that alter normal MPU operation are: SWI, SWI2, SWI3, CWAI, RTI and SYNC. An interrupt, $\overline{HALT}$ or $\overline{DMA/BREQ}$ can also alter the normal execution of instructions. Fig. 17 illustrates the flow chart for the HD6809.

■ ADDRESSING MODES

The basic instructions of any computer are greatly enhanced by the presence of powerful addressing modes. The HD6809 has the most complete set of addressing modes available on any microcomputer today. For example, the HD6809 has 59 basic instructions; however, it recognizes 1464 different variations of instructions and addressing modes. The addressing modes support modern programming techniques. The following addressing modes are available on the HD6809:

(1) Implied (Includes Accumulator)
(2) Immediate
(3) Extended
(4) Extended Indirect
(5) Direct
(6) Register
(7) Indexed
    Zero-Offset
    Constant Offset
    Accumulator Offset
    Auto Increment/Decrement
(8) Indexed Indirect
(9) Relative
(10) Program Counter Relative

• Implied (Includes Accumulator)

In this addressing mode, the opcode of the instruction contains all the address information necessary. Examples of Implied Addressing are: ABX, DAA, SWI, ASRA, and CLRB.

• Immediate Addressing

In Immediate Addressing, the effective address of the data is the location immediately following the opcode (i.e., the data to be used in the instruction immediately follows the opcode of the instruction). The HD6809 uses both 8 and 16-bit immediate values depending on the size of argument specified by the opcode. Examples of instructions with Immediate Addressing are:

```
LDA #$20
LDX #$F000
LDY #CAT
```

(NOTE) # signifies Immediate addressing, $ signifies hexadecimal value.

• Extended Addressing

In Extended Addressing, the contents of the two bytes immediately following the opcode fully specify the 16-bit effective address used by the instruction. Note that the address generated by an extended instruction defines an absolute address and is not position independent. Examples of Extended Addressing include:

```
LDA  CAT
STX  MOUSE
LDD  $2000
```

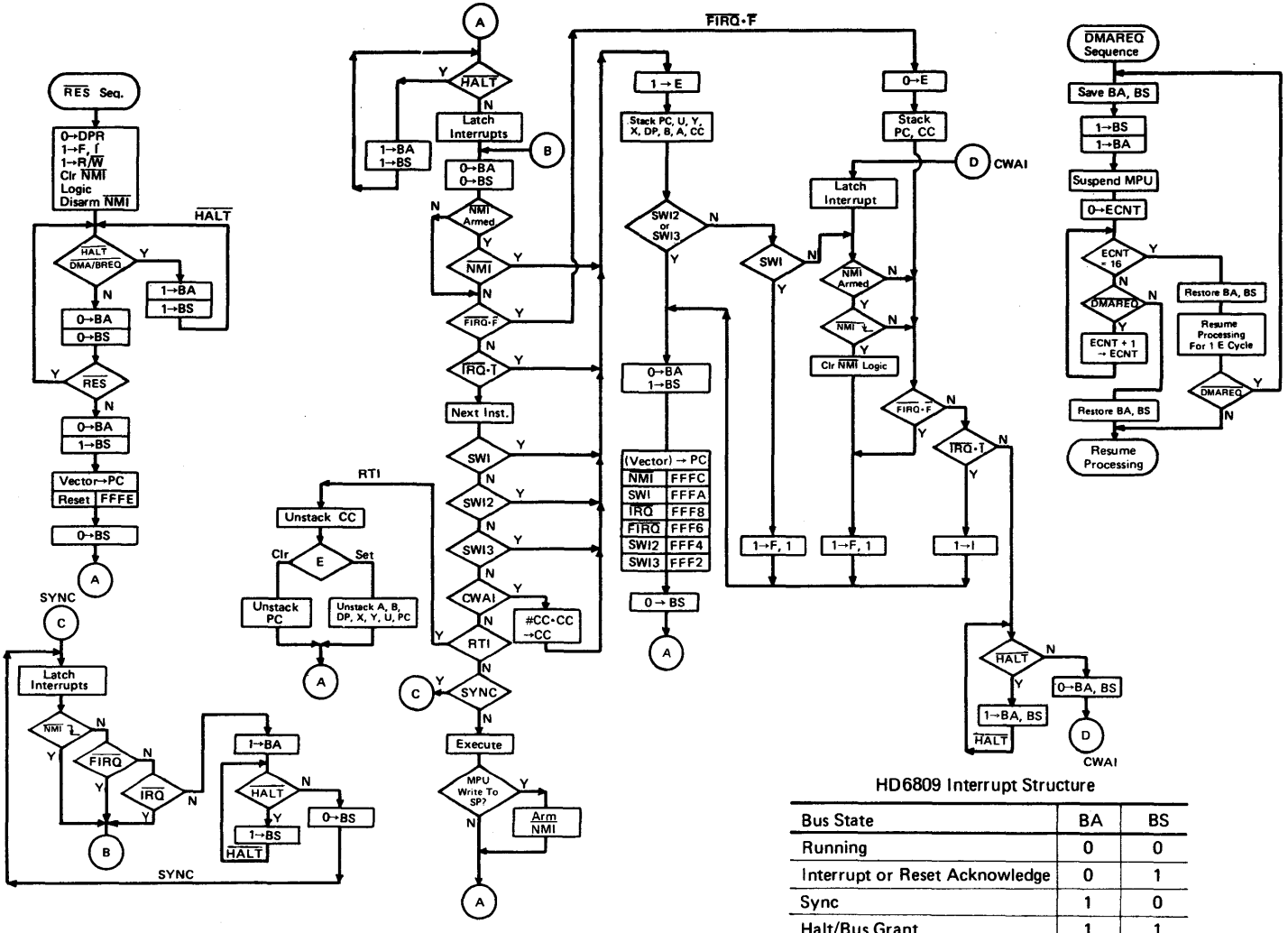
• Extended Indirect

As a special case of indexed addressing (discussed below), "1" level of indirection may be added to Extended Addressing. In Extended Indirect, the two bytes following the postbyte of an Indexed instruction contain the address of the data.

```
LDA  [CAT]
LDX  [$FFFE]
STU  [DOG]
```

• Direct Addressing

Direct addressing is similar to extended addressing except that only one byte of address follows the opcode. This byte specifies the lower 8-bit of the address to be used. The upper 8-bit of the address are supplied by the direct page register. Since only one byte of address is required in direct addressing, this mode requires less memory and executes faster than extended addressing. Of course, only 256 locations (one page) can be



HD6809 Interrupt Structure

Bus State	BA	BS
Running	0	0
Interrupt or Reset Acknowledge	0	1
Sync	1	0
Halt/Bus Grant	1	1

(NOTE) Asserting $\overline{RES}$ will result in entering the reset sequence from any point in the flow chart.

Figure 17 Flowchart for HD6809 Instruction

accessed without redefining the contents of the DP register. Since the DP register is set to $00 on Reset, direct addressing on the HD6809 is compatible with direct addressing on the HD6800. Indirection is not allowed in direct addressing. Some examples of direct addressing are:

```
LDA     $30
SETDP   $10 (Assembler directive)
LDB     $1030
LDD     <CAT
```

(NOTE) < is an assembler directive which forces direct addressing.

- **Register Addressing**

Some opcodes are followed by a byte that defines a register or set of registers to be used by the instruction. This is called a postbyte. Some examples of register addressing are:

```
TFR     X, Y        Transfers X into Y
EXG     A, B        Exchanges A with B
PSHS    A, B, X, Y  Push Y, X, B and A onto S
PULU    X, Y, D     Pull D, X, and Y from U
```

- **Indexed Addressing**

In all indexed addressing, one of the pointer registers (X, Y, U, S, and sometimes PC) is used in a calculation of the effective address of the operand to be used by the instruction. Five basic types of indexing are available and are discussed below. The postbyte of an indexed instruction specifies the basic type and variation of the addressing mode as well as the pointer register to be used. Fig. 18 lists the legal formats for the postbyte. Table 3 gives the assembler form and the number of cycles and bytes

Post-Byte Register Bit								Indexed Addressing Mode
7	6	5	4	3	2	1	0	
0	R	R	×	×	×	×	×	EA = ,R + 5 Bit Offset
1	R	R	0	0	0	0	0	,R +
1	R	R	0/1	0	0	0	1	,R + +
1	R	R	0	0	0	1	0	, -R
1	R	R	0/1	0	0	1	1	, - - R
1	R	R	0/1	0	1	0	0	EA = ,R + 0 Offset
1	R	R	0/1	0	1	0	1	EA = ,R + ACCB Offset
1	R	R	0/1	0	1	1	0	EA = ,R + ACCA Offset
1	R	R	0/1	1	0	0	0	EA = , R + 8 Bit Offset
1	R	R	0/1	1	0	0	1	EA = ,R + 16 Bit Offset
1	R	R	0/1	1	0	1	1	EA = ,R + D Offset
1	×	×	0/1	1	1	0	0	EA = ,PC + 8 Bit Offset
1	×	×	0/1	1	1	0	1	EA = ,PC + 16 Bit Offset
1	R	R	1	1	1	1	1	EA = [,Address]

Addressing Mode Field

Indirect Field
(Sigh bit when b7 = 0)
0 ..... Non Indirect
1 ..... Indirect

Register Field : RR
00 = X
01 = Y
10 = U
11 = S

× = Don't Care

Figure 18 Index Addressing Postbyte Register Bit Assignments

Table 3 Indexed Addressing Mode

Type	Forms	Non Indirect				Indirect			
		Assembler Form	Postbyte OP Code	+ ~	+ #	Assembler Form	Postbyte OP Code	+ ~	+ #
Constant Offset From R (2's Complement Offsets)	No Offset	,R	1RR00100	0	0	[,R]	1RR10100	3	0
	5 Bit Offset	n, R	0RRnnnnn	1	0	defaults to 8-bit			
	8 Bit Offset	n, R	1RR01000	1	1	[n, R]	1RR11000	4	1
	16 Bit Offset	n, R	1RR01001	4	2	[n, R]	1RR11001	7	2
Accumulator Offset From R (2's Complement Offsets)	A Register Offset	A, R	1RR00110	1	0	[A, R]	1RR10110	4	0
	B Register Offset	B, R	1RR00101	1	0	[B, R]	1RR10101	4	0
	D Register Offset	D, R	1RR01011	4	0	[D, R]	1RR11011	7	0
Auto Increment/Decrement R	Increment By 1	,R +	1RR00000	2	0	not allowed			
	Increment By 2	,R + +	1RR00001	3	0	[,R + +]	1RR10001	6	0
	Decrement By 1	, - R	1RR00010	2	0	not allowed			
	Decrement By 2	, - - R	1RR00011	3	0	[, - - R]	1RR10011	6	0
Constant Offset From PC (2's Complement Offsets)	8 Bit Offset	n, PCR	1xx01100	1	1	[n, PCR]	1xx11100	4	1
	16 Bit Offset	n, PCR	1xx01101	5	2	[n, PCR]	1xx11101	8	2
Extended Indirect	16 Bit Address	–	–	–	–	[n]	10011111	5	2

R = X, Y, U or S
× = Don't Care

RR:
00 = X
01 = Y
10 = U
11 = S

$\overset{+}{\sim}$ and $\overset{+}{\#}$ indicate the number of additional cycles and bytes for the particular variation.

added to the basic values for indexed addressing for each variation.

Zero-Offset Indexed

In this mode, the selected pointer register contains the effective address of the data to be used by the instruction. This is the fastest indexing mode.

Examples are:

```
LDD   0,X
LDA   S
```

Constant Offset Indexed

In this mode, a two's-complement offset and the contents of one of the pointer registers are added to form the effective address of the operand. The pointer register's initial content is unchanged by the addition.

Three sizes of offsets are available:

5-bit (−16 to +15)
8-bit (−128 to +127)
16-bit (−32768 to +32767)

The two's complement 5-bit offset is included in the postbyte and, therefore, is most efficient in use of bytes and cycles. The two's complement 8-bit offset is contained in a single byte following the postbyte. The two's complement 16-bit offset is in the two bytes following the postbyte. In most cases the programmer need not be concerned with the size of this offset since the assembler will select the optimal size automatically.

Examples of constant-offset indexing are:

```
LDA   23,X
LDX   −2,S
LDY   300,X
LDU   CAT,Y
```

Accumulator-Offset Indexed

This mode is similar to constant offset indexed except that the two's-complement value in one of the accumulators (A, B or D) and the contents of one of the pointer registers are added to form the effective address of the operand. The contents of both the accumulator and the pointer register are unchanged by the addition. The postbyte specifies which accumulator to use as an offset and no additional bytes are required. The advantage of an accumulator offset is that the value of the offset can be calculated by a program at run-time.

Some examples are:

```
LDA   B,Y
LDX   D,Y
LEAX  B,X
```

Auto Increment/Decrement Indexed

In the auto increment addressing mode, the pointer register contains the address of the operand. Then, after the pointer register is used it is incremented by one or two. This addressing mode is useful in stepping through tables, moving data, or for the creation of software stacks. In auto decrement, the pointer register is decremented prior to use as the address of the data. The use of auto decrement is similar to that of auto increment; but the tables, etc., are scanned from the "High" to "Low" addresses. The size of the increment/decrement can be either one or two to allow for tables of either 8 or 16-bit data to be accessed and is selectable by the programmer. The pre-decrement, post-increment nature of these modes allow them to be used to create additional software stacks that behave identically to the U and S stacks.

Some examples of the auto increment/decrement addressing modes are:

```
LDA   ,X+
STD   ,Y++
LDB   ,−Y
LDX   ,−−S
```

- **Indexed Indirect**

All of the indexing modes with the exception of auto increment/decrement by one, or a ±4-bit offset may have an additional level of indirection specified. In indirect addressing, the effective address is contained at the location specified by the contents of the Index register plus any offset. In the example below, the A accumulator is loaded indirectly using an effective address calculated from the Index register and an offset.

```
        Before Execution
        A = ×× (don't care)
        X = $F000
$0100   LDA [$10,X]      EA is now $F010

$F010   $F1              $F150 is now the
$F011   $50              new EA

$F150   $AA
        After Execution
        A = $AA Actual Data Loaded
        X = $F000
```

All modes of indexed indirect are included except those which are meaningless (e.g., auto increment/decrement by 1 indirect). Some examples of indexed indirect are:

```
LDA   [,X]
LDD   [10,S]
LDA   [B,Y]
LDD   [,X++]
```

- **Relative Addressing**

The byte(s) following the branch opcode is (are) treated as a signed offset which may be added to the program counter. If the branch condition is true then the calculated address (PC + signed offset) is loaded into the program counter. Program execution continues at the new location as indicated by the PC; short (1 byte offset) and long (2 bytes offset) relative addressing modes are available. All of memory can be reached in long relative addressing as an effective address is interpreted modulo 2^{16}. Some examples of relative addressing are:

```
        BEQ     CAT     (short)
        BGT     DOG     (short)
CAT     LBEQ    RAT     (long)
DOG     LBGT    RABBIT  (long)
        •
        •
        •
RAT     NOP
RABBIT  NOP
```

- **Program Counter Relative**

The PC can be used as the pointer register with 8 or 16-bit signed offsets. As in relative addressing, the offset is added to the current PC to create the effective address. The effective address is then used as the address of the operand or data. Program Counter Relative Addressing is used for writing

position independent programs. Tables related to a particular routine will maintain the same relationship after the routine is moved, if referenced relative to the Program Counter. Examples are:

```
LDA   CAT, PCR
LEAX  TABLE, PCR
```

Since program counter relative is a type of indexing, an additional level of indirection is available.

```
LDA   [CAT, PCR]
LDU   [DOG, PCR]
```

■ HD6809 INSTRUCTION SET

The instruction set of the HD6809 is similar to that of the HD6800 and is upward compatible at the source code level. The number of opcodes has been reduced from 72 to 59, but because of the expanded architecture and additional addressing modes, the number of available opcodes (with different addressing modes) has risen from 197 to 1464.

Some of the new instructions and addressing modes are described in detail below:

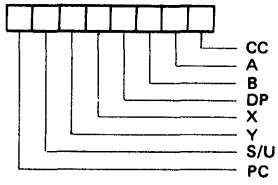
• PSHU/PSHS

The push instructions have the capability of pushing onto either the hardware stack (S) or user stack (U) any single register, or set of registers with a single instruction.

• PULU/PULS

The pull instructions have the same capability of the push instruction, in reverse order. The byte immediately following the push or pull opcode determines which register or registers are to be pushed or pulled. The actual PUSH/PULL sequence is fixed; each bit defines a unique register to push or pull, as shown in below.

PUSH/PULL POST BYTE



← Pull Order							Push Order →
PC	U	Y	X	DP	B	A	CC
FFFF...← increasing memory address 0000							
PC	S	Y	X	DP	B	A	CC

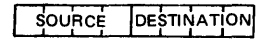
•TFR/EXG

Within the HD6809, any register may be transferred to or exchanged with another of like-size; i.e., 8-bit to 8-bit or 16-bit to 16-bit. Bits 4-7 of postbyte define the source register, while bits 0-3 represent the destination register. Three are denoted as follows:

0000 – D	0101 – PC
0001 – X	1000 – A
0010 – Y	1001 – B
0011 – U	1010 – CC
0100 – S	1011 – DP

(NOTE) All other combinations are undefined and INVALID.

TRANSFER/EXCHANGE POST BYTE



• LEAX/LEAY/LEAU/LEAS

The LEA (Load Effective Address) works by calculating the effective address used in an indexed instruction and stores that address value, rather than the data at that address, in a pointer register. This makes all the features of the internal addressing hardware available to the programmer. Some of the implications of this instruction are illustrated in Table 4.

The LEA instruction also allows the user to access data in a position independent manner. For example:

```
        LEAX    MSG1, PCR
        LBSR    PDATA (Print message routine)
          •
          •
MSG1 FCC        'MESSAGE'
```

This sample program prints: 'MESSAGE'. By writing MSG1, PCR, the assembler computes the distance between the present address and MSG1. This result is placed as a constant into the LEAX instruction which will be indexed from the PC value at the time of execution. No matter where the code is located, when it is executed, the computed offset from the PC will put the absolute address of MSG1 into the X pointer register. This code is totally position independent.

Table 4 LEA Examples

Instruction	Operation	Comment
LEAX 10, X	X + 10 → X	Adds 5-bit constant 10 to X
LEAX 500, X	X + 500 → X	Adds 16-bit constant 500 to X
LEAY A, Y	Y + A → Y	Adds 8-bit accumulator to Y
LEAY D, Y	Y + D → Y	Adds 16-bit D accumulator to Y
LEAU –10, U	U – 10 → U	Subtracts 10 from U
LEAS –10, S	S – 10 → S	Used to reserve area on stack
LEAS 10, S	S + 10 → S	Used to 'clean up' stack
LEAX 5, S	S + 5 → X	Transfers as well as adds

• MUL

Multiplies the unsigned binary numbers in the A and B accumulator and places the unsigned result into the 16-bit D accumulator. This unsigned multiply also allows multiple-precision multiplications.

Long And Short Relative Branches

The HD6809 has the capability of program counter relative branching throughout the entire memory map. In this mode, if the branch is to be taken, the 8 or 16-bit signed offset is added to the value of the program counter to be used as the effective address. This allows the program to branch anywhere in the 64k memory map. Position independent code can be easily generated through the use of relative branching. Both short (8-bit) and long (16-bit) branches are available.

- **SYNC**

After encountering a Sync instruction, the MPU enters a Sync state, stops processing instructions and waits for an interrupt. If the pending interrupt is non-maskable ($\overline{NMI}$) or maskable ($\overline{FIRQ}$, $\overline{IRQ}$) with its mask bit (F or I) clear, the processor will clear the Syne state and perform the normal interrupt stacking and service routine. Since $\overline{FIRQ}$ and $\overline{IRQ}$ are not edge-triggered, a "Low" level with a minimum duration of three bus cycles is required to assure that the interrupt will be taken. If the pending interrupt is maskable ($\overline{FIRQ}$, $\overline{IRQ}$) with its mask bit (F or I) set, the processor will clear the Sync state and continue processing by executing the next inline instruction. Fig. 19 depicts Sync timing.

Software Interrupts

A Software Interrupt is an instruction which will cause an interrupt, and its associated vector fetch. These Software Interrupts are useful in operating system calls, software debugging, trace operations, memory mapping, and software development systems. Three levels of SWI are available on this HD6809, and are prioritized in the following order: SWI, SWI2, SWI3.

16-Bit Operation

The HD6809 has the capability of processing 16-bit data. These instructions include loads, stores, compares, adds, subtracts, transfers, exchanges, pushes and pulls.

■ CYCLE-BY-CYCLE OPERATION

The address bus cycle-by-cycle performance chart illustrates the memory-access sequence corresponding to each possible instruction and addressing mode in the HD6809. Each instruction begins with an opcode fetch. While that opcode is being internally decoded, the next program byte is always fetched. (Most instructions will use the next byte, so this technique considerably speeds throughput.) Next, the operation of each opcode will follow the flow chart. $\overline{VMA}$ is an indication of $FFFF_{16}$ on the address bus, R/$\overline{W}$="High" and BS="Low". The following examples illustrate the use of the chart; see Fig. 20.

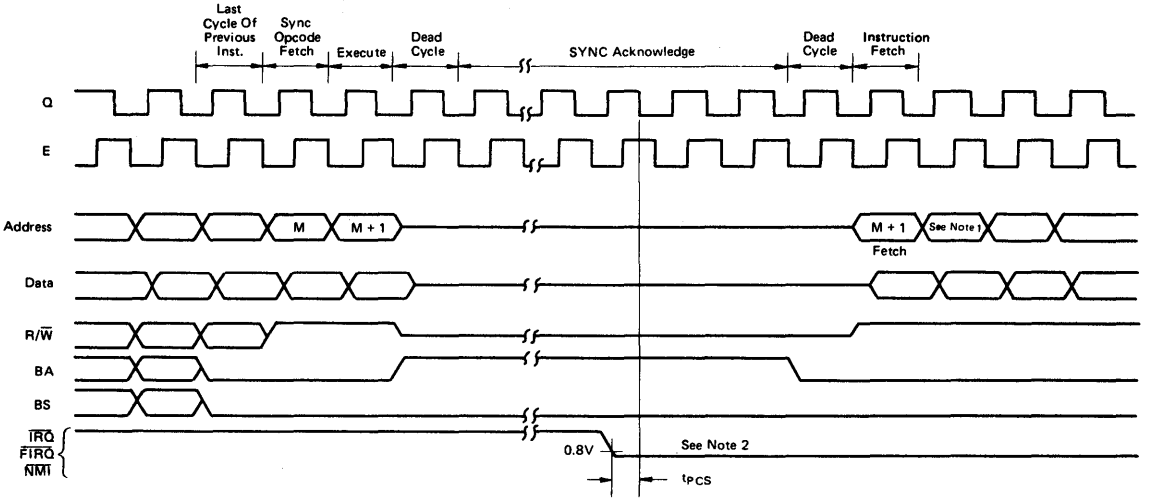
LBSR	(Branch taken)
Cycle # 1	opcode Fetch
2	opcode +
3	opcode +
4	$\overline{VMA}$
5	$\overline{VMA}$
6	ADDR
7	$\overline{VMA}$
8	STACK (write)
9	STACK (write)
DEC	(Extended)
Cycle # 1	opcode Fetch
2	opcode +
3	opcode +
4	$\overline{VMA}$
5	ADDR (read)
6	$\overline{VMA}$
7	ADDR (write)

■ HD6809 INSTRUCTION SET TABLES

The instructions of the HD6809 have been broken down into five different categories. They are as follows:

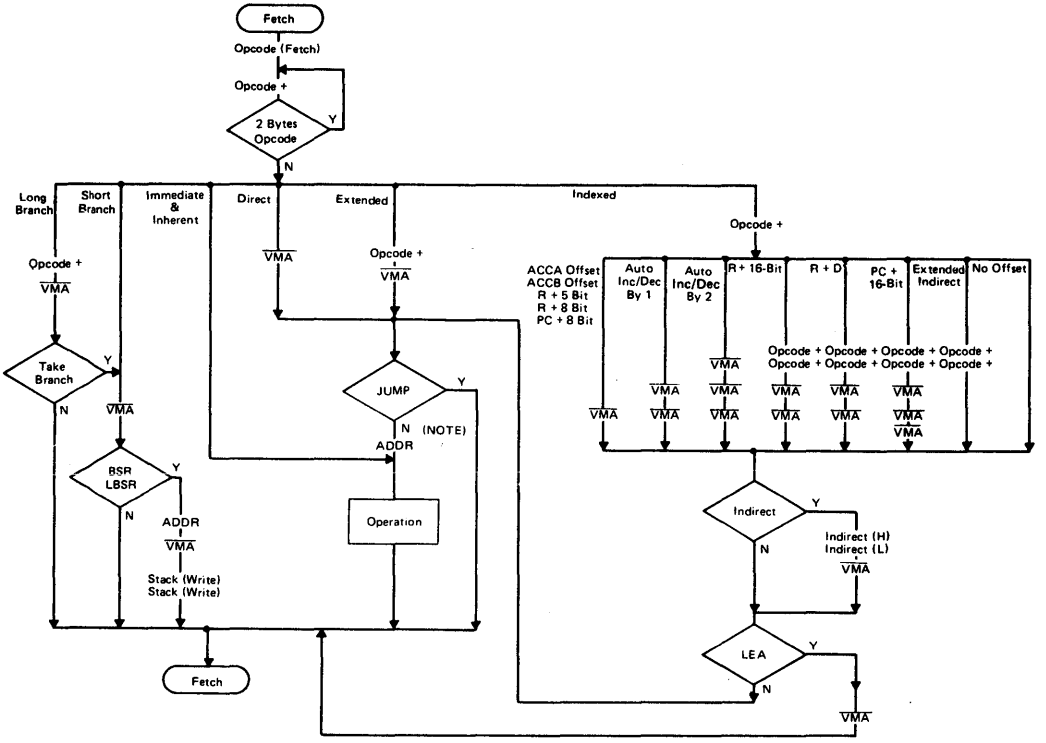
8-Bit operation (Table 5)
16-Bit operation (Table 6)
Index register/stack pointer instructions (Table 7)
Relative branches (long or short) (Table 8)
Miscellaneous instructions (Table 9)

HD6809 instruction set tables and Hexadecimal Values of instructions are shown in Table 10 and Table 11.



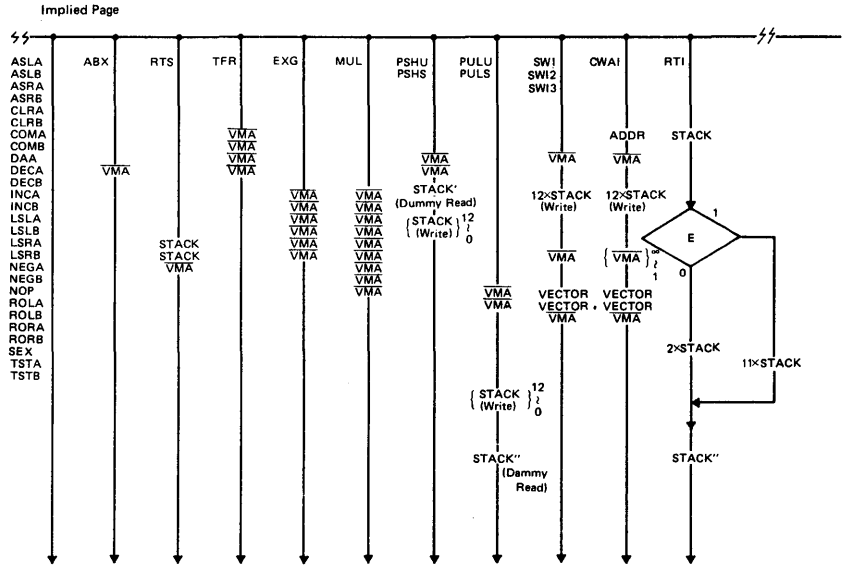
(NOTE) 1. If the associated mask bit is set when the interrupt is requested, this cycle will continue the instruction fetched from the previous cycle. However, if the interrupt is accepted ($\overline{NMI}$ or an unmasked $\overline{FIRQ}$ or $\overline{IRQ}$) the opcode address (placed on the bus from cycle M + 1) remains on the bus and interrupt processing continues with this cycle as (M + 2) on Figs. 9 and 10 (Interrupt Timing).
2. If mask bits are clear, $\overline{IRQ}$ and $\overline{FIRQ}$ must be held low for three cycles to guarantee interrupt to be taken, although only one cycle is necessary to bring the processor out of SYNC.

Figure 19 Sync Timing



(NOTE) Write operation during store instruction.

Figure 20 Address Bus Cycle-by-Cycle Performance



(NOTE) STACK' : Address stored in stack pointer before execution.
STACK": Address set to stack pointer as the result of the execution.

Figure 20 Address Bus Cycle-by-Cycle Performance (Continued)

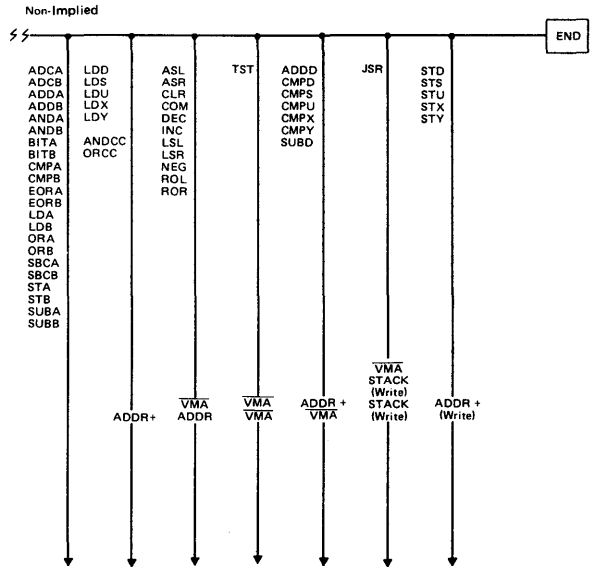


Figure 20 Address Bus Cycle-by-Cycle Performance (Continued)

Table 5 8-Bit Accumulator and Memory Instructions

Mnemonic(s)	Operation
ADCA, ADCB	Add memory to accumulator with carry
ADDA, ADDB	Add memory to accumulator
ANDA, ANDB	And memory with accumulator
ASL, ASLA, ASLB	Arithmetic shift of accumulator or memory left
ASR, ASRA, ASRB	Arithmetic shift of accumulator or memory right
BITA, BITB	Bit test memory with accumulator
CLR, CLRA, CLRB	Clear accumulator or memory location
CMPA, CMPB	Compare memory from accumulator
COM, COMA, COMB	Complement accumultor or memory location
DAA	Decimal adjust A accumulator
DEC, DECA, DECB	Decrement accumulator or memory location
EORA, EORB	Exclusive or memory with accumulator
EXG R1, R2	Exchange R1 with R2 (R1, R2 = A, B, CC, DP)
INC, INCA, INCB	Increment accumulator or memory location
LDA, LDB	Load accumulator from memory
LSL, LSLA, LSLB	Logical shift left accumulator or memory location
LSR, LSRA, LSRB	Logical shift right accumulator or memory location
MUL	Unsigned multiply (A × B → D)
NEG, NEGA, NEGB	Negate accumulator or memory
ORA, ORB	Or memory with accumulator
ROL, ROLA, ROLB	Rotate accumulator or memory left
ROR, RORA, RORB	Rotate accumulator or memory right
SBCA, SBCB	Subtract memory from accumulator with borrow
STA, STB	Store accumulator to memory
SUBA, SUBB	Subtract memory from accumulator
TST, TSTA, TSTB	Test accumulator or memory location
TFR R1, R2	Transfer R1 to R2 (R1, R2 = A, B, CC, DP)

(NOTE) A, B, CC or DP may be pushed to (pulled from) either stack with PSHS, PSHU (PULS, PULU) instructions.

Table 6 16-Bit Accumulator and Memory Instructions

Mnemonic(s)	Operation
ADDD	Add memory to D accumulator
CMPD	Compare memory from D accumulator
EXG D, R	Exchange D with X, Y, S, U or PC
LDD	Load D accumulator from memory
SEX	Sign Extend B accumulator into A accumulator
STD	Store D accumulator to memory
SUBD	Subtract memory from D accumulator
TFR D, R	Transfer D to X, Y, S, U or PC
TFR R, D	Transfer X, Y, S, U or PC to D

(NOTE) D may be pushed (pulled) to either stack with PSHS, PSHU (PULS, PULU) instructions.

Table 7 Index Register/Stack Pointer Instructions

Mnemonic(s)	Operation
CMPS, CMPU	Compare memory from stack pointer
CMPX, CMPY	Compare memory from index register
EXG R1, R2	Exchange D, X, Y, S, U or PC with D, X, Y, S, U or PC
LEAS, LEAU	Load effective address into stack pointer
LEAX, LEAY	Load effective address into index register
LDS, LDU	Load stack pointer from memory
LDX, LDY	Load index register from memory
PSHS	Push A, B, CC, DP, D, X, Y, U, or PC onto hardware stack
PSHU	Push A, B, CC, DP, D, X, Y, S, or PC onto user stack
PULS	Pull A, B, CC, DP, D, X, Y, U or PC from hardware stack
PULU	Pull A, B, CC, DP, D, X, Y, S or PC from user stack
STS, STU	Store stack pointer to memory
STX, STY	Store index register to memory
TFR R1, R2	Transfer D, X, Y, S, U or PC to D, X, Y, S, U or PC
ABX	Add B accumulator to X (unsigned)

Table 8 Branch Instructions

Mnemonic(s)	Operation
	SIMPLE BRANCHES
BEQ, LBEQ	Branch if equal
BNE, LBNE	Branch if not equal
BMI, LBMI	Branch if minus
BPL, LBPL	Branch if plus
BCS, LBCS	Branch if carry set
BCC, LBCC	Branch if carry clear
BVS, LBVS	Branch if overflow set
BVC, LBVC	Branch if overflow clear
	SIGNED BRANCHES
BGT, LBGT	Branch if greater (signed)
BGE, LBGE	Branch if greater than or equal (signed)
BEQ, LBEQ	Branch if equal
BLE, LBLE	Branch if less than or equal (signed)
BLT, LBLT	Branch if less than (signed)
	UNSIGNED BRANCHES
BHI, LBHI	Branch if higher (unsigned)
BHS, LBHS	Branch if higher or same (unsigned)
BEQ, LBEQ	Branch if equal
BLS, LBLS	Branch if lower or same (unsigned)
BLO, LBLO	Branch if lower (unsigned)
	OTHER BRANCHES
BSR, LBSR	Branch to subroutine
BRA, LBRA	Branch always
BRN, LBRN	Branch never

Table 9 Miscellaneous Instructions

Mnemonic(s)	Operation
ANDCC	AND condition code register
CWAI	AND condition code register, then wait for interrupt
NOP	No operation
ORCC	OR condition code register
JMP	Jump
JSR	Jump to subroutine
RTI	Return from interrupt
RTS	Return from subroutine
SWI, SWI2, SWI3	Software interrupt (absolute indirect)
SYNC	Synchronize with interrupt line

Table 10 HD6809 Instruction Set Table

INSTRUCTION/ FORMS		HD6809 ADDRESSING MODES																DESCRIPTION	5	3	2	1	0
		IMPLIED			DIRECT			EXTENDED			IMMEDIATE			INDEXED①			RELATIVE				H	N	Z	V	C
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~⑤	#						
ABX		3A	3	1																B + X → X (UNSIGNED)	•	•	•	•	•
ADC	ADCA				99	4	2	B9	5	3	89	2	2	A9	4+	2+				A + M + C → A	↕	↕	↕	↕	↕
	ADCB				D9	4	2	F9	5	3	C9	2	2	E9	4+	2+				B + M + C → B	↕	↕	↕	↕	↕
ADD	ADDA				9B	4	2	BB	5	3	8B	2	2	AB	4+	2+				A + M → A	↕	↕	↕	↕	↕
	ADDB				DB	4	2	FB	5	3	CB	2	2	EB	4+	2+				B + M → B	↕	↕	↕	↕	↕
	ADDD				D3	6	2	F3	7	3	C3	4	3	E3	6+	2+				D + M:M + 1 → D	•	↕	↕	↕	↕
AND	ANDA				94	4	2	B4	5	3	84	2	2	A4	4+	2+				A ∧ M → A	•	↕	↕	0	•
	ANDB				D4	4	2	F4	5	3	C4	2	2	E4	4+	2+				B ∧ M → B	•	↕	↕	0	•
	ANDCC										1C	3	2							CC ∧ IMM → CC	(		⑦		)
ASL	ASLA	48	2	1																A	⑧	↕	↕	↕	↕
	ASLB	58	2	1																B	⑧	↕	↕	↕	↕
	ASL				08	6	2	78	7	3				68	6+	2+				M	⑧	↕	↕	↕	↕
ASR	ASRA	47	2	1																A	⑧	↕	↕	•	↕
	ASRB	57	2	1																B	⑧	↕	↕	•	↕
	ASR				07	6	2	77	7	3				67	6+	2+				M	⑧	↕	↕	•	↕
BCC	BCC																24	3	2	Branch C = 0	•	•	•	•	•
	LBCC																10 24	5(6)	4	Long Branch C = 0	•	•	•	•	•
BCS	BCS																25	3	2	Branch C = 1	•	•	•	•	•
	LBCS																10 25	5(6)	4	Long Branch C = 1	•	•	•	•	•
BEQ	BEQ																27	3	2	Branch Z = 1	•	•	•	•	•
	LBEQ																10 27	5(6)	4	Long Branch Z = 1	•	•	•	•	•
BGE	BGE																2C	3	2	Branch N ⊕ V=0	•	•	•	•	•
	LBGE																10 2C	5(6)	4	Long Branch N ⊕ V=0	•	•	•	•	•
BGT	BGT																2E	3	2	Branch Z∨(N⊕V)=0	•	•	•	•	•
	LBGT																10 2E	5(6)	4	Long Branch Z∨(N ⊕ V)=0	•	•	•	•	•
BHI	BHI																22	3	2	Branch C∨Z=0	•	•	•	•	•
	LBHI																10 22	5(6)	4	Long Branch C∨Z=0	•	•	•	•	•
BHS	BHS																24	3	2	Branch C=0	•	•	•	•	•
	LBHS																10 24	5(6)	4	Long Branch C=0	•	•	•	•	•
BIT	BITA				95	4	2	B5	5	3	85	2	2	A5	4+	2+				Bit Test A (M ∧ A)	•	↕	↕	0	•
	BITB				D5	4	2	F5	5	3	C5	2	2	E5	4+	2+				Bit Test B (M ∧ B)	•	↕	↕	0	•
BLE	BLE																2F	3	2	Branch Z∨(N⊕V)=1	•	•	•	•	•
	LBLE																10 2F	5(6)	4	Long Branch Z∨(N ⊕ V)=1	•	•	•	•	•
BLO	BLO																25	3	2	Branch C=1	•	•	•	•	•
	LBLO																10 25	5(6)	4	Long Branch C=1	•	•	•	•	•
BLS	BLS																23	3	2	Branch C∨Z=1	•	•	•	•	•
	LBLS																10 23	5(6)	4	Long Branch C∨Z=1	•	•	•	•	•
BLT	BLT																2D	3	2	Branch N ⊕ V=1	•	•	•	•	•
	LBLT																10 2D	5(6)	4	Long Branch N ⊕ V=1	•	•	•	•	•
BMI	BMI																2B	3	2	Branch N=1	•	•	•	•	•
	LBMI																10 2B	5(6)	4	Long Branch N=1	•	•	•	•	•
BNE	BNE																26	3	2	Branch Z = 0	•	•	•	•	•
	LBNE																10 26	5(6)	4	Long Branch Z = 0	•	•	•	•	•
BPL	BPL																2A	3	2	Branch N = 0	•	•	•	•	•
	LBPL																10 2A	5(6)	4	Long Branch N = 0	•	•	•	•	•
BRA	BRA																20	3	2	Branch Always	•	•	•	•	•
	LBRA																16	5	3	Long Branch/ Always	•	•	•	•	•
BRN	BRN																21	3	2	Branch Never	•	•	•	•	•
	LBRN																10 21	5	4	Long Branch Never	•	•	•	•	•

(to be continued)

INSTRUCTION/ FORMS		HD6809 ADDRESSING MODES																DESCRIPTION	5	3	2	1	0
		IMPLIED			DIRECT			EXTENDED			IMMEDIATE			INDEXED①			RELATIVE				H	N	Z	V	C
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~⑤	#						
BSR	BSR																8D	7	2	Branch to Subroutine	•	•	•	•	•
	LBSR																17	9	3	Long Branch to Subroutine	•	•	•	•	•
BVC	BVC																28	3	2	Branch V = 0	•	•	•	•	•
	LBVC																10 28	5(6)	4	Long Branch V = 0	•	•	•	•	•
BVS	BVS																29	3	2	Branch V = 1	•	•	•	•	•
	LBVS																10 29	5(6)	4	Long Branch V = 1	•	•	•	•	•
CLR	CLRA	4F	2	1																0 → A	•	0	1	0	0
	CLRB	5F	2	1																0 → B	•	0	1	0	0
	CLR				0F	6	2	7F	7	3				6F	6+	2+				0 → M	•	0	1	0	0
CMP	CMPA				91	4	2	B1	5	3	81	2	2	A1	4+	2+				Compare M from A ⑧	‡	‡	‡	‡	‡
	CMPB				D1	4	2	F1	5	3	C1	2	2	E1	4+	2+				Compare M from B ⑧	‡	‡	‡	‡	‡
	CMPD				10 93	7	3	10 B3	8	4	10 83	5	4	10 A3	7+	3+				Compare M: M + 1 from D	•	‡	‡	‡	‡
	CMPS				11 9C	7	3	11 BC	8	4	11 8C	5	4	11 AC	7+	3+				Compare M: M + 1 from S	•	‡	‡	‡	‡
	CMPU				11 93	7	3	11 B3	8	4	11 83	5	4	11 A3	7+	3+				Compare M: M + 1 from U	•	‡	‡	‡	‡
	CMPX				9C	6	2	BC	7	3	8C	4	3	AC	6+	2+				Compare M: M + 1 from X	•	‡	‡	‡	‡
	CMPY				10 9C	7	3	10 BC	8	4	10 8C	5	4	10 AC	7+	3+				Compare M: M + 1 from Y	•	‡	‡	‡	‡
COM	COMA	43	2	1																$\overline{A}$ → A	•	‡	‡	0	1
	COMB	53	2	1																$\overline{B}$ → B	•	‡	‡	0	1
	COM				03	6	2	73	7	3				63	6+	2+				$\overline{M}$ → M	•	‡	‡	0	1
CWAI		3C	20	2																CC ∧ IMM → CC (except 1→E) Wait for Interrupt	(———		⑦	———	)
DAA		19	2	1																Decimal Adjust A	•	‡	‡	⑧	‡
DEC	DECA	4A	2	1																A − 1 → A	•	‡	‡	‡	•
	DECB	5A	2	1																B − 1 → B	•	‡	‡	‡	•
	DEC				0A	6	2	7A	7	3				6A	6+	2+				M − 1 → M	•	‡	‡	‡	•
EOR	EORA				98	4	2	B8	5	3	88	2	2	A8	4+	2+				A ⊕ M → A	•	‡	‡	0	•
	EORB				D8	4	2	F8	5	3	C8	2	2	E8	4+	2+				B ⊕ M → B	•	‡	‡	0	•
EXG	R1, R2	1E	7	2																R1 ↔ R2②	(———		⑩	———	)
INC	INCA	4C	2	1																A + 1 → A	•	‡	‡	‡	•
	INCB	5C	2	1																B + 1 → B	•	‡	‡	‡	•
	INC				0C	6	2	7C	7	3				6C	6+	2+				M + 1 → M	•	‡	‡	‡	•
JMP					0E	3	2	7E	4	3				6E	3+	2+				EA③ → PC	•	•	•	•	•
JSR					9D	7	2	BD	8	3				AD	7+	2+				Jump to Subroutine	•	•	•	•	•
LD	LDA				96	4	2	B6	5	3	86	2	2	A6	4+	2+				M → A	•	‡	‡	0	•
	LDB				D6	4	2	F6	5	3	C6	2	2	E6	4+	2+				M → B	•	‡	‡	0	•
	LDD				DC	5	2	FC	6	3	CC	3	3	EC	5+	2+				M: M + 1 → D	•	‡	‡	0	•
	LDS				10 DE	6	3	10 FE	7	4	10 CE	4	4	10 EE	6+	3+				M: M + 1 → S	•	‡	‡	0	•
	LDU				DE	5	2	FE	6	3	CE	3	3	EE	5+	2+				M: M + 1 → U	•	‡	‡	0	•
	LDX				9E	5	2	BE	6	3	8E	3	3	AE	5+	2+				M: M + 1 → X	•	‡	‡	0	•
	LDY				10 9E	6	3	10 BE	7	4	10 8E	4	4	10 AE	6+	3+				M: M + 1 → Y	•	‡	‡	0	•
LEA	LEAS													32	4+	2+				EA③ → S	•	•	•	•	•
	LEAU													33	4+	2+				EA③ → U	•	•	•	•	•
	LEAX													30	4+	2+				EA③ → X	•	•	‡	•	•
	LEAY													31	4+	2+				EA③ → Y	•	•	‡	•	•
LSL	LSLA	48	2	1																A	•	‡	‡	‡	‡
	LSLB	58	2	1																B	•	‡	‡	‡	‡
	LSL				08	6	2	78	7	3				68	6+	2+				M 	•	‡	‡	‡	‡
LSR	LSRA	44	2	1																A	•	0	‡	•	‡
	LSRB	54	2	1																B	•	0	‡	•	‡
	LSR				04	6	2	74	7	3				64	6+	2+				M 	•	0	‡	•	‡
MUL		3D	11	1																A × B → D (Unsigned)	•	•	‡	•	⑨
NEG	NEGA	40	2	1																$\overline{A}$ + 1 → A	⑧	‡	‡	‡	‡
	NEGB	50	2	1																$\overline{B}$ + 1 → B	⑧	‡	‡	‡	‡
	NEG				00	6	2	70	7	3				60	6+	2+				$\overline{M}$ + 1 → M	⑧	‡	‡	‡	‡
NOP		12	2	1																No Operation	•	•	•	•	•

(to be continued)

INSTRUCTION/ FORMS		HD6809 ADDRESSING MODES																	DESCRIPTION	5	3	2	1	0
		IMPLIED			DIRECT			EXTENDED			IMMEDIATE			INDEXED①			RELATIVE				H	N	Z	V	C
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~⑤	#						
OR	ORA				9A	4	2	BA	5	3	8A	2	2	AA	4+	2+				A ∨ M → A	•	↕	↕	0	•
	ORB				DA	4	2	FA	5	3	CA	2	2	EA	4+	2+				B ∨ M → B	•	↕	↕	0	•
	ORCC										1A	3	2							CC ∨ IMM → CC	(		⑦		)
PSH	PSHS	34	5+④	2																Push Registers on S Stack	•	•	•	•	•
	PSHU	36	5+④	2																Push Registers on U Stack	•	•	•	•	•
PUL	PULS	35	5+④	2																Pull Registers from S Stack	(		⑩		)
	PULU	37	5+④	2																Pull Registers from U Stack	(		⑩		)
ROL	ROLA	49	2	1																A	•	↕	↕	↕	↕
	ROLB	59	2	1																B	•	↕	↕	↕	↕
	ROL				09	6	2	79	7	3				69	6+	2+				M	•	↕	↕	↕	↕
ROR	RORA	46	2	1																A	•	↕	↕	•	↕
	RORB	56	2	1																B	•	↕	↕	•	↕
	ROR				06	6	2	76	7	3				66	6+	2+				M	•	↕	↕	•	↕
RTI		3B	6/15	1																Return From Interrupt	(		⑦		)
RTS		39	5	1																Return From Subroutine	•	•	•	•	•
SBC	SBCA				92	4	2	B2	5	3	82	2	2	A2	4+	2+				A − M − C → A	⑧	↕	↕	↕	↕
	SBCB				D2	4	2	F2	5	3	C2	2	2	E2	4+	2+				B − M − C → B	⑧	↕	↕	↕	↕
SEX		1D	2	1																Sign Extend B into A	•	↕	↕	•	•
ST	STA				97	4	2	B7	5	3				A7	4+	2+				A → M	•	↕	↕	0	•
	STB				D7	4	2	F7	5	3				E7	4+	2+				B → M	•	↕	↕	0	•
	STD				DD	5	2	FD	6	3				ED	5+	2+				D → M: M+ 1	•	↕	↕	0	•
	STS				10 DF	6	3	10 FF	7	4				10 EF	6+	3+				S → M: M + 1	•	↕	↕	0	•
	STU				DF	5	2	FF	6	3				EF	5+	2+				U → M: M + 1	•	↕	↕	0	•
	STX				9F	5	2	BF	6	3				AF	5+	2+				X → M: M + 1	•	↕	↕	0	•
	STY				10 9F	6	3	10 BF	7	4				10 AF	6+	3+				Y → M: M + 1	•	↕	↕	0	•
SUB	SUBA				90	4	2	B0	5	3	80	2	2	A0	4+	2+				A − M → A	⑧	↕	↕	↕	↕
	SUBB				D0	4	2	F0	5	3	C0	2	2	E0	4+	2+				B − M → B	⑧	↕	↕	↕	↕
	SUBD				93	6	2	B3	7	3	83	4	3	A3	6+	2+				D − M: M + 1 → D	•	↕	↕	↕	↕
SWI	SWI⑥	3F	19	1																Software Interrupt1	•	•	•	•	•
	SWI2⑥	10 3F	20	2																Software Interrupt2	•	•	•	•	•
	SWI3⑥	11 3F	20	2																Software Interrupt3	•	•	•	•	•
SYNC		13	≥2	1																Synchronize to Interrupt	•	•	•	•	•
TFR	R1, R2	1F	6	2																R1 → R2②	(		⑩		)
TST	TSTA	4D	2	1																Test A	•	↕	↕	0	•
	TSTB	5D	2	1																Test B	•	↕	↕	0	•
	TST				0D	6	2	7D	7	3				6D	6+	2+				Test M	•	↕	↕	0	•

(NOTES)

① This column gives a base cycle and byte count. To obtain total count, and the values obtained from the INDEXED ADDRESSING MODES table.
② R1 and R2 may be any pair of 8 bit or any pair of 16 bit registers.
The 8 bit registers are: A, B, CC, DP
The 16 bit registers are: X, Y, U, S, D, PC
③ EA is the effective address.
④ The PSH and PUL instructions require 5 cycle plus 1 cycle for each byte pushed or pulled.
⑤ 5(6) means: 5 cycles if branch not taken, 6 cycles if taken.
⑥ SWI sets 1 and F bits. SWI2 and SWI3 do not affect I and F.
⑦ Conditions Codes set as a direct result of the instruction.
⑧ Value of half-carry flag is undefined.
⑨ Special Case — Carry set if b7 is SET.
⑩ Condition Codes set as a direct result of the instruction if CC is specified, and not affected otherwise.

LEGEND:

OP	Operation Code (Hexadecimal)	Z	Zero (byte)
~	Number of MPU Cycles	V	Overflow, 2's complement
#	Number of Program Bytes	C	Carry from bit 7
+	Arithmetic Plus	↕	Test and set if true, cleared otherwise
−	Arithmetic Minus	•	Not Affected
×	Multiply	CC	Condition Code Register
$\overline{M}$	Complement of M	:	Concatenation
→	Transfer Into	∨	Logical or
H	Half-carry (from bit 3)	∧	Logical and
N	Negative (sign bit)	⊕	Logical Exclusive or

Table 11 Hexadecimal Values of Machine Codes

OP	Mnem	Mode	~	#	OP	Mnem	Mode	~	#	OP	Mnem	Mode	~	#
00	NEG	Direct	6	2	30	LEAX	Indexed	4+	2+	60	NEG	Indexed	6+	2+
01	*				31	LEAY		4+	2+	61	*			
02	*				32	LEAS		4+	2+	62	*			
03	COM		6	2	33	LEAU	Indexed	4+	2+	63	COM		6+	2+
04	LSR		6	2	34	PSHS	Implied	5+	2	64	LSR		6+	2+
05	*				35	PULS		5+	2	65	*			
06	ROR		6	2	36	PSHU		5+	2	66	ROR		6+	2+
07	ASR		6	2	37	PULU		5+	2	67	ASR		6+	2+
08	ASL, LSL		6	2	38	*				68	ASL, LSL		6+	2+
09	ROL		6	2	39	RTS		5	1	69	ROL		6+	2+
0A	DEC		6	2	3A	ABX		3	1	6A	DEC		6+	2+
0B	*				3B	RTI		6, 15	1	6B	*			
0C	INC		6	2	3C	CWAI		20	2	6C	INC		6+	2+
0D	TST		6	2	3D	MUL		11	1	6D	TST		6+	2+
0E	JMP		3	2	3E	*				6E	JMP		3+	2+
0F	CLR	Direct	6	2	3F	SWI	Implied	19	1	6F	CLR	Indexed	6+	2+
10	See	—	—	—	40	NEGA	Implied	2	1	70	NEG	Extended	7	3
11	Next Page	—	—	—	41	*				71	*			
12	NOP	Implied	2	1	42	*				72	*			
13	SYNC	Implied	2	1	43	COMA		2	1	73	COM		7	3
14	*				44	LSRA		2	1	74	LSR		7	3
15	*				45	*				75	*			
16	LBRA	Relative	5	3	46	RORA		2	1	76	ROR		7	3
17	LBSR	Relative	9	3	47	ASRA		2	1	77	ASR		7	3
18	*				48	ASLA, LSLA		2	1	78	ASL, LSL		7	3
19	DAA	Implied	2	1	49	ROLA		2	1	79	ROL		7	3
1A	ORCC	Immed	3	2	4A	DECA		2	1	7A	DEC		7	3
1B	*				4B	*				7B	*			
1C	ANDCC	Immed	3	2	4C	INCA		2	1	7C	INC		7	3
1D	SEX	Implied	2	1	4D	TSTA		2	1	7D	TST		7	3
1E	EXG		8	2	4E	*				7E	JMP		4	3
1F	TFR	Implied	6	2	4F	CLRA	Implied	2	1	7F	CLR	Extended	7	3
20	BRA	Relative	3	2	50	NEGB	Implied	2	1	80	SUBA	Immed	2	2
21	BRN		3	2	51	*				81	CMPA		2	2
22	BHI		3	2	52	*				82	SBCA		2	2
23	BLS		3	2	53	COMB		2	1	83	SUBD		4	3
24	BHS, BCC		3	2	54	LSRB		2	1	84	ANDA		2	2
25	BLO, BCS		3	2	55	*				85	BITA		2	2
26	BNE		3	2	56	RORB		2	1	86	LDA		2	2
27	BEQ		3	2	57	ASRA		2	1	87	*			
28	BVC		3	2	58	ASLB, LSLB		2	1	88	EORA		2	2
29	BVS		3	2	59	ROLB		2	1	89	ADCA		2	2
2A	BPL		3	2	5A	DECB		2	1	8A	ORA		2	2
2B	BMI		3	2	5B	*				8B	ADDA		2	2
2C	BGE		3	2	5C	INCB		2	1	8C	CMPX	Immed	4	3
2D	BLT		3	2	5D	TSTB		2	1	8D	BSR	Relative	7	2
2E	BGT		3	2	5E	*				8E	LDX	Immed	3	3
2F	BLE	Relative	3	2	5F	CLRB	Implied	2	1	8F	*			

LEGEND:
~ Number of MPU cycles (less possible push pull or indexed-mode cycles)
# Number of program bytes
* Denotes unused opcode

(to be continued)

OP	Mnem	Mode	~	#
90	SUBA	Direct	4	2
91	CMPA	↑	4	2
92	SBCA		4	2
93	SUBD		6	2
94	ANDA		4	2
95	BITA		4	2
96	LDA		4	2
97	STA		4	2
98	EORA		4	2
99	ADCA		4	2
9A	ORA		4	2
9B	ADDA		4	2
9C	CMPX		6	2
9D	JSR		7	2
9E	LDX	↓	5	2
9F	STX	Direct	5	2
A0	SUBA	Indexed	4+	2+
A1	CMPA	↑	4+	2+
A2	SBCA		4+	2+
A3	SUBD		6+	2+
A4	ANDA		4+	2+
A5	BITA		4+	2+
A6	LDA		4+	2+
A7	STA		4+	2+
A8	EORA		4+	2+
A9	ADCA		4+	2+
AA	ORA		4+	2+
AB	ADDA		4+	2+
AC	CMPX		6+	2+
AD	JSR		7+	2+
AE	LDX	↓	5+	2+
AF	STX	Indexed	5+	2+
B0	SUBA	Extended	5	3
B1	CMPA	↑	5	3
B2	SBCA		5	3
B3	SUBD		7	3
B4	ANDA		5	3
B5	BITA		5	3
B6	LDA		5	3
B7	STA		5	3
B8	EORA		5	3
B9	ADCA		5	3
BA	ORA		5	3
BB	ADDA		5	3
BC	CMPX		7	3
BD	JSR		8	3
BE	LDX	↓	6	3
BF	STX	Extended	6	3
C0	SUBB	Immed	2	2
C1	CMPB	↑	2	2
C2	SBCB		2	2
C3	ADDD		4	3
C4	ANDB		2	2
C5	BITB	↓ Immed	2	2
C6	LDB	Immed	2	2
C7	*	↑		
C8	EORB		2	2
C9	ADCB		2	2
CA	ORB		2	2
CB	ADDB		2	2
CC	LDD		3	3
CD	*	↓		
CE	LDU	Immed	3	3
CF	*			
D0	SUBB	Direct	4	2
D1	CMPB	↑	4	2
D2	SBCB		4	2
D3	ADDD		6	2
D4	ANDB		4	2
D5	BITB		4	2
D6	LDB		4	2
D7	STB		4	2
D8	EORB		4	2
D9	ADCB		4	2
DA	ORB		4	2
DB	ADDB		4	2
DC	LDD		5	2
DD	STD		5	2
DE	LDU	↓	5	2
DF	STU	Direct	5	2
E0	SUBB	Indexed	4+	2+
E1	CMPB	↑	4+	2+
E2	SBCB		4+	2+
E3	ADDD		6+	2+
E4	ANDB		4+	2+
E5	BITB		4+	2+
E6	LDB		4+	2+
E7	STB		4+	2+
E8	EORB		4+	2+
E9	ADCB		4+	2+
EA	ORB		4+	2+
EB	ADDB		4+	2+
EC	LDD		5+	2+
ED	STD		5+	2+
EE	LDU		5+	2+
EF	STU	↓ Indexed	5+	2+
F0	SUBB	Extended	5	3
F1	CMPB	↑	5	3
F2	SBCB		5	3
F3	ADDD		7	3
F4	ANDB		5	3
F5	BITB		5	3
F6	LDB		5	3
F7	STB		5	3
F8	EORB		5	3
F9	ADCB		5	3
FA	ORB		5	3
FB	ADDB	↓ Extended	5	3
FC	LDD	Extended	6	3
FD	STD	↕	6	3
FE	LDU		6	3
FF	STU	↓ Extended	6	3

2 Bytes Opcode

1021	LBRN	Relative	5	4
1022	LBHI	↑	5(6)	4
1023	LBLS		5(6)	4
1024	LBHS, LBCC		5(6)	4
1025	LBCS, LBLO		5(6)	4
1026	LBNE		5(6)	4
1027	LBEQ		5(6)	4
1028	LBVC		5(6)	4
1029	LBVS		5(6)	4
102A	LBPL		5(6)	4
102B	LBMI		5(6)	4
102C	LBGE		5(6)	4
102D	LBLT		5(6)	4
102E	LBGT		5(6)	4
102F	LBLE	↓ Relative	5(6)	4
103F	SWI2	Implied	20	2
1083	CMPD	Immed	5	4
108C	CMPY	↕	5	4
108E	LDY	Immed	4	4
1093	CMPD	Direct	7	3
109C	CMPY	↑	7	3
109E	LDY	↓	6	3
109F	STY	Direct	6	3
10A3	CMPD	Indexed	7+	3+
10AC	CMPY	↑	7+	3+
10AE	LDY	↓	6+	3+
10AF	STY	Indexed	6+	3+
10B3	CMPD	Extended	8	4
10BC	CMPY	↑	8	4
10BE	LDY	↓	7	4
10BF	STY	Extended	7	4
10CE	LDS	Immed	4	4
10DE	LDS	Direct	6	3
10DF	STS	Direct	6	3
10EE	LDS	Indexed	6+	3+
10EF	STS	Indexed	6+	3+
10FE	LDS	Extended	7	4
10FF	STS	Extended	7	4
113F	SWI3	Implied	20	2
1183	CMPU	Immed	5	4
118C	CMPS	Immed	5	4
1193	CMPU	Direct	7	3
119C	CMPS	Direct	7	3
11A3	CMPU	Indexed	7+	3+
11AC	CMPS	Indexed	7+	3+
11B3	CMPU	Extended	8	4
11BC	CMPS	Extended	8	4

(NOTE): All unused opcodes are both undefined and illegal

■ NOTE FOR USE

[1] Exceptional Operation of HD6809

(a) Exceptional Operations of $\overline{\text{DMA/BREQ}}$, BA signals (#1)

HD6809 acknowledges the input signal level of $\overline{\text{DMA/BREQ}}$ at the end of each cycle, then determines whether the next sequence is MPU or DMA. When "Low" level is detected, HD6809 executes DMA sequence by setting BA, BS to "High" level. However, in the conditions shown below the assertion of BA, BS delays one clock cycle.

< Conditions for the exception >

(1) $\overline{\text{DMA/BREQ}}$: "Low" for 6~13 cycles

(2) $\overline{\text{DMA/BREQ}}$: "High" for 3 cycles

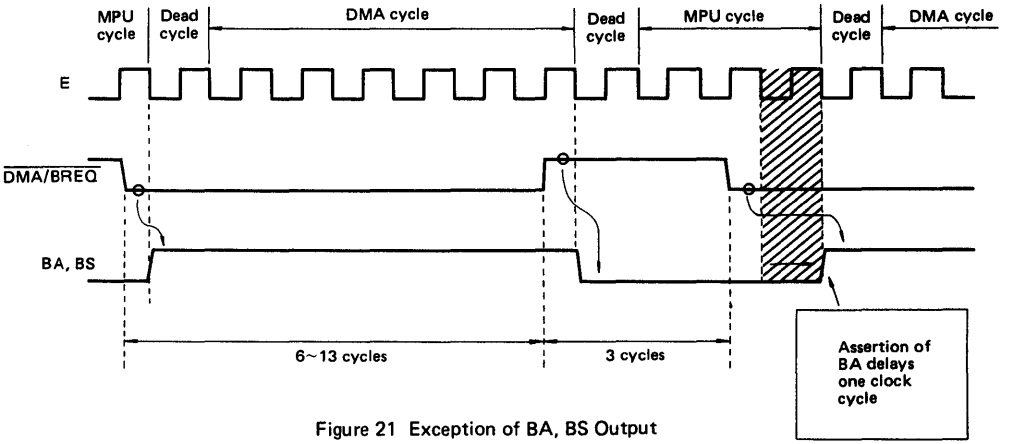


Figure 21 Exception of BA, BS Output

(b) Exceptional Operations of $\overline{\text{DMA/BREQ}}$, BA signals (#2)

HD6809 includes a self refresh counter for the reverce cycle steal. And it is only cleared if $\overline{\text{DMA/BREQ}}$ is inactive ("High") for 3 or more MPU cycles. So 1 or 2 inactive cycle(s) doesn't affect the self refresh counter.

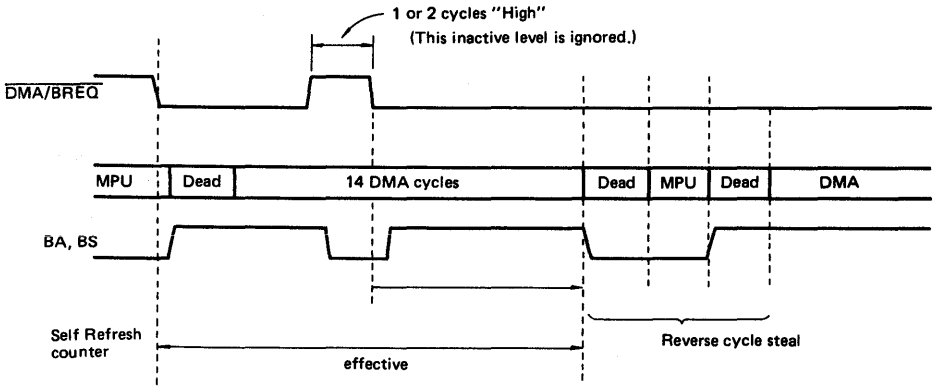


Figure 22 Exception of $\overline{\text{DMA/BREQ}}$

(c) **How to avoid these exceptional operations**

It is necessary to provide 4 or more cycles for inactive $\overline{DMA/BREQ}$ level as shown in Fig. 23.

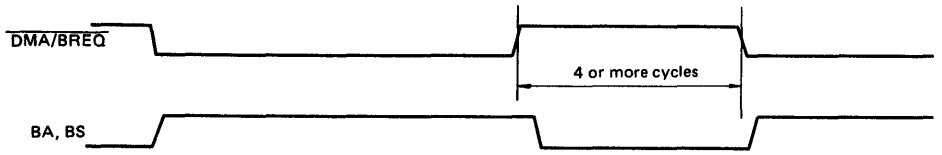


Figure 23 How to Avoid Exceptional Operations

[2] **Restriction for DMA Transfer**

There is a restriction for the DMA transfer in the HD6809 (MPU), HD6844 (DMAC) system. Please take care of following.

(a) **An Example of the System Configuration**

This restriction is applied to the following system.

(1) $\overline{DMA/BREQ}$ is used for DMA request.

(2) "Halt Burst Mode" is used for DMA transfer

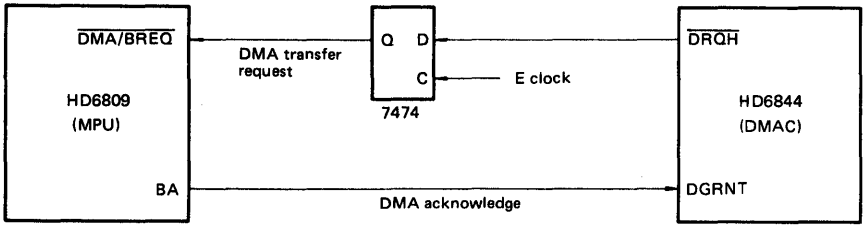


Figure 24 An Example of HD6809, HD6844 System

(The restriction is also applied to the system which doesn't use 7474 Flip-Flop. Fig. 24, Fig. 25 shows an example which uses 7474 for synchronizing DMA request with E.)

(b) **Restriction**

"The number of transfer Byte per one DMA Burst transfer must be less than or equal to 14."

Halt burst DMA transfer should be less than or equal to 14 cycles. In another word, the number stored into DMA Byte count register should be 0~14.

★ Please than care of the section [1](b) if 2 or more DMA channels are used for the DMA transfer.

(c) **Incorrect operation of HD6809, HD6844 system**

"Incorrect Operation" will occur if the number of DMA transfer Byte is more than 14 bytes. If $\overline{DMA/BREQ}$ is kept in "Low" level HD6809 performs reverse cycle steals once in 14 DMA cycles by taking back the bus control. In this case, however, the action taken by MPU is a little bit different from the DMAC.

As shown in Fig. 25, DMA controller can't stop DMA transfer (Ⓐ) by BA falling edge and excutes an extra DMA cycle during HD6809 dead cycle. So MPU cycle is excuted right after DMA cycle, the Bus confliction occurs at the beginning of MPU cycle.

(d) **How to impliment Halt Bust DMA transfer (> 14 cycles)**

Please use $\overline{HALT}$ input of HD6809 for the DMA request instead of $\overline{DMA/BREQ}$.

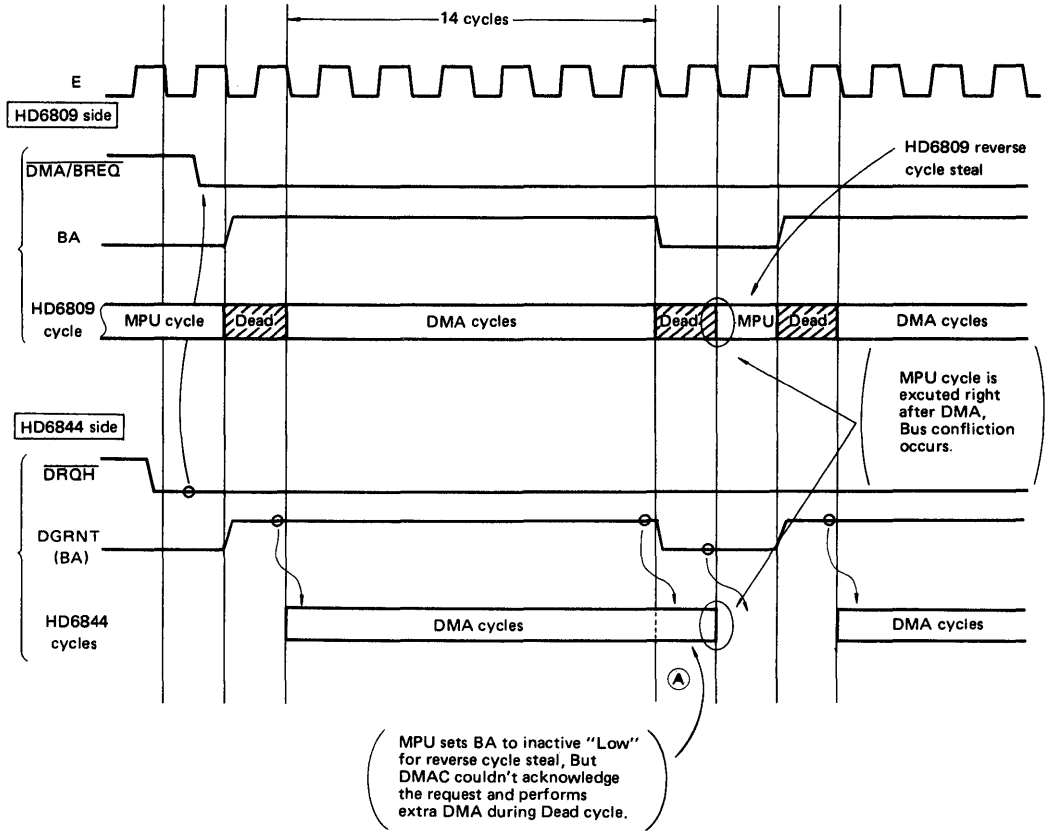


Figure 25 Comparison of HD6809, HD6844 DMA cycles

HD6809E, HD68A09E, HD68B09E—
MPU (Micro Processing Unit) —PRELIMINARY—

The HD6809E is a revolutionary high performance 8-bit microprocessor which supports modern programming techniques such as position independence, reentrancy, and modular programming.

This third-generation addition to the HMCS6800 family has major architectural improvements which include additional registers, instructions and addressing modes.

The basic instructions of any computer are greatly enhanced by the presence of powerful addressing modes. The HD6809E has the most complete set of addressing modes available on any 8-bit microprocessor today.

The HD6809E has hardware and software features which make it an ideal processor for higher level language execution or standard controller applications. External clock inputs are provided to allow synchronization with peripherals, systems or other MPUs.

HD6800 COMPATIBLE

- Hardware – Interfaces with All HMCS6800 Peripherals
- Software – Upward Source Code Compatible Instruction Set and Addressing Modes

■ **ARCHITECTURAL FEATURES**

- Two 16-bit Index Registers
- Two 16-bit Indexable Stack Pointers
- Two 8-bit Accumulators can be Concatenated to Form One 16-Bit Accumulator
- Direct Page Register Allows Direct Addressing Throughout Memory

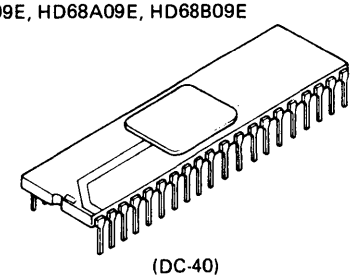
■ **HARDWARE FEATURES**

- External Clock Inputs, E and Q, Allow Synchronization
- TSC Input Controls Internal Bus Buffers
- LIC Indicates Opcode Fetch
- AVMA Allows Efficient Use of Common Resources in A Multiprocessor System
- BUSY is a Status Line for Multiprocessing
- Fast Interrupt Request Input Stacks Only Condition Code Register and Program Counter
- Interrupt Acknowledge Output Allows Vectoring By Devices
- SYNC Acknowledge Output Allows for Synchronization to External Event
- Single Bus-Cycle RESET
- Single 5-Volt Supply Operation
- NMI Blocked After RESET Until After First Load of Stack Pointer
- Early Address Valid Allows Use With Slower Memories
- Early Write-Data for Dynamic Memories

■ **SOFTWARE FEATURES**

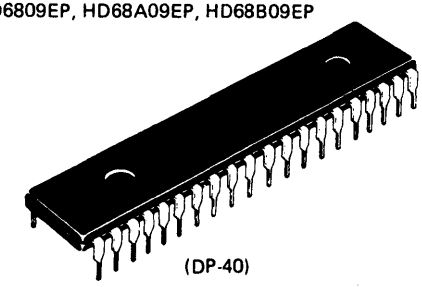
- 10 Addressing Modes
  - HMCS6800 Upward Compatible Addressing Modes
  - Direct Addressing Anywhere in Memory Map
  - Long Relative Branches
  - Program Counter Relative
  - True Indirect Addressing
  - Expanded Indexed Addressing:
    - 0, 5, 8, or 16-bit Constant Offsets
    - 8, or 16-bit Accumulator Offsets
    - Auto-Increment/Decrement by 1 or 2
- Improved Stack Manipulation
- 1464 Instruction with Unique Addressing Modes
- 8 x 8 Unsigned Multiply
- 16-bit Arithmetic

HD6809E, HD68A09E, HD68B09E



(DC-40)

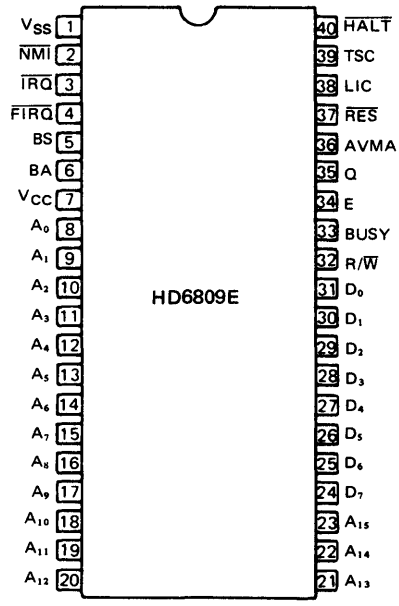
HD6809EP, HD68A09EP, HD68B09EP



(DP-40)

- Transfer/Exchange All Registers
- Push/Pull Any Registers or Any Set of Registers
- Load Effective Address

■ **PIN ARRANGEMENT**



(Top View)

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	Vcc*	−0.3 ~ +7.0	V
Input Voltage	Vin*	−0.3 ~ +7.0	V
Operating Temperature Range	Topr	−20 ~ +75	°C
Storage Temperature Range	Tstg	−55 ~ +150	°C

* With respect to Vss (SYSTEM GND)

(NOTE) Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ RECOMMENDED OPERATING CONDITIONS

Item		Symbol	min	typ	max	unit
Supply Voltage		Vcc*	4.75	5.0	5.25	V
Input Voltage	Logic, Q, $\overline{RES}$	V_{IL}*	−0.3	–	0.8	V
	E	V_{ILC}*	−0.3	–	0.4	V
	Logic	V_{IH}*	2.0	–	Vcc*	V
	$\overline{RES}$		4.0	–	Vcc*	V
	E	V_{IHC}*	Vcc* −0.75	–	Vcc* +0.3	V
Operating Temperature		Topr	−20	25	75	°C

* With respect to Vss (SYSTEM GND)

■ ELECTRICAL CHARACTERISTICS

• DC CHARACTERISTICS (V_{CC} = 5.0V ±5%, V_{SS} = 0V, Ta = −20 ~ +70°C, unless otherwise noted.)

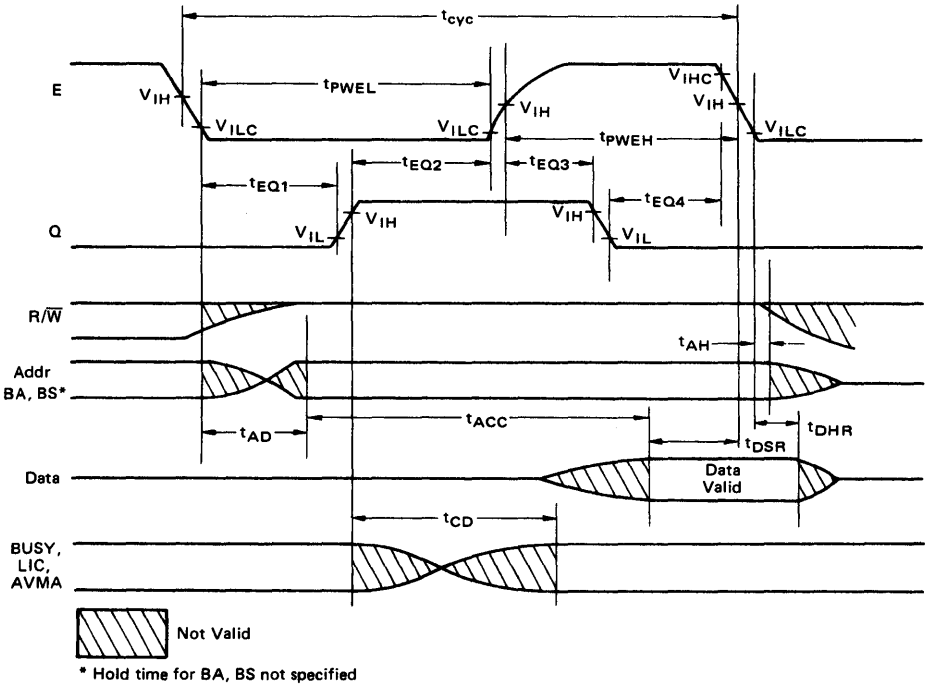
Item		Symbol	Test Condition	HD6809E			HD68A09E			HD68B09E			Unit
				min	typ*	max	min	typ*	max	min	typ*	max	
Input "High" Voltage	Logic, Q	V_{IH}		2.0	–	Vcc	2.0	–	Vcc	2.0	–	Vcc	V
	$\overline{RES}$	V_{IHR}		4.0	–	Vcc	4.0	–	Vcc	4.0	–	Vcc	V
	E	V_{IHC}		Vcc −0.75	–	Vcc +0.3	Vcc −0.75	–	Vcc +0.3	Vcc −0.75	–	Vcc +0.3	V
Input "Low" Voltage	Logic, Q, $\overline{RES}$	V_{IL}		−0.3	–	0.8	−0.3	–	0.8	−0.3	–	0.8	V
	E	V_{ILC}		−0.3	–	0.4	−0.3	–	0.4	−0.3	–	0.4	V
Input Leakage Current	Logic, Q, $\overline{RES}$	I_{in}	Vin = 0 ~ 5.25V, Vcc = max	−2.5	–	2.5	−2.5	–	2.5	−2.5	–	2.5	μA
	E			−100	–	100	−100	–	100	−100	–	100	μA
Output "High" Voltage	$D_0 \sim D_7$	V_{OH}	I_{Load} = −205μA, Vcc = min	2.4	–	–	2.4	–	–	2.4	–	–	V
	$A_0 \sim A_{15}$, R/$\overline{W}$		I_{Load} = −145μA, Vcc = min	2.4	–	–	2.4	–	–	2.4	–	–	V
	BA, BS, LIC, AVMA, BUSY		I_{Load} = −100μA, Vcc = min	2.4	–	–	2.4	–	–	2.4	–	–	V
Output Low Voltage		V_{OL}	I_{Load} = 2mA, Vcc = min	–	–	0.5	–	–	0.5	–	–	0.5	V
Power Dissipation		P_D		–	–	1.0	–	–	1.0	–	–	1.0	W
Input Capacitance	$D_0 \sim D_7$, Logic Input, Q, $\overline{RES}$	C_{in}	Vin = 0V, Ta = 25°C, f = 1MHz	–	10	15	–	10	15	–	10	15	pF
	E			–	30	50	–	30	50	–	30	50	pF
Output Capacitance	$A_0 \sim A_{15}$, R/$\overline{W}$, BA, BS, LIC, AVMA, BUSY	C_{out}	Vin = 0V, Ta = 25°C, f = 1MHz	–	10	15	–	10	15	–	10	15	pF
Frequency of Operation	E, Q	f		0.1	–	1.0	0.1	–	1.5	0.1	–	2.0	MHz
Three-State (Off State) Input Current	$D_0 \sim D_7$	I_{TSI}	Vin = 0.4 ~ 2.4V, Vcc = max	−10	–	10	−10	–	10	−10	–	10	μA
	$A_0 \sim A_{15}$, R/$\overline{W}$			−100	–	100	−100	–	100	−100	–	100	μA

* Ta = 25°C, V_{CC} = 5V

- **AC CHARACTERISTICS (V_{CC} = 5.0V ±5%, V_{SS} = 0V, Ta = -20 ~ +75°C, unless otherwise noted.)**

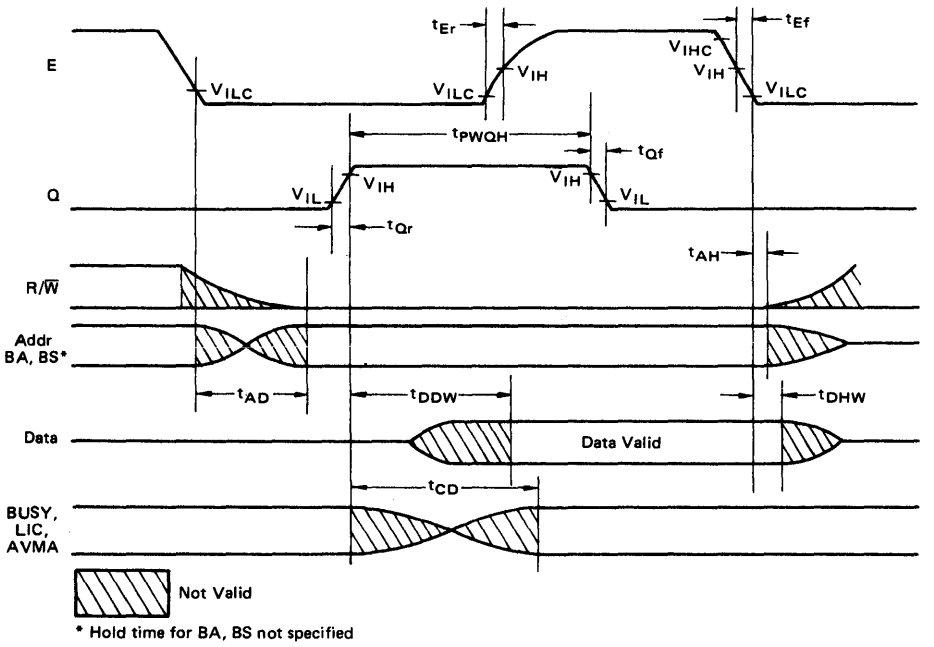
READ/WRITE TIMING

Item	Symbol	Test Condition	HD6809E			HD68A09E			HD68B09E			Unit
			min	typ	max	min	typ	max	min	typ	max	
Cycle Time	t_{cyc}	Fig. 1, 2, 7 ~ 11, 13, 14 and 17	1000	–	10000	667	–	10000	500	–	10000	ns
Peripheral Read Access Times $t_{cyc} - t_{Ef} - t_{AD} - t_{DSR} = t_{ACC}$	t_{ACC}		695	–	–	440	–	–	330	–	–	ns
Data Setup Time (Read)	t_{DSR}		80	–	–	60	–	–	40	–	–	ns
Input Data Hold Time	t_{DHR}		10	–	–	10	–	–	10	–	–	ns
Output Data Hold Time	t_{DHW}		30	–	–	30	–	–	30	–	–	ns
Address Hold Time (Address, R/$\overline{W}$)	t_{AH}		20	–	–	20	–	–	20	–	–	ns
Address Delay	t_{AD}		–	–	200	–	–	140	–	–	110	ns
Data Delay Time (Write)	t_{DDW}		–	–	200	–	–	140	–	–	110	ns
E Clock "Low"	t_{PWEL}		450	–	9500	295	–	9500	210	–	9500	ns
E Clock "High" (Measured at V_{IH})	t_{PWEH}		450	–	9500	280	–	9500	220	–	9500	ns
E Rise and Fall Time	t_{Er}, t_{Ef}		–	–	25	–	–	25	–	–	20	ns
Q Clock "High"	t_{PWQH}		450	–	9500	280	–	9500	220	–	9500	ns
Q Rise and Fall Time	t_{Qr}, t_{Qf}		–	–	25	–	–	25	–	–	20	ns
E "Low" to Q Rising	t_{EQ1}		200	–	–	130	–	–	100	–	–	ns
Q "High" to E Rising	t_{EQ2}		200	–	–	130	–	–	100	–	–	ns
E "High" to Q Falling	t_{EQ3}		200	–	–	130	–	–	100	–	–	ns
Q "Low" to E Falling	t_{EQ4}		200	–	–	130	–	–	100	–	–	ns
Interrupts $\overline{HALT}$, $\overline{RES}$ and TSC Setup Time	t_{PCS}		200	–	–	140	–	–	110	–	–	ns
TSC Drive to Valid Logic Levels	t_{TSA}		–	–	210	–	–	150	–	–	120	ns
TSC Release MOS Buffers to High Impedance	t_{TSR}		–	–	200	–	–	140	–	–	110	ns
TSC Three-State Delay	t_{TSD}		–	–	120	–	–	85	–	–	80	ns
Control Delay (BUSY, LIC, AVMA)	t_{CD}		–	–	300	–	–	250	–	–	200	ns
Processor Control Rise/Fall	t_{PCr}, t_{PCf}		–	–	100	–	–	100	–	–	100	ns



(NOTE) Waveform measurements for all inputs and outputs are specified at logic "High" = V_{IHmin} and logic "Low" = V_{ILmax} unless otherwise specified.

Figure 1 Read Data from Memory or Peripherals



(NOTE) Waveform measurements for all inputs and outputs are specified at logic "High" = V_{IHmin} and logic "Low" = V_{ILmax} unless otherwise specified.

Figure 2 Write Data to Memory or Peripherals

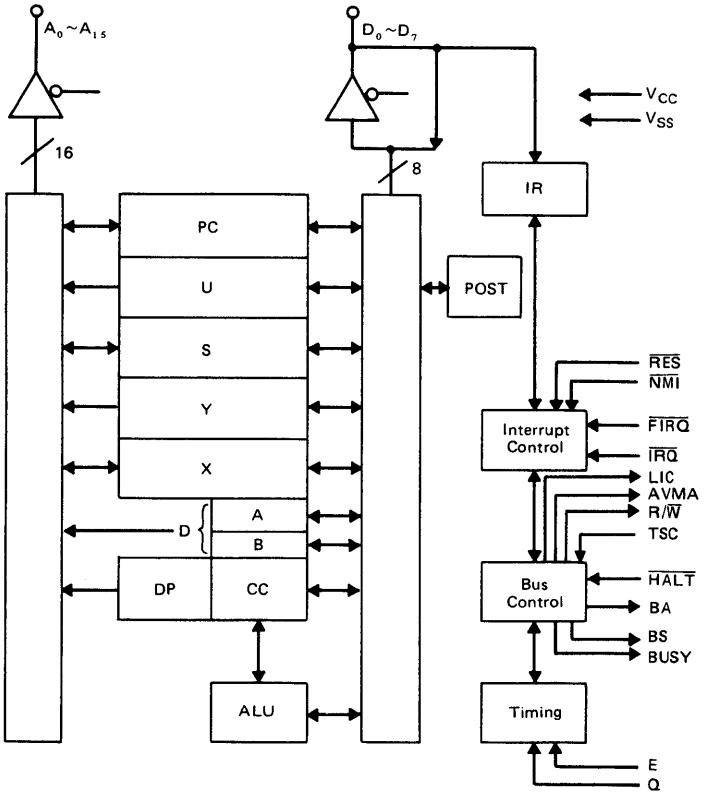
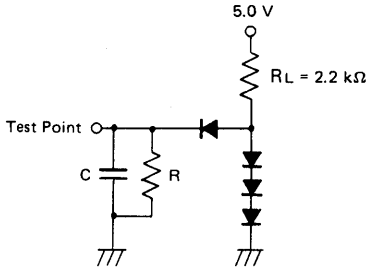


Figure 3 HD6809E Expanded Block Diagram



C = 30 pF for BA, BS, LIC, AVMA, BUSY
130 pF for $D_0 \sim D_7$
90 pF for $A_0 \sim A_{15}$, R/$\overline{W}$

R = 11.7 kΩ for $D_0 \sim D_7$
16.5 kΩ for $A_0 \sim A_{15}$, R/$\overline{W}$
24 kΩ for BA, BS
LIC, AVMA, BUSY

All diodes are 1S2074 (H) or equivalent.
C includes stray capacitance.

Figure 4 Bus Timing Test Load

■ PROGRAMMING MODEL

As shown in Figure 5, the HD6809E adds three registers to the set available in the HD6800. The added registers include a Direct Page Register, the User Stack pointer and a second Index Register.

• Accumulators (A, B, D)

The A and B registers are general purpose accumulators which are used for arithmetic calculations and manipulation of data.

Certain instructions concatenate the A and B registers to form a single 16-bit accumulator. This is referred to as the D Register, and is formed with the A Register as the most significant byte.

• Direct Page Register (DP)

The Direct Page Register of the HD6809E serves to enhance the Direct Addressing Mode. The content of this register appears at the higher address outputs ($A_8 \sim A_{15}$) during direct addressing instruction execution. This allows the direct mode to be used at any place in memory, under program control. To ensure HMCS6800 compatibility, all bits of this register are cleared during Processor Reset.

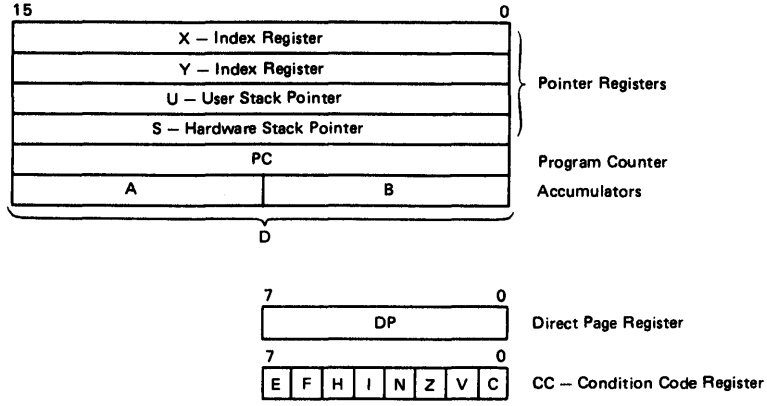


Figure 5 Programming Model of The Microprocessing Unit

- **Index Registers (X, Y)**

The Index Registers are used in indexed mode of addressing. The 16-bit address in this register takes part in the calculation of effective addresses. This address may be used to point to data directly or may be modified by an optional constant or register offset. During some indexed modes, the contents of the index register are incremented and decremented to point to the next item of tabular type data. All four pointer registers (X, Y, U, S) may be used as index registers.

- **Stack Pointer (U, S)**

The Hardware Stack Pointer (S) is used automatically by the processor during subroutine calls and interrupts. The User Stack Pointer (U) is controlled exclusively by the programmer thus allowing arguments to be passed to and from subroutines with ease. The U-register is frequently used as a stack marker. Both Stack Pointers have the same indexed mode addressing capabilities as the X and Y registers, but also support **Push** and **Pull** instructions. This allows the HD6809E to be used efficiently as a stack processor, greatly enhancing its ability to support higher level languages and modular programming.

(NOTE) The stack pointers of the HD6809E point to the top of the stack, in contrast to the HD6800 stack pointer, which pointed to the next free location on stack.

- **Program Counter (PC)**

The Program Counter is used by the processor to point to the address of the next instruction to be executed by the processor. Relative Addressing is provided allowing the Program Counter to be used like an index register in some situations.

- **Condition Code Register (CC)**

The Condition Code Register defines the state of the processor at any given time. See Figure 6.

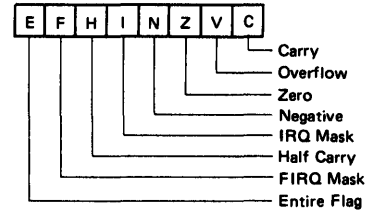


Figure 6 Condition Code Register Format

■ CONDITION CODE REGISTER DESCRIPTION

- **Bit 0 (C)**

Bit 0 is the carry flag, and is usually the carry from the binary ALU. C is also used to represent a 'borrow' from subtract like instructions (CMP, NEG, SUB, SBC) and is the complement of the carry from the binary ALU.

- **Bit 1 (V)**

Bit 1 is the overflow flag, and is set to a one by an operation which causes a signed two's complement arithmetic overflow. This overflow is detected in an operation in which the carry from the MSB in the ALU does not match the carry from the MSB-1.

- **Bit 2 (Z)**

Bit 2 is the zero flag, and is set to a one if the result of the previous operation was identically zero.

- **Bit 3 (N)**

Bit 3 is the negative flag, which contains exactly the value of the MSB of the result of the preceding operation. Thus, a negative two's-complement result will leave N set to a one.

- **Bit 4 (I)**

Bit 4 is the $\overline{IRQ}$ mask bit. The processor will not recognize interrupts from the $\overline{IRQ}$ line if this bit is set to a one. $\overline{NMI}$, $\overline{FIRQ}$, $\overline{IRQ}$, $\overline{RES}$ and SWI all set I to a one; SWI2 and SWI3 do not affect I.

- **Bit 5 (H)**

Bit 5 is the half-carry bit, and is used to indicate a carry from bit 3 in the ALU as a result of an 8-bit addition only (ADC or ADD). This bit is used by the DAA instruction to perform a BCD decimal add adjust operation. The state of this flag is undefined in all subtract-like instructions.

- **Bit 6 (F)**

Bit 6 is the $\overline{FIRQ}$ mask bit. The processor will not recognize interrupts from the $\overline{FIRQ}$ line if this bit is a one. $\overline{NMI}$, $\overline{FIRQ}$, SWI, and $\overline{RES}$ all set F to a one. $\overline{IRQ}$, SWI2 and SWI3 do not affect F.

- **Bit 7 (E)**

Bit 7 is the entire flag, and when set to a one indicates that the complete machine state (all the registers) was stacked, as opposed to the subset state (PC and CC). The E bit of the stacked CC is used on a return from interrupt (RTI) to determine the extent of the unstacking. Therefore, the current E left in the Condition Code Register represents past action.

■ HD6809E MPU SIGNAL DESCRIPTION

- **Power (V_{SS}, V_{CC})**

Two pins are used to supply power to the part: V_{SS} is ground or 0 volts, while V_{CC} is +5.0 V ±5%.

- **Address Bus ($A_0 \sim A_{15}$)**

Sixteen pins are used to output address information from the MPU onto the Address Bus. When the processor does not require the bus for a data transfer, it will output address $FFFF_{16}$, R/$\overline{W}$ = "High", and BS = "Low"; this is a "dummy access" or $\overline{VMA}$ cycle. All address bus drivers are made high-impedance when output Bus Available (BA) is "High" or when TSC is asserted. Each pin will drive one Schottky TTL load or four LS TTL loads, and 90 pF. Refer to Figures 1 and 2.

- **Data Bus ($D_0 \sim D_7$)**

These eight pins provide communication with the system bi-directional data bus. Each pin will drive one Schottky TTL load or four LS TTL loads, and 130 pF.

- **Read/Write (R/$\overline{W}$)**

This signal indicates the direction of data transfer on the data bus. A "Low" indicates that the MPU is writing data onto the data bus. R/$\overline{W}$ is made high impedance when BA is "High" or when TSC is asserted. Refer to Figures 1 and 2.

- **$\overline{RES}$**

A "Low" level on this Schmitt-trigger input for greater than one bus cycle will reset the MPU, as shown in Figure 7. The Reset vectors are fetched from locations $FFFE_{16}$ and $FFFF_{16}$ (Table 1) when Interrupt Acknowledge is true, ($\overline{BA}$ · BS = 1). During initial power-on, the Reset line should be held "Low" until the clock input signals are fully operational.

Because the HD6809E Reset pin has a Schmitt-trigger input with a threshold voltage higher than that of standard peripherals, a simple R/C network may be used to reset the entire system. This higher threshold voltage ensures that all peripherals are out of the reset state before the Processor.

Table 1 Memory Map for Interrupt Vectors

Memory Map for Vector Locations		Interrupt Vector Description
MS	LS	
FFFE	FFFF	$\overline{RES}$
FFFC	FFFD	$\overline{NMI}$
FFFA	FFFB	SWI
FFF8	FFF9	$\overline{IRQ}$
FFF6	FFF7	$\overline{FIRQ}$
FFF4	FFF5	SWI2
FFF2	FFF3	SWI3
FFF0	FFF1	Reserved

- **$\overline{HALT}$**

A "Low" level on this input pin will cause the MPU to stop running at the end of the present instruction and remain halted indefinitely without loss of data. When halted, the BA output is driven "High" indicating the buses are high impedance. BS is also "High" which indicates the processor is in the Halt state. While halted, the MPU will not respond to external real-time requests ($\overline{FIRQ}$, $\overline{IRQ}$) although $\overline{NMI}$ or $\overline{RES}$ will be latched for later response. During the Halt state Q and E should continue to run normally. A halted state (BA · BS = 1) can be achieved by pulling $\overline{HALT}$ "Low" while $\overline{RES}$ is still "Low". See Figure 8.

- **Bus Available, Bus Status (BA, BS)**

The Bus Available output is an indication of an internal control signal which makes the MOS buses of the MPU high impedance. When BA goes "Low", a dead cycle will elapse before the MPU acquires the bus. BA will not be asserted when TSC is active, thus allowing dead cycle consistency.

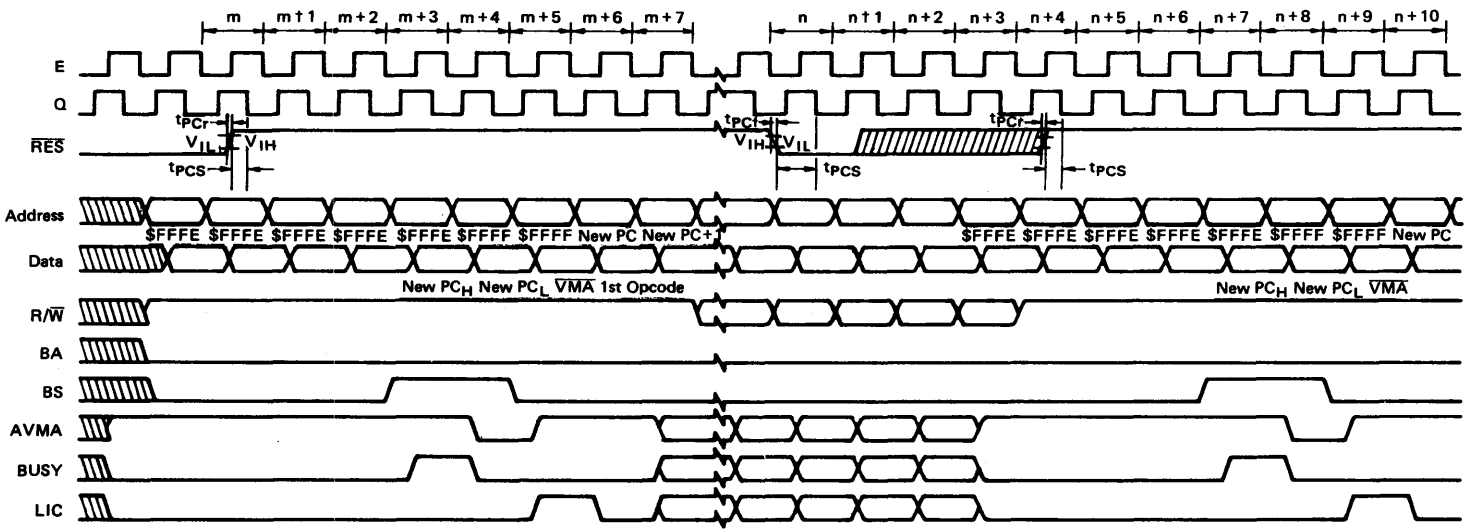
The Bus Status output signal, when decoded with BA, represents the MPU state (valid with leading edge of Q).

MPU State		MPU State Definition
BA	BS	
0	0	Normal (Running)
0	1	Interrupt or RESET Acknowledge
1	0	SYNC Acknowledge
1	1	HALT Acknowledge

Interrupt Acknowledge is indicated during both cycles of a hardware-vector-fetch ($\overline{RES}$, $\overline{NMI}$, $\overline{FIRQ}$, $\overline{IRQ}$, SWI, SWI2, SWI3). This signal, plus decoding of the lower four address lines, can provide the user with an indication of which interrupt level is being serviced and allow vectoring by device. See Table 1.

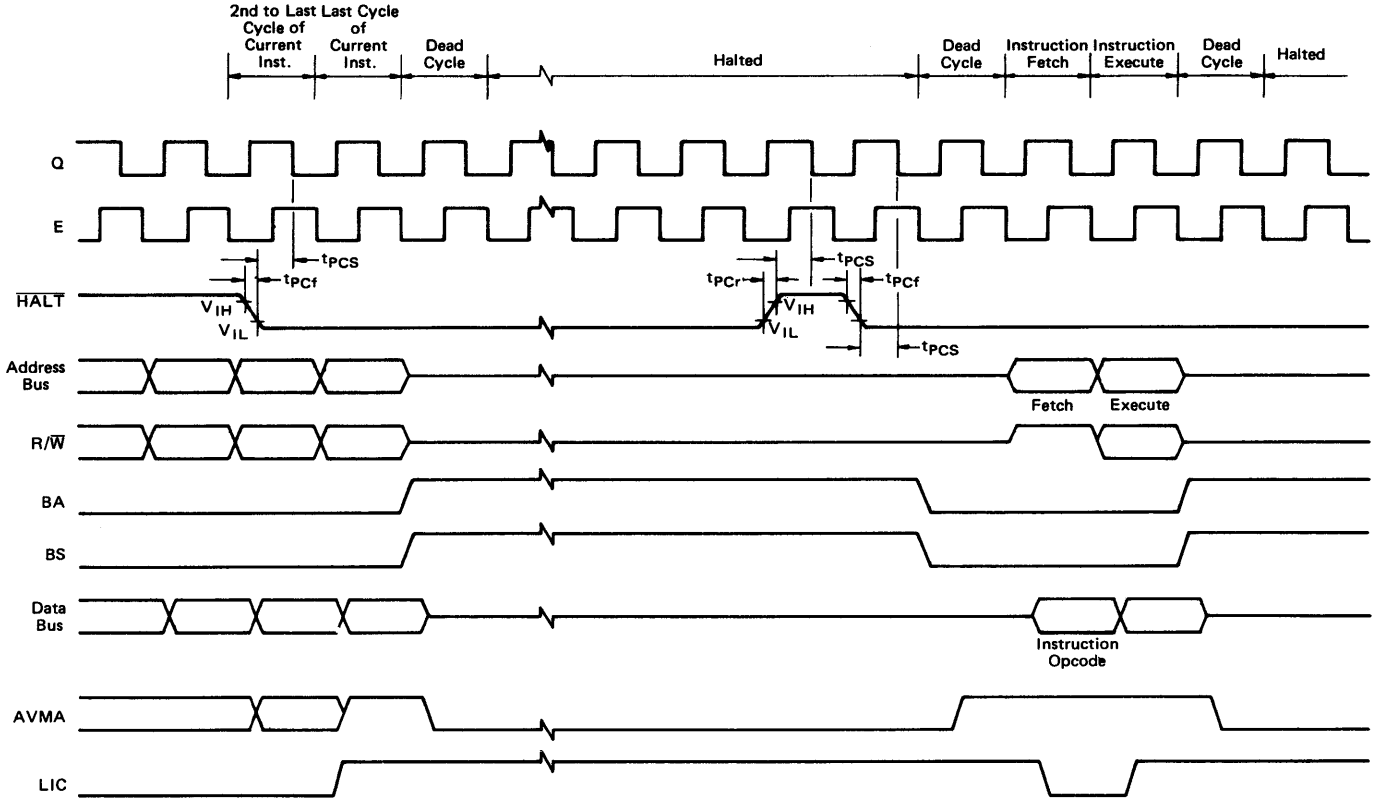
Sync Acknowledge is indicated while the MPU is waiting for external synchronization on an interrupt line.

Halt Acknowledge is indicated when the HD6809E is in a Halt condition.



(NOTE) Waveform measurements for all inputs and outputs are specified at logic "High" = V_{IHmin} and logic "Low" = V_{ILmax} unless otherwise specified.

Figure 7 $\overline{RES}$ Timing



(NOTE) Waveform measurements for all inputs and outputs are specified at logic "High" = V_{IHmin} and logic "Low" = V_{ILmax} unless otherwise specified.

Figure 8 $\overline{\text{HALT}}$ and Single Instruction Execution for System Debug

- **Non Maskable Interrupt ($\overline{NMI}$)***

A negative transition on this input requests that a non-maskable interrupt sequence be generated. A non-maskable interrupt cannot be inhibited by the program, and also has a higher priority than $\overline{FIRQ}$, $\overline{IRQ}$ or software interrupts. During recognition of an $\overline{NMI}$, the entire machine state is saved on the hardware stack. After reset, an $\overline{NMI}$ will not be recognized until the first program load of the Hardware Stack Pointer (S). The pulse width of $\overline{NMI}$ low must be at least one E cycle. If the $\overline{NMI}$ input does not meet the minimum set up with respect to Q, the interrupt will not be recognized until the next cycle. See Figure 9.

- **Fast-Interrupt Request ($\overline{FIRQ}$)***

A "Low" level on this input pin will initiate a fast interrupt sequence, provided its mask bit (F) in the CC is clear. This sequence has priority over the standard Interrupt Request ($\overline{IRQ}$), and is fast in the sense that it stacks only the contents of the condition code register and the program counter. The interrupt service routine should clear the source of the interrupt before doing an RTI. See Figure 10.

- **Interrupt Request ($\overline{IRQ}$)***

A "Low" level input on this pin will initiate an Interrupt Request sequence provided the mask bit (I) in the CC is clear. Since $\overline{IRQ}$ stacks the entire machine state it provides a slower response to interrupts than $\overline{FIRQ}$. $\overline{IRQ}$ also has a lower priority than $\overline{FIRQ}$. Again, the interrupt service routine should clear the source of the interrupt before doing an RTI. See Figure 9.

* $\overline{NMI}$, $\overline{FIRQ}$, and $\overline{IRQ}$ requests are sampled on the falling edge of Q. One cycle is required for synchronization before these interrupts are recognized. The pending interrupt(s) will not be serviced until completion of the current instruction unless a SYNC or CWAI condition is present. If $\overline{IRQ}$ and $\overline{FIRQ}$ do not remain "Low" until completion of the current instruction they may not be recognized. However, $\overline{NMI}$ is latched and need only remain "Low" for one cycle.

- **Clock Inputs E, Q**

E and Q are the clock signals required by the HD6809E. Q must lead E; that is, a transition on Q must be followed by a similar transition on E after a minimum delay. Addresses will be valid from the MPU, t_{AD} after the falling edge of E, and data will be latched from the bus by the falling edge of E. While the Q input is fully TTL compatible, the E input directly drives internal MOS circuitry and, thus, requires levels above normal TTL levels. This approach minimizes clock skew inherent with an internal buffer. Timing and waveforms for E and Q are shown in Figures 1 and 2 while Figure 11 shows a simple clock generator for the HD6809E. Proper operation of the MPU is not guaranteed unless t_{cyc}, t_{PWEL}, t_{PWEH} timings are met.

- **BUSY**

Busy will be "High" for the read and modify cycles of a read-modify-write instruction and during the access of the first byte of a double-byte operation (e.g., LDX, STD, ADDD). Busy is also "High" during the first byte of any indirect or other vector fetch (e.g., jump extended, SWI indirect etc.).

In a multi-processor system, busy indicates the need to defer the rearbitration of the next bus cycle to insure the integrity of the above operations. This difference provides the indivisible memory access required for a "test-and-set" primitive, using any one of several read-modify-write instructions.

Busy does not become active during PSH or PUL operations. A typical read-modify-write instruction (ASL) is shown in Figure 12. Timing information is given in Figure 13. Busy is valid t_{CD} after the rising edge of Q.

- **AVMA**

AVMA is the Advanced VMA signal and indicates that the MPU will use the bus in the following bus cycle. The predictive nature of the AVMA signal allows efficient shared-bus multiprocessor systems. AVMA is "Low" when the MPU is in either a HALT or SYNC state. AVMA is valid t_{CD} after the rising edge of Q.

- **LIC**

LIC (Last Instruction Cycle) is "High" during the last cycle of every instruction, and its transition from "High" to "Low" will indicate that the first byte of an opcode will be latched at the end of the present bus cycle. LIC will be "High" when the MPU is Halted at the end of an instruction, (i.e., not in CWAI or RESET) in SYNC state or while stacking during interrupts. LIC is valid t_{CD} after the rising edge of Q.

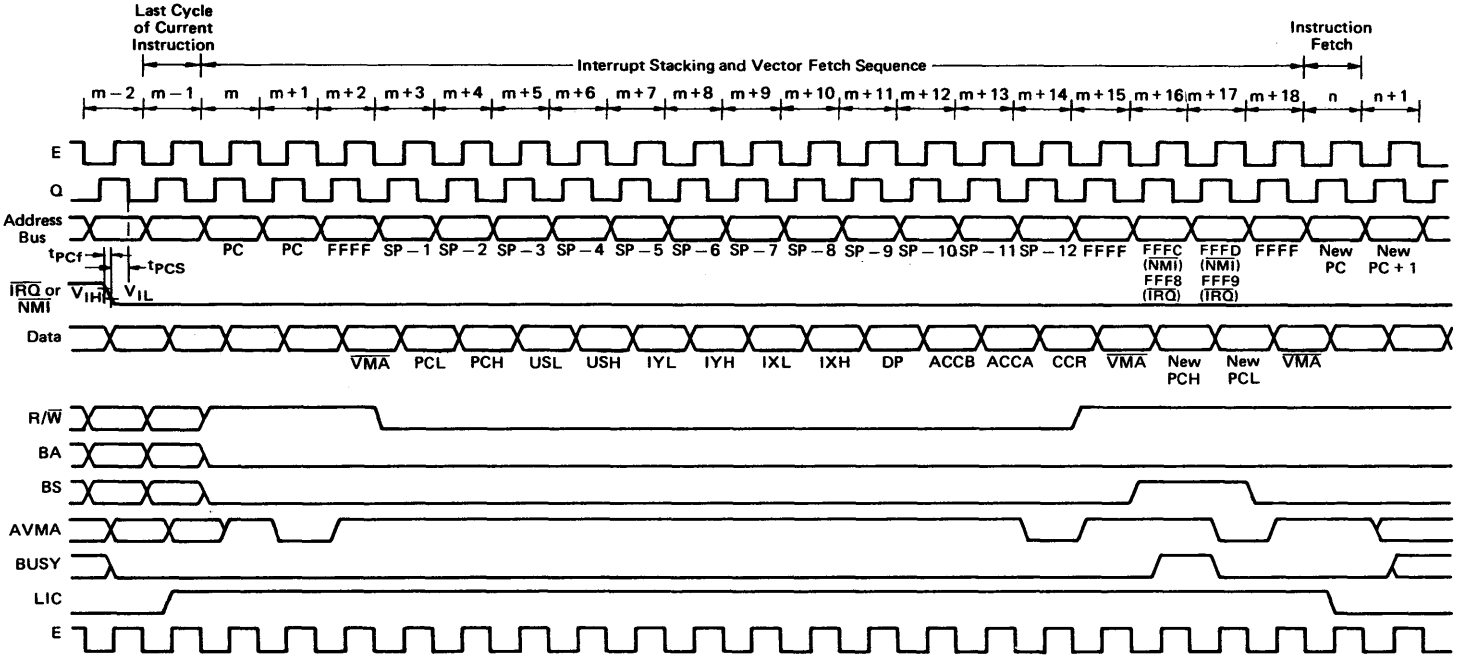
- **TSC**

TSC (Three-State Control) will cause MOS address, data, and R/$\overline{W}$ buffers to assume a high-impedance state. The control signals (BA, BS, BUSY, AVMA and LIC) will not go to the high-impedance state. TSC is intended to allow a single bus to be shared with other bus masters (processors or DMA controllers).

While E is "Low", TSC controls the address buffers and R/$\overline{W}$ directly. The data bus buffers during a write operation are in a high-impedance state until Q rises at which time, if TSC is true, they will remain in a high-impedance state. If TSC is held beyond the rising edge of E, then it will be internally latched, keeping the bus drivers in a high-impedance state for the remainder of the bus cycle. See Figure 14.

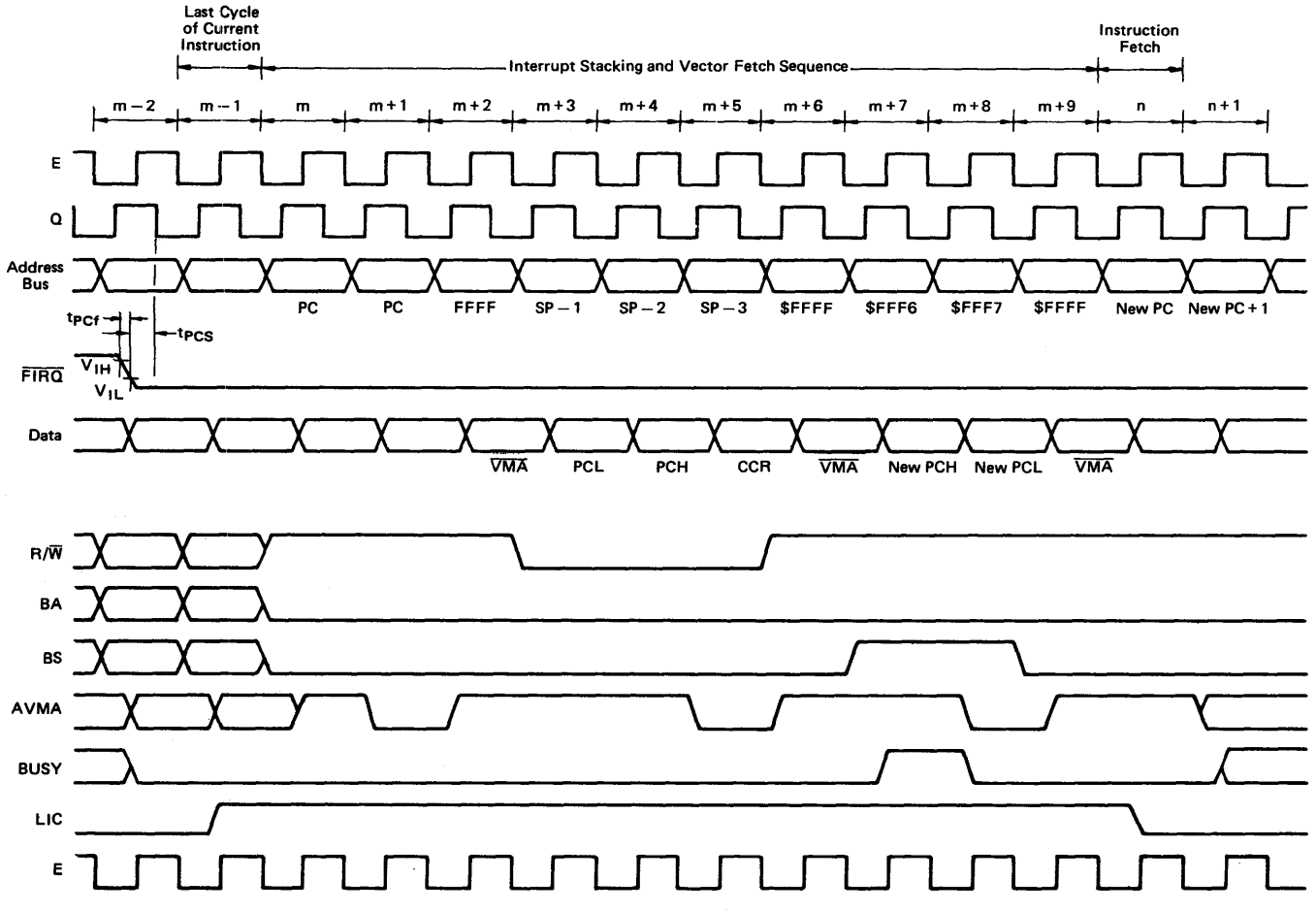
- **MPU Operation**

During normal operation, the MPU fetches an instruction from memory and then executes the requested function. This sequence begins after $\overline{RES}$ and is repeated indefinitely unless altered by a special instruction or hardware occurrence. Software instructions that alter normal MPU operation are: SWI, SWI2, SWI3, CWAI, RTI and SYNC. An interrupt or $\overline{HALT}$ input can also alter the normal execution of instructions. Figure 15 illustrates the flow chart for the HD6809E.



(NOTE) Waveform measurements for all inputs and outputs are specified at logic "High" = V_{IHmin} and logic "Low" = V_{ILmax} unless otherwise specified. E clock shown for reference only.

Figure 9 $\overline{IRQ}$ and $\overline{NMI}$ Interrupt Timing



(NOTE) Waveform measurements for all inputs and outputs are specified at logic "High" = V_{IHmin} and logic "Low" = V_{ILmax} unless otherwise specified. E clock shown for reference only.

Figure 10 $\overline{\text{FIRQ}}$ Interrupt Timing

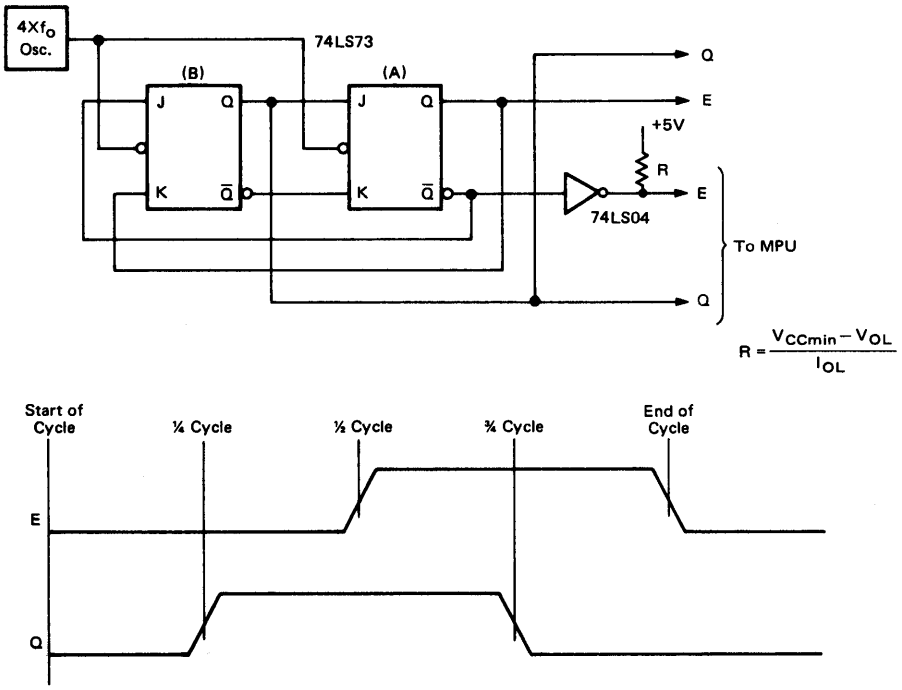


Figure 11 HD6809E Clock Generator

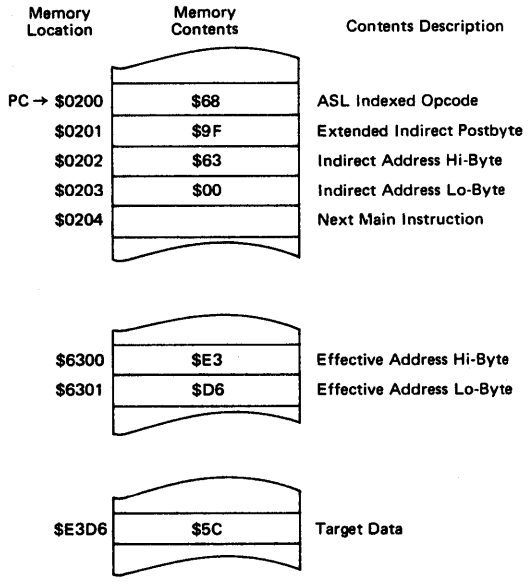
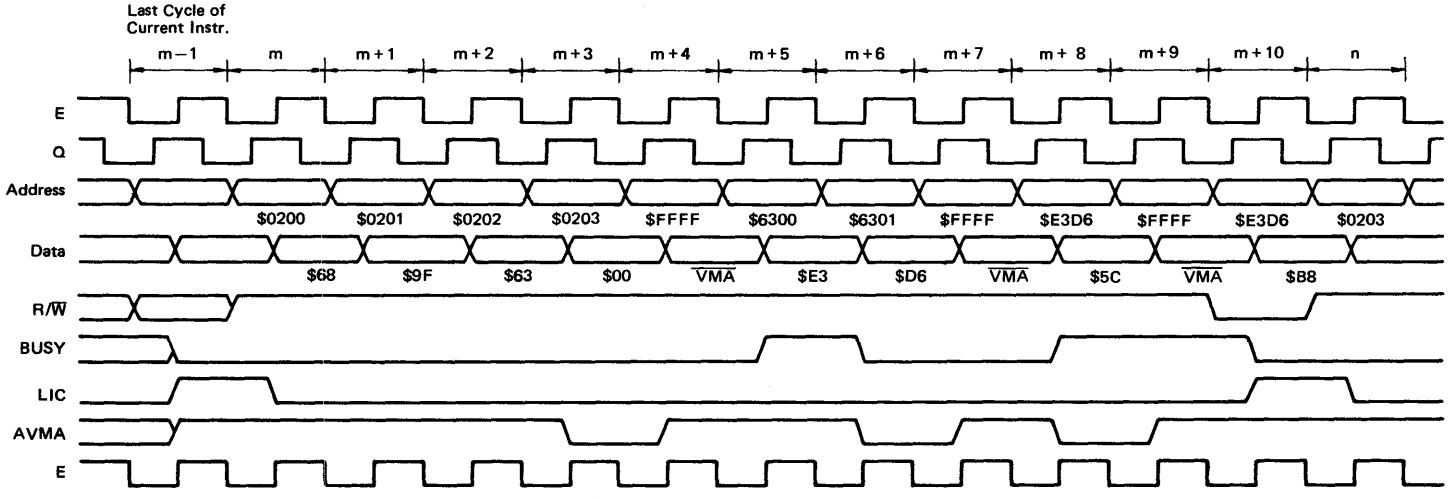
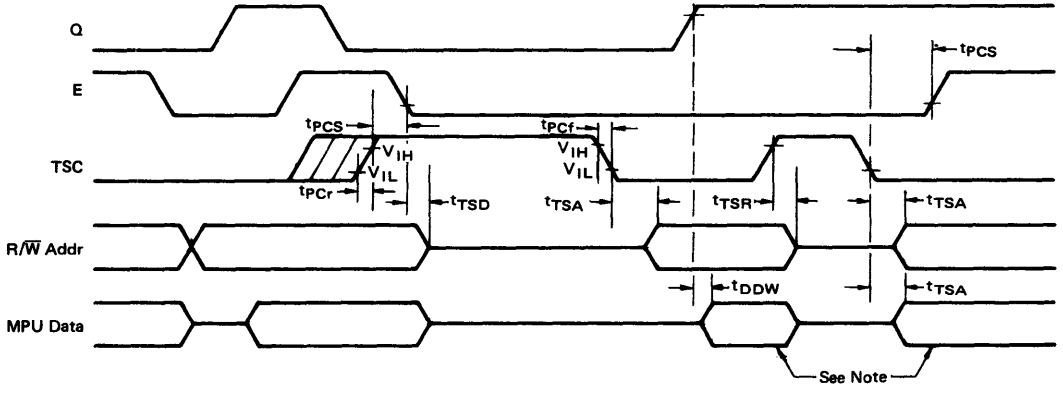


Figure 12 Read Modify Write Instruction Example (ASL Extended Indirect)



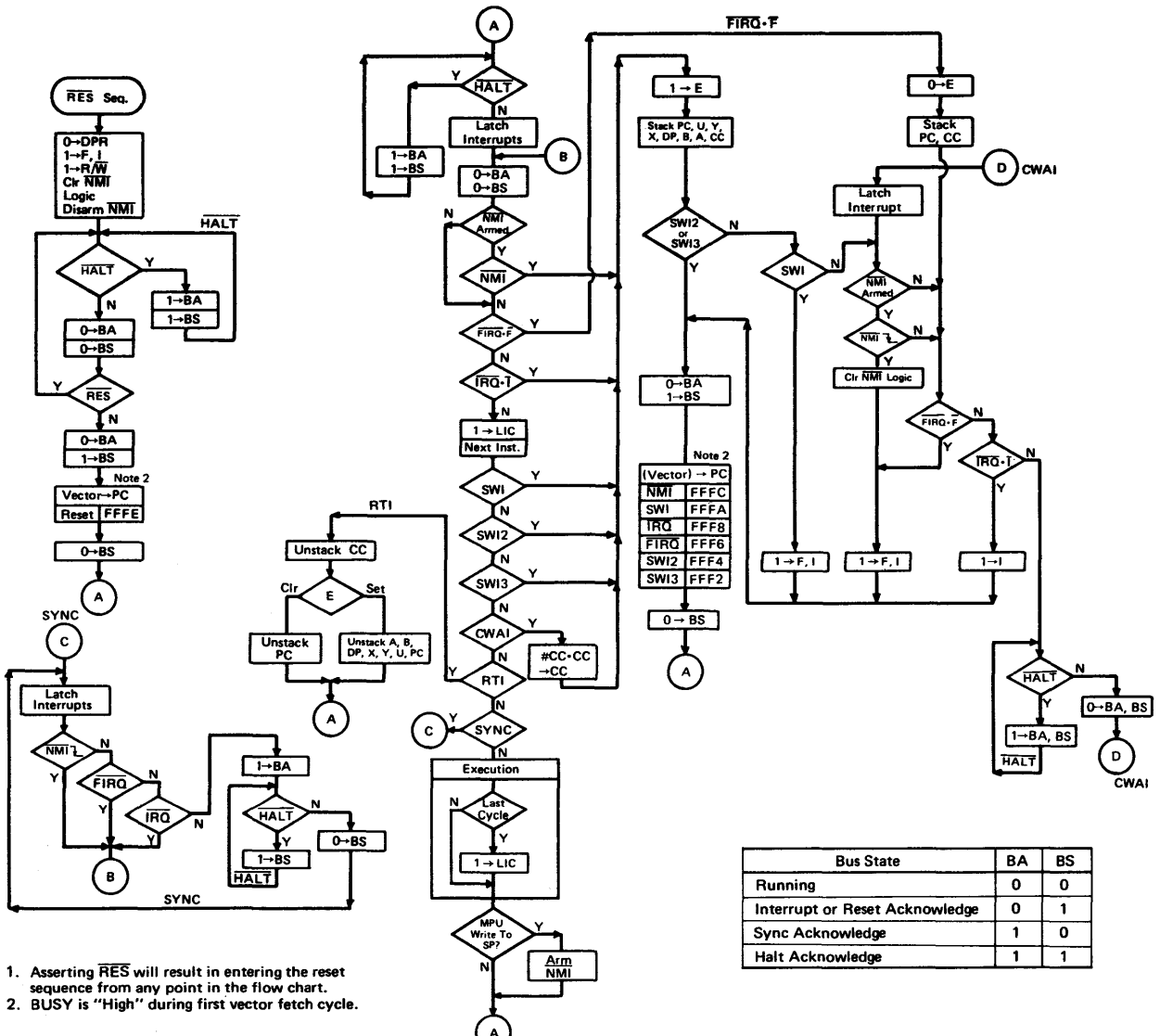
(NOTE) Waveform measurements for all inputs and outputs are specified at logic "High" = V_{IHmin} and logic "Low" = V_{ILmax} unless otherwise specified.

Figure 13 BUSY Timing



(NOTES) Data will be asserted by the MPU only during the interval while R/W is "Low" and E or Q is "High".
Waveform measurements for all inputs and outputs are specified at logic "High" = V_{IHmin} and logic "Low" = V_{ILmax} unless otherwise specified.

Figure 14 TSC Timing



Bus State	BA	BS
Running	0	0
Interrupt or Reset Acknowledge	0	1
Sync Acknowledge	1	0
Halt Acknowledge	1	1

(NOTES) 1. Asserting $\overline{RES}$ will result in entering the reset sequence from any point in the flow chart.
2. BUSY is "High" during first vector fetch cycle.

Figure 15 Flowchart for HD6809E Instruction

■ ADDRESSING MODES

The basic instructions of any computer are greatly enhanced by the presence of powerful addressing modes. The HD6809E has the most complete set of addressing modes available on any microcomputer today. For example, the HD6809E has 59 basic instructions; however, it recognizes 1464 different variations of instructions and addressing modes. The addressing modes support modern programming techniques. The following addressing modes are available on the HD6809E:

(1) Implied (Includes Accumulator)
(2) Immediate
(3) Extended
(4) Extended Indirect
(5) Direct
(6) Register
(7) Indexed
    - Zero-Offset
    - Constant Offset
    - Accumulator Offset
    - Auto Increment/Decrement
(8) Indexed Indirect
(9) Relative
(10) Program Counter Relative

• Implied (Includes Accumulator)

In this addressing mode, the opcode of the instruction contains all the address information necessary. Examples of Implied Addressing are: ABX, DAA, SWI, ASRA, and CLRB.

• Immediate Addressing

In Immediate Addressing, the effective address of the data is the location immediately following the opcode (i.e., the data to be used in the instruction immediately follows the opcode of the instruction). The HD6809E uses both 8 and 16-bit immediate values depending on the size of argument specified by the opcode. Examples of instructions with immediate Addressing are:

```
LDA  #$20
LDX  #$F000
LDY  #CAT
```

(NOTE) # signifies immediate addressing, $ signifies hexadecimal value.

• Extended Addressing

In Extended Addressing, the contents of the two bytes immediately following the opcode fully specify the 16-bit effective address used by the instruction. Note that the address generated by an extended instruction defines an absolute address and is not position independent. Examples of Extended Addressing include:

```
LDA  CAT
STX  MOUSE
LDD  $2000
```

• Extended Indirect

As a special case of indexed addressing (discussed below), one level of indirection may be added to Extended Addressing. In Extended Indirect, the two bytes following the postbyte of an Indexed instruction contain the address of the data.

```
LDA  [CAT]
LDX  [$FFFE]
STU  [DOG]
```

• Direct Addressing

Direct addressing is similar to extended addressing except that only one byte of address follows the opcode. This byte specifies the lower 8 bits of the address to be used. The upper 8 bits of the address are supplied by the direct page register. Since only one byte of address is required in direct addressing, this mode requires less memory and executes faster than extended addressing. Of course, only 256 locations (one page) can be accessed without redefining the contents of the DP register. Since the DP register is set to $00 on Reset, direct addressing on the HD6809E is compatible with direct addressing on the HMCS6800. Indirection is not allowed in direct addressing. Some examples of direct addressing are:

```
LDA   $30
SETDP $10     (Assembler directive)
LDB   $1030
LDD   <CAT
```

(NOTE) < is an assembler directive which forces direct addressing.

• Register Addressing

Some opcodes are followed by a byte that defines a register or set of registers to be used by the instruction. This is called a postbyte. Some examples of register addressing are:

```
TFR   X, Y       Transfer X into Y
EXG   A, B       Exchanges A with B
PSHS  A, B, X, Y Push Y, X, B and A onto S
PULU  X, Y, D    Pull D, X, and Y from U
```

• Indexed Addressing

In all indexed addressing, one of the pointer registers (X, Y, U, S, and sometimes PC) is used in a calculation of the effective address of the operand to be used by the instruction. Five basic types of indexing are available and are discussed below. The postbyte of an indexed instruction specifies the basic type and variation of the addressing mode as well as the pointer register to be used. Figure 16 lists the legal formats for the postbyte. Table 2 gives the assembler form and the number of cycles and bytes added to the basic values for indexed addressing for each variation.

Post-Byte Register Bit								Indexed Addressing Mode
7	6	5	4	3	2	1	0	
0	R	R	d	d	d	d	d	EA = ,R + 5 Bit Offset
1	R	R	0	0	0	0	0	,R +
1	R	R	i	0	0	0	1	,R + +
1	R	R	0	0	0	1	0	, -R
1	R	R	i	0	0	1	1	, - - R
1	R	R	i	0	1	0	0	EA = ,R + 0 Offset
1	R	R	i	0	1	0	1	EA = ,R + ACCB Offset
1	R	R	i	0	1	1	0	EA = ,R + ACCA Offset
1	R	R	i	1	0	0	0	EA = , R + 8 Bit Offset
1	R	R	i	1	0	0	1	EA = ,R + 16 Bit Offset
1	R	R	i	1	0	1	1	EA = ,R + D Offset
1	x	x	i	1	1	0	0	EA = ,PC + 8 Bit Offset
1	x	x	i	1	1	0	1	EA = ,PC + 16 Bit Offset
1	R	R	i	1	1	1	1	EA = [,Address]

Addressing Mode Field
Indirect Field (Sigh bit when b7 = 0)
Register Field : RR
00 = X
01 = Y
10 = U
11 = S

x = Don't Care
d = Offset Bit
i = { 0 = Non Indirect, 1 = Indirect }

Figure 16 Index Addressing Postbyte Register Bit Assignments

Table 2 Indexed Addressing Mode

Type	Forms	Non Indirect				Indirect			
		Assembler Form	Postbyte OP Code	+ ~	+ #	Assembler Form	Postbyte OP Code	+ ~	+ #
Constant Offset From R (2's Complement Offsets)	No Offset	,R	1RR00100	0	0	[,R]	1RR10100	3	0
	5 Bit Offset	n, R	0RRnnnnn	1	0	defaults to 8-bit			
	8 Bit Offset	n, R	1RR01000	1	1	[n, R]	1RR11000	4	1
	16 Bit Offset	n, R	1RR01001	4	2	[n, R]	1RR11001	7	2
Accumulator Offset From R (2's Complement Offsets)	A Register Offset	A, R	1RR00110	1	0	[A, R]	1RR10110	4	0
	B Register Offset	B, R	1RR00101	1	0	[B, R]	1RR10101	4	0
	D Register Offset	D, R	1RR01011	4	0	[D, R]	1RR11011	7	0
Auto Increment/Decrement R	Increment By 1	,R +	1RR00000	2	0	not allowed			
	Increment By 2	,R + +	1RR00001	3	0	[,R + +]	1RR10001	6	0
	Decrement By 1	, - R	1RR00010	2	0	not allowed			
	Decrement By 2	, - - R	1RR00011	3	0	[, - - R]	1RR10011	6	0
Constant Offset From PC (2's Complement Offsets)	8 Bit Offset	n, PCR	1xx01100	1	1	[n, PCR]	1xx11100	4	1
	16 Bit Offset	n, PCR	1xx01101	5	2	[n, PCR]	1xx11101	8	2
Extended Indirect	16 Bit Address	–	–	–	–	[n]	10011111	5	2

R = X, Y, U or S
x = Don't Care

RR:
00 = X
01 = Y
10 = U
11 = S

$\overset{+}{\sim}$ and $\overset{+}{\#}$ indicate the number of additional cycles and bytes for the particular variation.

Zero-Offset Indexed

In this mode, the selected pointer register contains the effective address of the data to be used by the instruction. This is the fastest indexing mode.

Examples are:

```
LDD   0,X
LDA   S
```

Constant Offset Indexed

In this mode, a two's-complement offset and the contents of one of the pointer registers are added to form the effective address of the operand. The pointer register's initial content is unchanged by the addition.

Three sizes of offsets are available:

5-bit (–16 to +15)
8-bit (–128 to +127)
16-bit (–32768 to +32767)

The two's complement 5-bit offset is included in the postbyte and, therefore, is most efficient in use of bytes and cycles. The two's complement 8-bit offset is contained in a single byte following the postbyte. The two's complement 16-bit offset is in the two bytes following the postbyte. In most cases the programmer need not be concerned with the size of this offset since the assembler will select the optimal size automatically.

Examples of constant-offset indexing are:

```
LDA   23,X
LDX   –2,S
LDY   300,X
LDU   CAT,Y
```

Accumulator-Offset Indexed

This mode is similar to constant offset indexed except that the two's-complement value in one of the accumulators (A, B or D) and the contents of one of the pointer registers are added to form the effective address of the operand. The contents of both the accumulator and the pointer register are unchanged by the addition. The postbyte specifies which accumulator to use as an offset and no additional bytes are required. The advantage of an accumulator offset is that the value of the offset can be calculated by a program at run-time.

Some examples are:

```
LDA   B,Y
LDX   D,Y
LEAX  B,X
```

Auto Increment/Decrement Indexed

In the auto increment addressing mode, the pointer register contains the address of the operand. Then, after the pointer register is used it is incremented by one or two. This addressing mode is useful in stepping through tables, moving data, or for the creation of software stacks. In auto decrement, the pointer register is decremented prior to use as the address of the data. The use of auto decrement is similar to that of auto increment; but the tables, etc., are scanned from the high to low addresses. The size of the increment/decrement can be either one or two to allow for tables of either 8- or 16-bit data to be accessed and is selectable by the programmer. The pre-

decrement, post-increment nature of these modes allow them to be used to create additional software stacks that behave identically to the U and S stacks.

Some examples of the auto increment/decrement addressing modes are:

```
LDA   ,X+
STD   ,Y++
LDB   ,-Y
LDX   ,--S
```

Care should be taken in performing operations on 16-bit pointer registers (X, Y, U, S) where the same register is used to calculate the effective address.

Consider the following instruction:

STX 0, X++ (X initialized to 0)

The desired result is to store a 0 in locations $0000 and $0001 then increment X to point to $0002. In reality, the following occurs:

0 → temp	calculate the EA; temp is a holding register
X + 2 → X	perform autoincrement
X → (temp)	do store operation

- **Indexed Indirect**

All of the indexing modes with the exception of auto increment/decrement by one, or a ±4-bit offset may have an additional level of indirection specified. In indirect addressing, the effective address is contained at the location specified by the contents of the Index Register plus any offset. In the example below, the A accumulator is loaded indirectly using an effective address calculated from the Index Register and an offset.

```
          Before Execution
          A = XX (don't care)
          X = $F000
$0100     LDA [$10, X]      EA is now $F010
$F010     $F1               $F150 is now the
$F011     $50               new EA
$F150     $AA
          After Execution
          A = $AA (Actual Data Loaded)
          X = $F000
```

All modes of indexed indirect are included except those which are meaningless (e.g., auto increment/decrement by 1 indirect). Some examples of indexed indirect are:

```
LDA   [,X]
LDD   [10,S]
LDA   [B,Y]
LDD   [,X++]
```

- **Relative Addressing**

The byte(s) following the branch opcode is (are) treated as a signed offset which may be added to the program counter. If the branch condition is true then the calculated address (PC + signed offset) is loaded into the program counter. Program execution continues at the new location as indicated by the PC; short (1 byte offset) and long (2 bytes offset) relative addressing modes are available. All of memory can be reached in long relative addressing as an effective address is interpreted modulo 2^{16}. Some examples of relative addressing are:

```
BEQ     CAT     (short)
BGT     DOG     (short)
CAT     LBEQ    RAT     (long)
DOG     LBGT    RABBIT  (long)
        .
        .
        .
RAT     NOP
RABBIT  NOP
```

- **Program Counter Relative**

The PC can be used as the pointer register with 8 or 16-bit signed offsets. As in relative addressing, the offset is added to the current PC to create the effective address. The effective address is then used as the address of the operand or data. Program Counter Relative Addressing is used for writing position independent programs. Tables related to a particular routine will maintain the same relationship after the routine is moved, if referenced relative to the Program Counter. Examples are:

```
LDA   CAT,PCR
LEAX  TABLE,PCR
```

Since program counter relative is a type of indexing, an additional level of indirection is available.

```
LDA   [CAT,PCR]
LDU   [DOG,PCR]
```

■ HD6809E INSTRUCTION SET

The instruction set of the HD6809E is similar to that of the HD6800 and is upward compatible at the source code level. The number of opcodes has been reduced from 72 to 59, but because of the expanded architecture and additional addressing modes, the number of available opcodes (with different addressing modes) has risen from 197 to 1464.

Some of the new instructions are described in detail below:

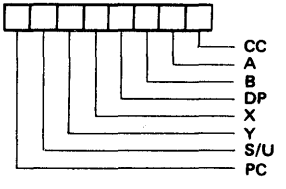
- **PSHU/PSHS**

The push instructions have the capability of pushing onto either the hardware stack (S) or user stack (U) any single register, or set of registers with a single instruction.

- **PULU/PULS**

The pull instructions have the same capability of the push instruction, in reverse order. The byte immediately following the push or pull opcode determines which register or registers are to be pushed or pulled. The actual PUSH/PULL sequence is fixed; each bit defines a unique register to push or pull, as shown in below.

PUSH/PULL POST BYTE




```
          ← Pull Order          Push Order →
PC        U   Y   X   DP  B   A   CC
FFFF ..... ← increasing memory address ..... 0000
PC        S   Y   X   DP  B   A   CC
```

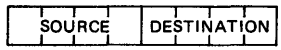
- **TFR/EXG**

Within the HD6809E, any register may be transferred to or exchanged with another of like-size; i.e., 8-bit to 8-bit or 16-bit to 16-bit. Bits 4~7 of postbyte define the source register, while bits 0~3 represent the destination register. These are denoted as follows:

0000 – D	0101 – PC
0001 – X	1000 – A
0010 – Y	1001 – B
0011 – U	1010 – CC
0100 – S	1011 – DP

(NOTE) All other combinations are undefined and INVALID.

TRANSFER/EXCHANGE POST BYTE



- **LEAX/LEAY/LEAU/LEAS**

The LEA (Load Effective Address) works by calculating the effective address used in an indexed instruction and stores that address value, rather than the data at that address, in a pointer register. This makes all the features of the internal addressing hardware available to the programmer. Some of the implications of this instruction are illustrated in Table 3.

The LEA instruction also allows the user to access data in a position independent manner. For example:

```
        LEAX    MSG1, PCR
        LBSR    PDATA (Print message routine)
        .
        .
MSG1    FCC     'MESSAGE'
```

This sample program prints: 'MESSAGE'. By writing MSG1, PCR, the assembler computes the distance between the present address and MSG1. This result is placed as a constant into the LEAX instruction which will be indexed from the PC value at the time of execution. No matter where the code is located, when it is executed, the computed offset from the PC will put the absolute address of MSG1 into the X pointer register. This code is totally position independent.

The LEA instructions are very powerful and use an internal holding register (temp). Care must be exercised when using the LEA instructions with the autoincrement and autodecrement addressing modes due to the sequence of internal operations. The LEA internal sequence is outlined as follows:

```
LEAa, b+        (any of the 16-bit pointer registers X, Y, U
                 or S may be substituted for a and b.)
1. b → temp     (calculate the EA)
2. b + 1 → b    (modify b, postincrement)
3. temp → a     (load a)

LEAa, – b
1. b – 1 → temp (calculate EA with predecrement)
2. b – 1 → b    (modify b, predecrement)
3. temp → a     (load a)
```

Autoincrement-by-two and autodecrement-by-two instructions work similarly. Note that LEAX, X+ does not change X, however LEAX, –X does decrement X. LEAX 1, X should be used to increment X by one.

Table 3 LEA Examples

Instruction	Operation	Comment
LEAX 10, X	X + 10 → X	Adds 5-bit constant 10 to X
LEAX 500, X	X + 500 → X	Adds 16-bit constant 500 to X
LEAY A, Y	Y + A → Y	Adds 8-bit A accumulator to Y
LEAY D, Y	Y + D → Y	Adds 16-bit D accumulator to Y
LEAU –10, U	U – 10 → U	Subtracts 10 from U
LEAS –10, S	S – 10 → S	Used to reserve area on stack
LEAS 10, S	S + 10 → S	Used to 'clean up' stack
LEAX 5, S	S + 5 → X	Transfers as well as adds

- **MUL**

Multiplies the unsigned binary numbers in the A and B accumulator and places the unsigned result into the 16-bit D accumulator. This unsigned multiply also allows multiple-precision multiplications.

Long and Short Relative Branches

The HD6809E has the capability of program counter relative branching throughout the entire memory map. In this mode, if the branch is to be taken, the 8 or 16-bit signed offset is added to the value of the program counter to be used as the effective address. This allows the program to branch anywhere in the 64k memory map. Position independent code can be easily generated through the use of relative branching. Both short (8-bit) and long (16-bit) branches are available.

- **SYNC**

After encountering a Sync instruction, the MPU enters a Sync state, stops processing instructions and waits for an interrupt. If the pending interrupt is non-maskable ($\overline{NMI}$) or maskable ($\overline{FIRQ}$, $\overline{IRQ}$) with its mask bit (F or I) clear, the processor will clear the Sync state and perform the normal interrupt stacking and service routine. Since $\overline{FIRQ}$ and $\overline{IRQ}$ are not edge-triggered, a low level with a minimum duration of three bus cycles is required to assure that the interrupt will be taken. If the pending interrupt is maskable ($\overline{FIRQ}$, $\overline{IRQ}$) with its mask bit (F or I) set, the processor will clear the Sync state and continue processing by executing the next inline instruction. Figure 17 depicts Sync timing.

Software Interrupts

A Software Interrupt is an instruction which will cause an interrupt, and its associated vector fetch. These Software Interrupts are useful in operating system calls, software debugging, trace operations, memory mapping, and software development systems. Three levels of SWI are available on this HD6809E, and are prioritized in the following order: SWI, SWI2, SWI3.

16-Bit Operation

The HD6809E has the capability of processing 16-bit data. These instructions include loads, stores, compares, adds, subtracts, transfers, exchanges, pushes and pulls.

■ CYCLE-BY-CYCLE OPERATION

The address bus cycle-by-cycle performance chart illustrates the memory-access sequence corresponding to each possible instruction and addressing mode in the HD6809E. Each instruction begins with an opcode fetch. While that opcode is being internally decoded, the next program byte is always fetched. (Most instructions will use the next byte, so this

technique considerably speeds throughput.) Next, the operation of each opcode will follow the flow chart. $\overline{VMA}$ is an indication of $FFFF_{16}$ on the address bus, R/$\overline{W}$ = "High" and BS = "Low". The following examples illustrate the use of the chart; see Figure 18.

Example 1: LBSR (Branch Taken)

Before Execution SP = F000

```
                    .
                    .
                    .
$8000               LBSR    CAT
                    .
                    .
                    .
$A000   CAT         .
```

CYCLE-BY-CYCLE FLOW

Cycle #	Address	Date	R/$\overline{W}$	Description
1	8000	17	1	Opcode Fetch
2	8001	20	1	Offset High Byte
3	8002	00	1	Offset Low Byte
4	FFFF	*	1	$\overline{VMA}$ Cycle
5	FFFF	*	1	$\overline{VMA}$ Cycle
6	FFFF	*	1	$\overline{VMA}$ Cycle
7	FFFF	*	1	$\overline{VMA}$ Cycle
8	EFFF	80	0	Stack High Order Byte of Return Address
9	EFFE	03	0	Stack Low Order Byte of Return Address

Example 2: DEC (Extended)

```
$8000    DEC    $A000
$A000    FCB    $80
```

CYCLE-BY-CYCLE FLOW

Cycle #	Address	Date	R/$\overline{W}$	Description
1	8000	7A	1	Opcode Fetch
2	8001	A0	1	Operand Address, High Byte
3	8002	00	1	Operand Address, Low Byte
4	FFFF	*	1	$\overline{VMA}$ Cycle
5	A000	80	1	Read the Data
6	FFFF	*	1	$\overline{VMA}$ Cycle
7	A000	7F	0	Store the Decremented Data

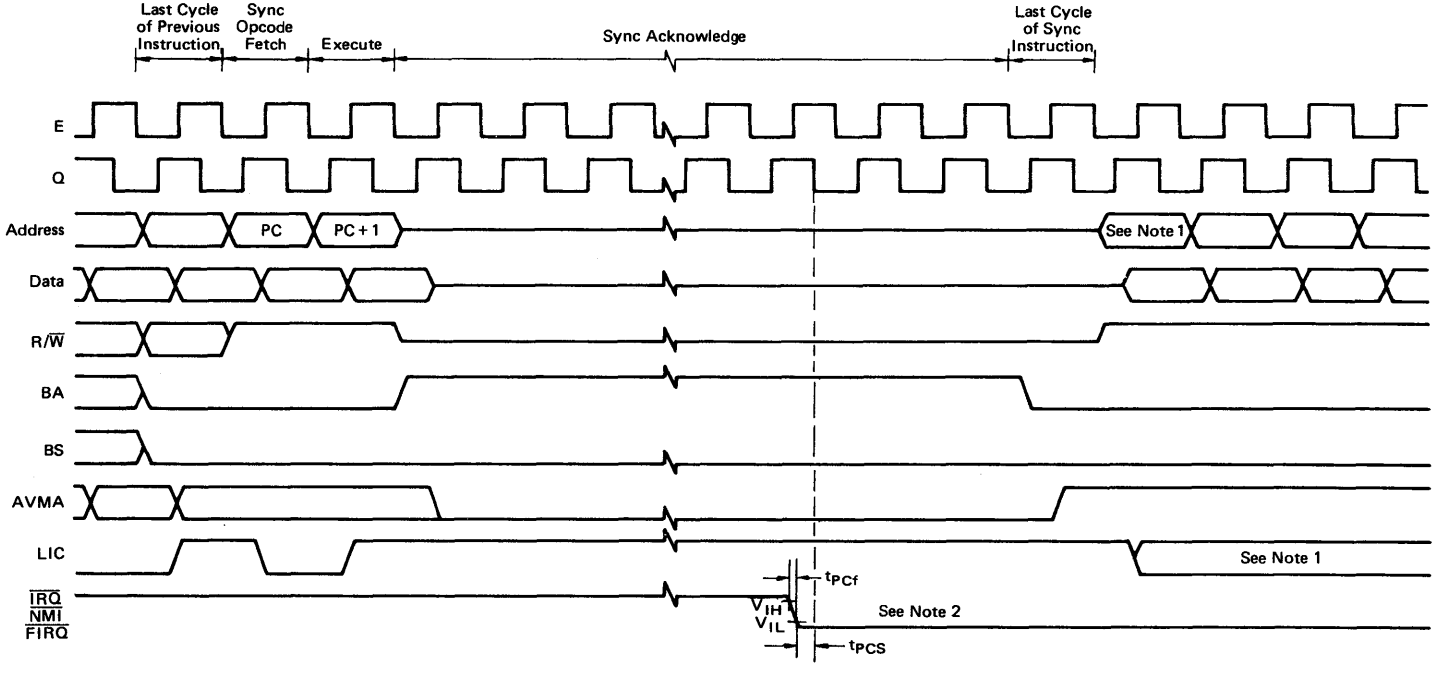
* The data bus has the data at that particular address.

■ HD6809E INSTRUCTION SET TABLES

The instructions of the HD6809E have been broken down into five different categories. They are as follows:

- 8-Bit operation (Table 4)
- 16-Bit operation (Table 5)
- Index register/stack pointer instructions (Table 6)
- Relative branches (long or short) (Table 7)
- Miscellaneous instructions (Table 8)

HD6809E instruction set tables and Hexadecimal Values of instructions are shown in Table 9 and Table 10.

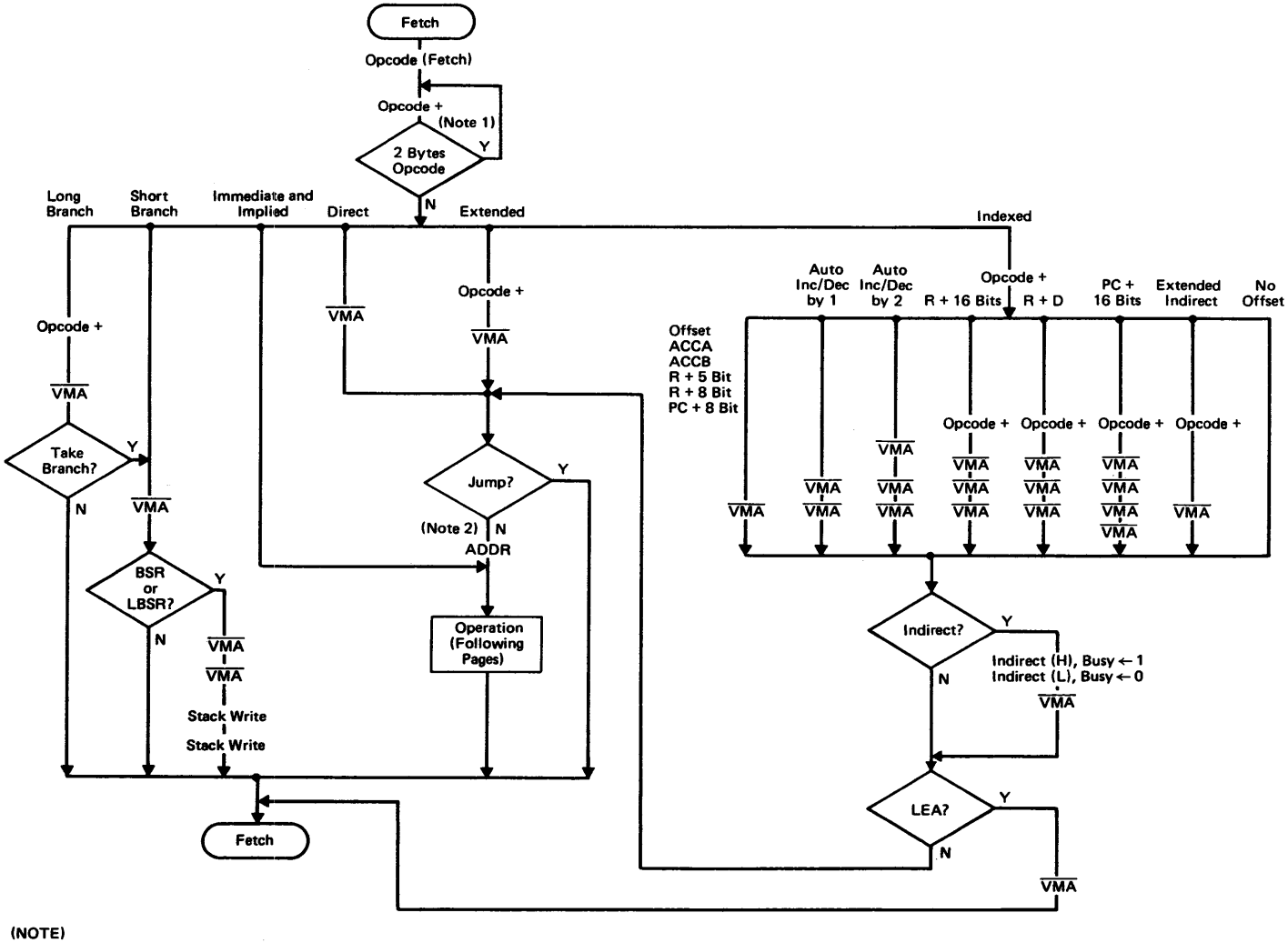


(NOTES) 1. If the associated mask bit is set when the interrupt is requested, LIC will go "Low" and this cycle will be an instruction fetch from address location PC + 1. However, if the interrupt is accepted ($\overline{NMI}$ or an unmasked $\overline{FIRQ}$ or $\overline{IRQ}$) LIC will remain "High" and interrupt processing will start with this cycle as (m) on Figure 9 and 10 (Interrupt Timing).

2. If mask bits are clear, $\overline{IRQ}$ and $\overline{FIRQ}$ must be held "Low" for three cycles to guarantee that interrupt will be taken, although only one cycle is necessary to bring the processor out of SYNC.

3. Waveform measurements for all inputs and outputs are specified at logic "High" = V_{IHmin} and logic "Low" = V_{ILmax} unless otherwise specified.

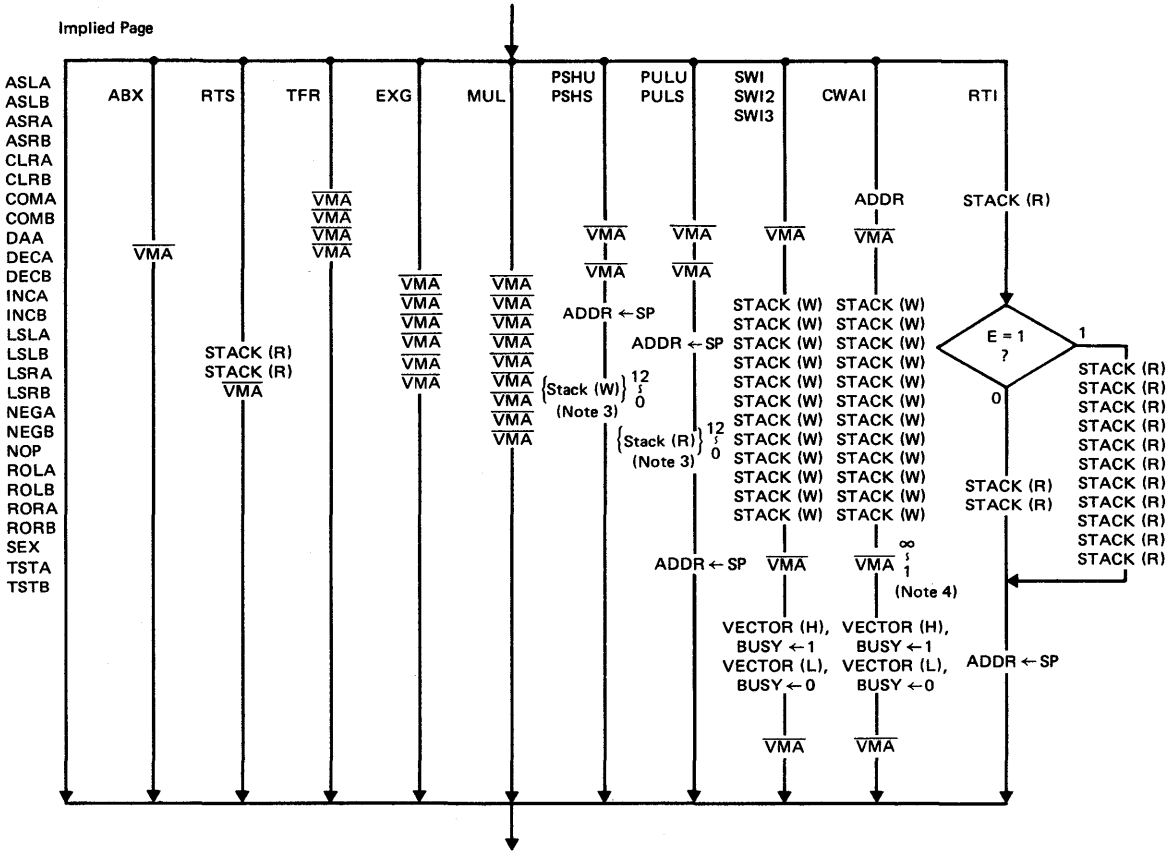
Figure 17 SYNC Timing



(NOTE)

1. Busy = "High" during access of first byte of double byte immediate load.
2. Write operation during store instruction. Busy = "High" during first two cycles of a double-byte access and the first cycle of read-modify-write access.
3. AVMA is asserted on the cycle before a $\overline{VMA}$ cycle.

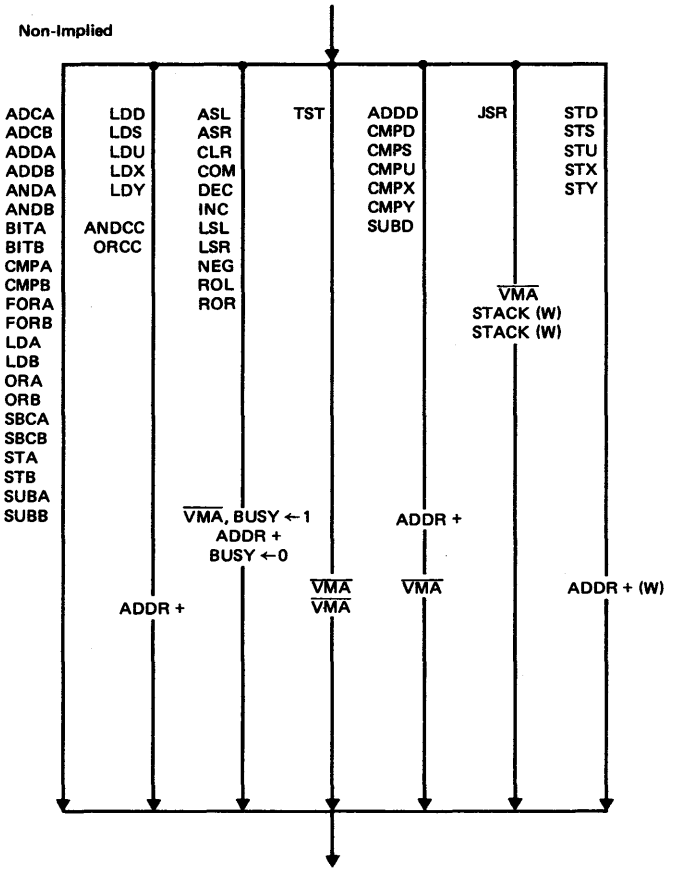
Figure 18 Address Bus Cycle-by-Cycle Performance



(NOTES)

1. Stack (W) refers to the following sequence: SP ← SP − 1, then ADDR ← SP with R/$\overline{W}$ = "Low"
   Stack (R) refers to the following sequence: ADDR ← SP with R/$\overline{W}$ = "High", then SP ← SP + 1.
   PSHU, PULU instructions use the user stack pointer (i.e., SP = U) and PSHS, PULS use the hardware stack pointer (i.e., SP = S).
2. Vector refers to the address of an interrupt or reset vector (see Table 1).
3. The number of stack accesses will vary according to the number of bytes saved.
4. $\overline{VMA}$ cycles will occur until an interrupt occurs.

Figure 18 Address Bus Cycle-by-Cycle Performance (Continued)



(NOTES)

1. Stack (W) refers to the following sequence: SP ← SP − 1, then ADDR ← SP with R/$\overline{W}$ = "Low"
   Stack (R) refers to the following sequence: ADDR ← SP with R/$\overline{W}$ = "High", then SP ← SP + 1.
   PSHU, PULU instructions use the user stack pointer (i.e., SP = U) and PSHS, PULS use the hardware stack pointer (i.e., SP = S).
2. Vector refers to the address of an interrupt or reset vector (see Table 1).
3. The number of stack accesses will vary according to the number of bytes saved.
4. $\overline{VMA}$ cycles will occur until an interrupt occurs.

Figure 18 Address Bus Cycle-by-Cycle Performance (Continued)

Table 4 8-Bit Accumulator and Memory Instructions

Mnemonic(s)	Operation
ADCA, ADCB	Add memory to accumulator with carry
ADDA, ADDB	Add memory to accumulator
ANDA, ANDB	And memory with accumulator
ASL, ASLA, ASLB	Arithmetic shift of accumulator or memory left
ASR, ASRA, ASRB	Arithmetic shift of accumulator or memory right
BITA, BITB	Bit test memory with accumulator
CLR, CLRA, CLRB	Clear accumulator or memory location
CMPA, CMPB	Compare memory from accumulator
COM, COMA, COMB	Complement accumultor or memory location
DAA	Decimal adjust A accumulator
DEC, DECA, DECB	Decrement accumulator or memory location
EORA, EORB	Exclusive or memory with accumulator
EXG R1, R2	Exchange R1 with R2 (R1, R2 = A, B, CC, DP)
INC, INCA, INCB	Increment accumulator or memory location
LDA, LDB	Load accumulator from memory
LSL, LSLA, LSLB	Logical shift left accumulator or memory location
LSR, LSRA, LSRB	Logical shift right accumulator or memory location
MUL	Unsigned multiply (A × B → D)
NEG, NEGA, NEGB	Negate accumulator or memory
ORA, ORB	Or memory with accumulator
ROL, ROLA, ROLB	Rotate accumulator or memory left
ROR, RORA, RORB	Rotate accumulator or memory right
SBCA, SBCB	Subtract memory from accumulator with borrow
STA, STB	Store accumulator to memory
SUBA, SUBB	Subtract memory from accumulator
TST, TSTA, TSTB	Test accumulator or memory location
TFR R1, R2	Transfer R1 to R2 (R1, R2 = A, B, CC, DP)

(NOTE) A, B, CC or DP may be pushed to (pulled from) either stack with PSHS, PSHU (PULS, PULU) instructions.

Table 5 16-Bit Accumulator and Memory Instructions

Mnemonic(s)	Operation
ADDD	Add memory to D accumulator
CMPD	Compare memory from D accumulator
EXG D, R	Exchange D with X, Y, S, U or PC
LDD	Load D accumulator from memory
SEX	Sign Extend B accumulator into A accumulator
STD	Store D accumulator to memory
SUBD	Subtract memory from D accumulator
TFR D, R	Transfer D to X, Y, S, U or PC
TFR R, D	Transfer X, Y, S, U or PC to D

(NOTE) D may be pushed (pulled) to either stack with PSHS, PSHU (PULS, PULU) instructions.

Table 6 Index Register Stack Pointer Instructions

Mnemonic(s)	Operation
CMPS, CMPU	Compare memory from stack pointer
CMPX, CMPY	Compare memory from index register
EXG R1, R2	Exchange D, X, Y, S, U or PC with D, X, Y, S, U or PC
LEAS, LEAU	Load effective address into stack pointer
LEAX, LEAY	Load effective address into index register
LDS, LDU	Load stack pointer from memory
LDX, LDY	Load index register from memory
PSHS	Push A, B, CC, DP, D, X, Y, U, or PC onto hardware stack
PSHU	Push A, B, CC, DP, D, X, Y, S, or PC onto user stack
PULS	Pull A, B, CC, DP, D, X, Y, U or PC from hardware stack
PULU	Pull A, B, CC, DP, D, X, Y, S or PC from user stack
STS, STU	Store stack pointer to memory
STX, STY	Store index register to memory
TFR R1, R2	Transfer D, X, Y, S, U or PC to D, X, Y, S, U or PC
ABX	Add B accumulator to X (unsigned)

Table 7 Branch Instructions

Mnemonic(s)	Operation
	SIMPLE BRANCHES
BEQ, LBEQ	Branch if equal
BNE, LBNE	Branch if not equal
BMI, LBMI	Branch if minus
BPL, LBPL	Branch if plus
BCS, LBCS	Branch if carry set
BCC, LBCC	Branch if carry clear
BVS, LBVS	Branch if overflow set
BVC, LBVC	Branch if overflow clear
	SIGNED BRANCHES
BGT, LBGT	Branch if greater (signed)
BGE, LBGE	Branch if greater than or equal (signed)
BEQ, LBEQ	Branch if equal
BLE, LBLE	Branch if less than or equal (signed)
BLT, LBLT	Branch if less than (signed)
	UNSIGNED BRANCHES
BHI, LBHI	Branch if higher (unsigned)
BHS, LBHS	Branch if higher or same (unsigned)
BEQ, LBEQ	Branch if equal
BLS, LBLS	Branch if lower or same (unsigned)
BLO, LBLO	Branch if lower (unsigned)
	OTHER BRANCHES
BSR, LBSR	Branch to subroutine
BRA, LBRA	Branch always
BRN, LBRN	Branch never

Table 8 Miscellaneous Instructions

Mnemonic(s)	Operation
ANDCC	AND condition code register
CWAI	AND condition code register, then wait for interrupt
NOP	No operation
ORCC	OR condition code register
JMP	Jump
JSR	Jump to subroutine
RTI	Return from interrupt
RTS	Return from subroutine
SWI, SWI2, SWI3	Software interrupt (absolute indirect)
SYNC	Synchronize with interrupt line

Table 9 HD6809E Instruction Set Table

INSTRUCTION/ FORMS		HD6809E ADDRESSING MODES																DESCRIPTION	5	3	2	1	0
		IMPLIED			DIRECT			EXTENDED			IMMEDIATE			INDEXED①			RELATIVE				H	N	Z	V	C
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~⑤	#						
ABX		3A	3	1																B + X → X (UNSIGNED)	•	•	•	•	•
ADC	ADCA				99	4	2	B9	5	3	89	2	2	A9	4+	2+				A + M + C → A	↕	↕	↕	↕	↕
	ADCB				D9	4	2	F9	5	3	C9	2	2	E9	4+	2+				B + M + C → B	↕	↕	↕	↕	↕
ADD	ADDA				9B	4	2	BB	5	3	8B	2	2	AB	4+	2+				A + M → A	↕	↕	↕	↕	↕
	ADDB				DB	4	2	FB	5	3	CB	2	2	EB	4+	2+				B + M → B	↕	↕	↕	↕	↕
	ADDD				D3	6	2	F3	7	3	C3	4	3	E3	6+	2+				D + M:M + 1 → D	•	↕	↕	↕	↕
AND	ANDA				94	4	2	B4	5	3	84	2	2	A4	4+	2+				A ∧ M → A	•	↕	↕	0	•
	ANDB				D4	4	2	F4	5	3	C4	2	2	E4	4+	2+				B ∧ M → B	•	↕	↕	0	•
	ANDCC										1C	3	2							CC ∧ IMM → CC	(	—	⑦	—	)
ASL	ASLA	48	2	1																A	⑧	↕	↕	↕	↕
	ASLB	58	2	1																B	⑧	↕	↕	↕	↕
	ASL				08	6	2	78	7	3				68	6+	2+				M	⑧	↕	↕	↕	↕
ASR	ASRA	47	2	1																A	⑧	↕	↕	•	↕
	ASRB	57	2	1																B	⑧	↕	↕	•	↕
	ASR				07	6	2	77	7	3				67	6+	2+				M	⑧	↕	↕	•	↕
BCC	BCC																24	3	2	Branch C = 0	•	•	•	•	•
	LBCC																10 24	5(6)	4	Long Branch C = 0	•	•	•	•	•
BCS	BCS																25	3	2	Branch C = 1	•	•	•	•	•
	LBCS																10 25	5(6)	4	Long Branch C = 1	•	•	•	•	•
BEQ	BEQ																27	3	2	Branch Z = 1	•	•	•	•	•
	LBEQ																10 27	5(6)	4	Long Branch Z = 1	•	•	•	•	•
BGE	BGE																2C	3	2	Branch N ⊕ V=0	•	•	•	•	•
	LBGE																10 2C	5(6)	4	Long Branch N ⊕ V=0	•	•	•	•	•
BGT	BGT																2E	3	2	Branch ZV(N⊕V)=0	•	•	•	•	•
	LBGT																10 2E	5(6)	4	Long Branch ZV(N ⊕ V)=0	•	•	•	•	•
BHI	BHI																22	3	2	Branch CVZ=0	•	•	•	•	•
	LBHI																10 22	5(6)	4	Long Branch CVZ=0	•	•	•	•	•
BHS	BHS																24	3	2	Branch C=0	•	•	•	•	•
	LBHS																10 24	5(6)	4	Long Branch C=0	•	•	•	•	•
BIT	BITA				95	4	2	B5	5	3	85	2	2	A5	4+	2+				Bit Test A (M ∧ A)	•	↕	↕	0	•
	BITB				D5	4	2	F5	5	3	C5	2	2	E5	4+	2+				Bit Test B (M ∧ B)	•	↕	↕	0	•
BLE	BLE																2F	3	2	Branch ZV(N⊕V)=1	•	•	•	•	•
	LBLE																10 2F	5(6)	4	Long Branch ZV(N ⊕ V)=1	•	•	•	•	•
BLO	BLO																25	3	2	Branch C=1	•	•	•	•	•
	LBLO																10 25	5(6)	4	Long Branch C=1	•	•	•	•	•
BLS	BLS																23	3	2	Branch CVZ=1	•	•	•	•	•
	LBLS																10 23	5(6)	4	Long Branch CVZ=1	•	•	•	•	•
BLT	BLT																2D	3	2	Branch N ⊕ V=1	•	•	•	•	•
	LBLT																10 2D	5(6)	4	Long Branch N ⊕ V=1	•	•	•	•	•
BMI	BMI																2B	3	2	Branch N=1	•	•	•	•	•
	LBMI																10 2B	5(6)	4	Long Branch N=1	•	•	•	•	•
BNE	BNE																26	3	2	Branch Z = 0	•	•	•	•	•
	LBNE																10 26	5(6)	4	Long Branch Z = 0	•	•	•	•	•
BPL	BPL																2A	3	2	Branch N = 0	•	•	•	•	•
	LBPL																10 2A	5(6)	4	Long Branch N = 0	•	•	•	•	•
BRA	BRA																20	3	2	Branch Always	•	•	•	•	•
	LBRA																16	5	3	Long Branch/ Always	•	•	•	•	•
BRN	BRN																21	3	2	Branch Never	•	•	•	•	•
	LBRN																10 21	5	4	Long Branch Never	•	•	•	•	•

(to be continued)

INSTRUCTION/ FORMS		HD6809E ADDRESSING MODES																	DESCRIPTION	5	3	2	1	0
		IMPLIED			DIRECT			EXTENDED			IMMEDIATE			INDEXED①			RELATIVE				H	N	Z	V	C
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~⑤	#						
BSR	BSR																8D	7	2	Branch to Subroutine	•	•	•	•	•
	LBSR																17	9	3	Long Branch to Subroutine	•	•	•	•	•
BVC	BVC																28	3	2	Branch V = 0	•	•	•	•	•
	LBVC																10 28	5(6)	4	Long Branch V = 0	•	•	•	•	•
BVS	BVS																29	3	2	Branch V = 1	•	•	•	•	•
	LBVS																10 29	5(6)	4	Long Branch V = 1	•	•	•	•	•
CLR	CLRA	4F	2	1																0 → A	•	0	1	0	0
	CLRB	5F	2	1																0 → B	•	0	1	0	0
	CLR				0F	6	2	7F	7	3				6F	6+	2+				0 → M	•	0	1	0	0
CMP	CMPA				91	4	2	B1	5	3	81	2	2	A1	4+	2+				Compare M from A	⑧	↕	↕	↕	↕
	CMPB				D1	4	2	F1	5	3	C1	2	2	E1	4+	2+				Compare M from B	⑧	↕	↕	↕	↕
	CMPD				10 93	7	3	10 B3	8	4	10 83	5	4	10 A3	7+	3+				Compare M: M + 1 from D	•	↕	↕	↕	↕
	CMPS				11 9C	7	3	11 BC	8	4	11 8C	5	4	11 AC	7+	3+				Compare M: M + 1 from S	•	↕	↕	↕	↕
	CMPU				11 93	7	3	11 B3	8	4	11 83	5	4	11 A3	7+	3+				Compare M: M + 1 from U	•	↕	↕	↕	↕
	CMPX				9C	6	2	BC	7	3	8C	4	3	AC	6+	2+				Compare M: M + 1 from X	•	↕	↕	↕	↕
	CMPY				10 9C	7	3	10 BC	8	4	10 8C	5	4	10 AC	7+	3+				Compare M: M + 1 from Y	•	↕	↕	↕	↕
COM	COMA	43	2	1																$\overline{A}$ → A	•	↕	↕	0	1
	COMB	53	2	1																$\overline{B}$ → B	•	↕	↕	0	1
	COM				03	6	2	73	7	3				63	6+	2+				$\overline{M}$ → M	•	↕	↕	0	1
CWAI		3C	20	2																CC ∧ IMM → CC (except 1→E) Wait for Interrupt	(			⑦	)
DAA		19	2	1																Decimal Adjust A	•	↕	↕	⑧	↕
DEC	DECA	4A	2	1																A − 1 → A	•	↕	↕	↕	•
	DECB	5A	2	1																B − 1 → B	•	↕	↕	↕	•
	DEC				0A	6	2	7A	7	3				6A	6+	2+				M − 1 → M	•	↕	↕	↕	•
EOR	EORA				98	4	2	B8	5	3	88	2	2	A8	4+	2+				A ⊕ M → A	•	↕	↕	0	•
	EORB				D8	4	2	F8	5	3	C8	2	2	E8	4+	2+				B ⊕ M → B	•	↕	↕	0	•
EXG	R1, R2	1E	7	2																R1 ↔ R2②	(			⑩	)
INC	INCA	4C	2	1																A + 1 → A	•	↕	↕	↕	•
	INCB	5C	2	1																B + 1 → B	•	↕	↕	↕	•
	INC				0C	6	2	7C	7	3				6C	6+	2+				M + 1 → M	•	↕	↕	↕	•
JMP					0E	3	2	7E	4	3				6E	3+	2+				EA③ → PC	•	•	•	•	•
JSR					9D	7	2	BD	8	3				AD	7+	2+				Jump to Subroutine	•	•	•	•	•
LD	LDA				96	4	2	B6	5	3	86	2	2	A6	4+	2+				M → A	•	↕	↕	0	•
	LDB				D6	4	2	F6	5	3	C6	2	2	E6	4+	2+				M → B	•	↕	↕	0	•
	LDD				DC	5	2	FC	6	3	CC	3	3	EC	5+	2+				M: M + 1 → D	•	↕	↕	0	•
	LDS				10 DE	6	3	10 FE	7	4	10 CE	4	4	10 EE	6+	3+				M: M + 1 → S	•	↕	↕	0	•
	LDU				DE	5	2	FE	6	3	CE	3	3	EE	5+	2+				M: M + 1 → U	•	↕	↕	0	•
	LDX				9E	5	2	BE	6	3	8E	3	3	AE	5+	2+				M: M + 1 → X	•	↕	↕	0	•
	LDY				10 9E	6	3	10 BE	7	4	10 8E	4	4	10 AE	6+	3+				M: M + 1 → Y	•	↕	↕	0	•
LEA	LEAS													32	4+	2+				EA③ → S	•	•	•	•	•
	LEAU													33	4+	2+				EA③ → U	•	•	•	•	•
	LEAX													30	4+	2+				EA③ → X	•	•	↕	•	•
	LEAY													31	4+	2+				EA③ → Y	•	•	↕	•	•
LSL	LSLA	48	2	1																A	•	↕	↕	↕	↕
	LSLB	58	2	1																B	•	↕	↕	↕	↕
	LSL				08	6	2	78	7	3				68	6+	2+				M	•	↕	↕	↕	↕
LSR	LSRA	44	2	1																A	•	0	↕	•	↕
	LSRB	54	2	1																B	•	0	↕	•	↕
	LSR				04	6	2	74	7	3				64	6+	2+				M	•	0	↕	•	↕
MUL		3D	11	1																A × B → D (Unsigned)	•	•	↕	•	⑨
NEG	NEGA	40	2	1																$\overline{A}$ + 1 → A	⑧	↕	↕	↕	↕
	NEGB	50	2	1																$\overline{B}$ + 1 → B	⑧	↕	↕	↕	↕
	NEG				00	6	2	70	7	3				60	6+	2+				$\overline{M}$ + 1 → M	⑧	↕	↕	↕	↕
NOP		12	2	1																No Operation	•	•	•	•	•

(to be continued)

INSTRUCTION/ FORMS		HD6809E ADDRESSING MODES																DESCRIPTION	5	3	2	1	0
		IMPLIED			DIRECT			EXTENDED			IMMEDIATE			INDEXED①			RELATIVE				H	N	Z	V	C
		OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~⑤	#						
OR	ORA				9A	4	2	BA	5	3	8A	2	2	AA	4+	2+				A ∨ M → A	•	↕	↕	0	•
	ORB				DA	4	2	FA	5	3	CA	2	2	EA	4+	2+				B ∨ M → B	•	↕	↕	0	•
	ORCC										1A	3	2							CC ∨ IMM → CC	(		⑦		)
PSH	PSHS	34	5+④	2																Push Registers on S Stack	•	•	•	•	•
	PSHU	36	5+④	2																Push Registers on U Stack	•	•	•	•	•
PUL	PULS	35	5+④	2																Pull Registers from S Stack	(		⑩		)
	PULU	37	5+④	2																Pull Registers from U Stack	(		⑩		)
ROL	ROLA	49	2	1																A	•	↕	↕	↕	↕
	ROLB	59	2	1																B	•	↕	↕	↕	↕
	ROL				09	6	2	79	7	3				69	6+	2+				M	•	↕	↕	↕	↕
ROR	RORA	46	2	1																A	•	↕	↕	•	↕
	RORB	56	2	1																B	•	↕	↕	•	↕
	ROR				06	6	2	76	7	3				66	6+	2+				M	•	↕	↕	•	↕
RTI		3B	6/15	1																Return From Interrupt	(		⑦		)
RTS		39	5	1																Return From Subroutine	•	•	•	•	•
SBC	SBCA				92	4	2	B2	5	3	82	2	2	A2	4+	2+				A − M − C → A	⑧	↕	↕	↕	↕
	SBCB				D2	4	2	F2	5	3	C2	2	2	E2	4+	2+				B − M − C → B	⑧	↕	↕	↕	↕
SEX		1D	2	1																Sign Extend B into A	•	↕	↕	•	•
ST	STA				97	4	2	B7	5	3				A7	4+	2+				A → M	•	↕	↕	0	•
	STB				D7	4	2	F7	5	3				E7	4+	2+				B → M	•	↕	↕	0	•
	STD				DD	5	2	FD	6	3				ED	5+	2+				D → M: M+ 1	•	↕	↕	0	•
	STS				10 DF	6	3	10 FF	7	4				10 EF	6+	3+				S → M: M + 1	•	↕	↕	0	•
	STU				DF	5	2	FF	6	3				EF	5+	2+				U → M: M + 1	•	↕	↕	0	•
	STX				9F	5	2	BF	6	3				AF	5+	2+				X → M: M + 1	•	↕	↕	0	•
	STY				10 9F	6	3	10 BF	7	4				10 AF	6+	3+				Y → M: M + 1	•	↕	↕	0	•
SUB	SUBA				90	4	2	B0	5	3	80	2	2	A0	4+	2+				A − M → A	⑧	↕	↕	↕	↕
	SUBB				D0	4	2	F0	5	3	C0	2	2	E0	4+	2+				B − M → B	⑧	↕	↕	↕	↕
	SUBD				93	6	2	B3	7	3	83	4	3	A3	6+	2+				D − M: M + 1 → D	•	↕	↕	↕	↕
SWI	SWI⑥	3F	19	1																Software Interrupt1	•	•	•	•	•
	SWI2⑥	10 3F	20	2																Software Interrupt2	•	•	•	•	•
	SWI3⑥	11 3F	20	2																Software Interrupt3	•	•	•	•	•
SYNC		13	≥2	1																Synchronize to Interrupt	•	•	•	•	•
TFR	R1, R2	1F	6	2																R1 → R2②	(		⑩		)
TST	TSTA	4D	2	1																Test A	•	↕	↕	0	•
	TSTB	5D	2	1																Test B	•	↕	↕	0	•
	TST				0D	6	2	7D	7	3				6D	6+	2+				Test M	•	↕	↕	0	•

(NOTES)

① This column gives a base cycle and byte count. To obtain total count, and the values obtained from the INDEXED ADDRESSING MODES table.
② R1 and R2 may be any pair of 8 bit or any pair of 16 bit registers.
The 8 bit registers are: A, B, CC, DP
The 16 bit registers are: X, Y, U, S, D, PC
③ EA is the effective address.
④ The PSH and PUL instructions require 5 cycle plus 1 cycle for each byte pushed or pulled.
⑤ 5(6) means: 5 cycles if branch not taken, 6 cycles if taken.
⑥ SWI sets I and F bits. SWI2 and SWI3 do not affect I and F.
⑦ Conditions Codes set as a direct result of the instruction.
⑧ Value of half-carry flag is undefined.
⑨ Special Case – Carry set if b7 is SET.
⑩ Condition Codes set as a direct result of the instruction if CC is specified, and not affected otherwise.

LEGEND:

OP	Operation Code (Hexadecimal)	Z	Zero (byte)
~	Number of MPU Cycles	V	Overflow, 2's complement
#	Number of Program Bytes	C	Carry from bit 7
+	Arithmetic Plus	↕	Test and set if true, cleared otherwise
−	Arithmetic Minus	•	Not Affected
×	Multiply	CC	Condition Code Register
M̄	Complement of M	:	Concatenation
→	Transfer Into	∨	Logical or
H	Half-carry (from bit 3)	∧	Logical and
N	Negative (sign bit)	⊕	Logical Exclusive or

Table 10 Hexadecimal Values of Machine Codes

OP	Mnem	Mode	~	#	OP	Mnem	Mode	~	#	OP	Mnem	Mode	~	#
00	NEG	Direct	6	2	30	LEAX	Indexed	4+	2+	60	NEG	Indexed	6+	2+
01	*	↑			31	LEAY	↑	4+	2+	61	*	↑		
02	*	|			32	LEAS	↓	4+	2+	62	*	|		
03	COM	|	6	2	33	LEAU	Indexed	4+	2+	63	COM	|	6+	2+
04	LSR	|	6	2	34	PSHS	Implied	5+	2	64	LSR	|	6+	2+
05	*	|			35	PULS	↑	5+	2	65	*	|		
06	ROR	|	6	2	36	PSHU	|	5+	2	66	ROR	|	6+	2+
07	ASR	|	6	2	37	PULU	|	5+	2	67	ASR	|	6+	2+
08	ASL, LSL	|	6	2	38	*	|			68	ASL, LSL	|	6+	2+
09	ROL	|	6	2	39	RTS	|	5	1	69	ROL	|	6+	2+
0A	DEC	|	6	2	3A	ABX	|	3	1	6A	DEC	|	6+	2+
0B	*	|			3B	RTI	|	6, 15	1	6B	*	|		
0C	INC	|	6	2	3C	CWAI	|	20	2	6C	INC	|	6+	2+
0D	TST	|	6	2	3D	MUL	|	11	1	6D	TST	|	6+	2+
0E	JMP	|	3	2	3E	*	↓			6E	JMP	↓	3+	2+
0F	CLR	Direct	6	2	3F	SWI	Implied	19	1	6F	CLR	Indexed	6+	2+
10	See	—	—	—	40	NEGA	Implied	2	1	70	NEG	Extended	7	3
11	Next Page	—	—	—	41	*	↑			71	*	↑		
12	NOP	Implied	2	1	42	*	|			72	*	|		
13	SYNC	Implied	2	1	43	COMA	|	2	1	73	COM	|	7	3
14	*	|			44	LSRA	|	2	1	74	LSR	|	7	3
15	*	|			45	*	|			75	*	|		
16	LBRA	Relative	5	3	46	RORA	|	2	1	76	ROR	|	7	3
17	LBSR	Relative	9	3	47	ASRA	|	2	1	77	ASR	|	7	3
18	*	|			48	ASLA, LSLA	|	2	1	78	ASL, LSL	|	7	3
19	DAA	Implied	2	1	49	ROLA	|	2	1	79	ROL	|	7	3
1A	ORCC	Immed	3	2	4A	DECA	|	2	1	7A	DEC	|	7	3
1B	*	|			4B	*	|			7B	*	|		
1C	ANDCC	Immed	3	2	4C	INCA	|	2	1	7C	INC	|	7	3
1D	SEX	Implied	2	1	4D	TSTA	|	2	1	7D	TST	|	7	3
1E	EXG	↕	8	2	4E	*	|			7E	JMP	|	4	3
1F	TFR	Implied	6	2	4F	CLRA	Implied	2	1	7F	CLR	Extended	7	3
20	BRA	Relative	3	2	50	NEGB	Implied	2	1	80	SUBA	Immed	2	2
21	BRN	↑	3	2	51	*	↑			81	CMPA	↑	2	2
22	BHI	|	3	2	52	*	|			82	SBCA	|	2	2
23	BLS	|	3	2	53	COMB	|	2	1	83	SUBD	|	4	3
24	BHS, BCC	|	3	2	54	LSRB	|	2	1	84	ANDA	|	2	2
25	BLO, BCS	|	3	2	55	*	|			85	BITA	|	2	2
26	BNE	|	3	2	56	RORB	|	2	1	86	LDA	|	2	2
27	BEQ	|	3	2	57	ASRA	|	2	1	87	*	|		
28	BVC	|	3	2	58	ASLB, LSLB	|	2	1	88	EORA	|	2	2
29	BVS	|	3	2	59	ROLB	|	2	1	89	ADCA	|	2	2
2A	BPL	|	3	2	5A	DECB	|	2	1	8A	ORA	|	2	2
2B	BMI	|	3	2	5B	*	|			8B	ADDA	↓	2	2
2C	BGE	|	3	2	5C	INCB	|	2	1	8C	CMPX	Immed	4	3
2D	BLT	|	3	2	5D	TSTB	|	2	1	8D	BSR	Relative	7	2
2E	BGT	|	3	2	5E	*	↓			8E	LDX	Immed	3	3
2F	BLE	Relative	3	2	5F	CLRB	Implied	2	1	8F	*			

LEGEND:

~ Number of MPU cycles (less possible push pull or indexed-mode cycles)
# Number of program bytes
* Denotes unused opcode

(to be continued)

OP	Mnem	Mode	~	#
90	SUBA	Direct	4	2
91	CMPA	↑	4	2
92	SBCA		4	2
93	SUBD		6	2
94	ANDA		4	2
95	BITA		4	2
96	LDA		4	2
97	STA		4	2
98	EORA		4	2
99	ADCA		4	2
9A	ORA		4	2
9B	ADDA		4	2
9C	CMPX		6	2
9D	JSR		7	2
9E	LDX		5	2
9F	STX	Direct	5	2
A0	SUBA	Indexed	4+	2+
A1	CMPA	↑	4+	2+
A2	SBCA		4+	2+
A3	SUBD		6+	2+
A4	ANDA		4+	2+
A5	BITA		4+	2+
A6	LDA		4+	2+
A7	STA		4+	2+
A8	EORA		4+	2+
A9	ADCA		4+	2+
AA	ORA		4+	2+
AB	ADDA		4+	2+
AC	CMPX		6+	2+
AD	JSR		7+	2+
AE	LDX		5+	2+
AF	STX	Indexed	5+	2+
B0	SUBA	Extended	5	3
B1	CMPA	↑	5	3
B2	SBCA		5	3
B3	SUBD		7	3
B4	ANDA		5	3
B5	BITA		5	3
B6	LDA		5	3
B7	STA		5	3
B8	EORA		5	3
B9	ADCA		5	3
BA	ORA		5	3
BB	ADDA		5	3
BC	CMPX		7	3
BD	JSR		8	3
BE	LDX		6	3
BF	STX	Extended	6	3
C0	SUBB	Immed	2	2
C1	CMPB	↑	2	2
C2	SBCB		2	2
C3	ADDD		4	3
C4	ANDB		2	2
C5	BITB	Immed	2	2

OP	Mnem	Mode	~	#
C6	LDB	Immed	2	2
C7	*	↑		
C8	EORB		2	2
C9	ADCB		2	2
CA	ORB		2	2
CB	ADDB		2	2
CC	LDD		3	3
CD	*			
CE	LDU	Immed	3	3
CF	*			
D0	SUBB	Direct	4	2
D1	CMPB	↑	4	2
D2	SBCB		4	2
D3	ADDD		6	2
D4	ANDB		4	2
D5	BITB		4	2
D6	LDB		4	2
D7	STB		4	2
D8	EORB		4	2
D9	ADCB		4	2
DA	ORB		4	2
DB	ADDB		4	2
DC	LDD		5	2
DD	STD		5	2
DE	LDU		5	2
DF	STU	Direct	5	2
E0	SUBB	Indexed	4+	2+
E1	CMPB	↑	4+	2+
E2	SBCB		4+	2+
E3	ADDD		6+	2+
E4	ANDB		4+	2+
E5	BITB		4+	2+
E6	LDB		4+	2+
E7	STB		4+	2+
E8	EORB		4+	2+
E9	ADCB		4+	2+
EA	ORB		4+	2+
EB	ADDB		4+	2+
EC	LDD		5+	2+
ED	STD		5+	2+
EE	LDU		5+	2+
EF	STU	Indexed	5+	2+
F0	SUBB	Extended	5	3
F1	CMPB	↑	5	3
F2	SBCB		5	3
F3	ADDD		7	3
F4	ANDB		5	3
F5	BITB		5	3
F6	LDB		5	3
F7	STB		5	3
F8	EORB		5	3
F9	ADCB		5	3
FA	ORB		5	3
FB	ADDB	Extended	5	3

OP	Mnem	Mode	~	#
FC	LDD	Extended	6	3
FD	STD	↑	6	3
FE	LDU		6	3
FF	STU	Extended	6	3
		2 Bytes Opcode		
1021	LBRN	Relative	5	4
1022	LBHI	↑	5(6)	4
1023	LBLS		5(6)	4
1024	LBHS, LBCC		5(6)	4
1025	LBCS, LBLO		5(6)	4
1026	LBNE		5(6)	4
1027	LBEQ		5(6)	4
1028	LBVC		5(6)	4
1029	LBVS		5(6)	4
102A	LBPL		5(6)	4
102B	LBMI		5(6)	4
102C	LBGE		5(6)	4
102D	LBLT		5(6)	4
102E	LBGT		5(6)	4
102F	LBLE	Relative	5(6)	4
103F	SWI2	Implied	20	2
1083	CMPD	Immed	5	4
108C	CMPY	↑	5	4
108E	LDY	Immed	4	4
1093	CMPD	Direct	7	3
109C	CMPY	↑	7	3
109E	LDY		6	3
109F	STY	Direct	6	3
10A3	CMPD	Indexed	7+	3+
10AC	CMPY	↑	7+	3+
10AE	LDY		6+	3+
10AF	STY	Indexed	6+	3+
10B3	CMPD	Extended	8	4
10BC	CMPY	↑	8	4
10BE	LDY		7	4
10BF	STY	Extended	7	4
10CE	LDS	Immed	4	4
10DE	LDS	Direct	6	3
10DF	STS	Direct	6	3
10EE	LDS	Indexed	6+	3+
10EF	STS	Indexed	6+	3+
10FE	LDS	Extended	7	4
10FF	STS	Extended	7	4
113F	SWI3	Implied	20	2
1183	CMPU	Immed	5	4
118C	CMPS	Immed	5	4
1193	CMPU	Direct	7	3
119C	CMPS	Direct	7	3
11A3	CMPU	Indexed	7+	3+
11AC	CMPS	Indexed	7+	3+
11B3	CMPU	Extended	8	4
11BC	CMPS	Extended	8	4

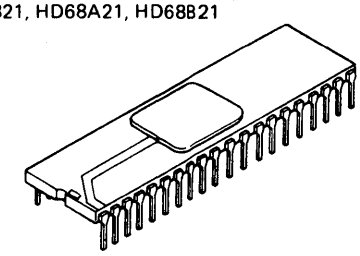
(NOTE): All unused opcodes are both undefined and illegal.

HD6821, HD68A21, HD68B21—
PIA (Peripheral Interface Adapter)

The HD6821 Peripheral Interface Adapter provides the universal means of interfacing peripheral equipment to the HD6800 Microprocessing Unit (MPU). This device is capable of interfacing the MPU to peripherals through two 8-bit bi-directional peripheral data buses and four control lines. No external logic is required for interfacing to most peripheral devices.

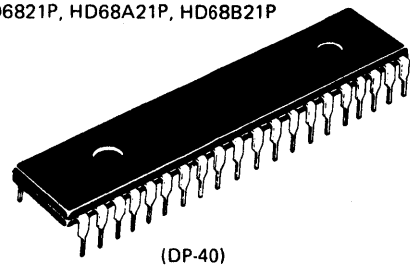
The functional configuration of the PIA is programmed by the MPU during system initialization. Each of the peripheral data lines can be programmed to act as an input or output, and each of the four control/interrupt lines may be programmed for one of several control modes. This allows a high degree of flexibility in the over-all operation of the interface.

HD6821, HD68A21, HD68B21



(DC-40)

HD6821P, HD68A21P, HD68B21P

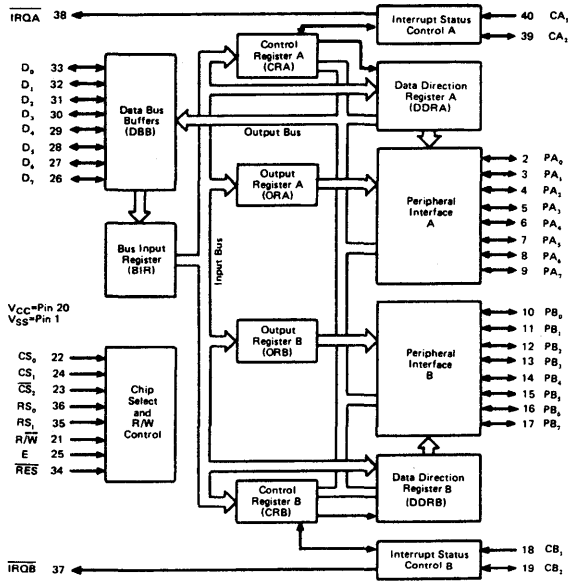


(DP-40)

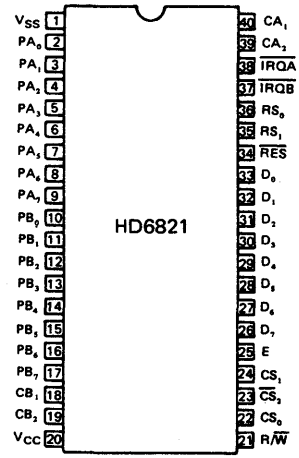
■ FEATURES

- Two Bi-directional 8-Bit Peripheral Data Bus for interface to Peripheral devices
- Two Programmable Control Registers
- Two Programmable Data Direction Registers
- Four Individually-Controlled Interrupt Input Lines: Two Usable as Peripheral Control Outputs
- Handshake Control Logic for Input and Output Peripheral Operation
- High-Impedance 3-State and Direct Transistor Drive Peripheral Lines
- Program Controlled Interrupt and Interrupt Disable Capability
- CMOS Drive Capability on Side A Peripheral Lines
- Two TTL Drive Capability on All A and B Side Buffers
- N Channel Silicon Gate MOS
- Compatible with MC6821, MC68A21 and MC68B21

■ BLOCK DIATRAM



■ PIN ARRANGEMENT



(Top View)

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC}*	-0.3 ~ +7.0	V
Input Voltage	V_{in}*	-0.3 ~ +7.0	V
Operating Temperature	T_{opr}	-20 ~ +75	°C
Storage Temperature	T_{stg}	-55 ~ +150	°C

* With respect to V_{SS} (SYSTEM GND)

(NOTE) Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ RECOMMENDED OPERATING CONDITIONS

Item	Symbol	min	typ	max	Unit
Supply Voltage	V_{CC}*	4.75	5.0	5.25	V
Input Voltage	V_{IL}*	-0.3	–	0.8	V
	V_{IH}*	2.0	–	V_{CC}	
Operating Temperature	T_{opr}	-20	25	75	°C

* With respect to V_{SS} (SYSTEM GND)

■ ELECTRICAL CHARACTERISTICS

• DC CHARACTERISTICS (V_{CC}=5.0V±5%, V_{SS}=0V, Ta=-20~+75°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ*	max	Unit
Input "High" Voltage	All Inputs	V_{IH}		2.0	–	V_{CC}	V
Input "Low" Voltage	All Inputs	V_{IL}		-0.3	–	0.8	V
Input Leakage Current	R/$\overline{W}$, $\overline{RES}$, $\overline{RS_0}$, RS_1, CS_0, CS_1, $\overline{CS_2}$, CA_1, CB_1, E	I_{in}	V_{in} = 0~5.25V	-2.5	–	2.5	μA
Three-State (Off State) Input Current	D_0~D_7, PB_0~PB_7, CB_2	I_{TSI}	V_{in} = 0.4~2.4V	-10	–	10	μA
Input "High" Current	PA_0~PA_7, CA_2	I_{IH}	V_{IH} = 2.4V	-200	–	–	μA
Input "Low" Current	PA_0~PA_7, CA_2	I_{IL}	V_{IL} = 0.4V	–	–	-2.4	mA
Output "High" Voltage	D_0~D_7	V_{OH}	I_{OH} = -205μA	2.4	–	–	V
	PA_0~PA_7, CA_2		I_{OH} = -200μA	2.4**	–	–	
			I_{OH} = -10μA	V_{CC}-1.0	–	–	
	PB_0~PB_7, CB_2		I_{OH} = -200μA	2.4	–	–	
Output "Low" Voltage	D_0~D_7, $\overline{IRQA}$, $\overline{IRQB}$	V_{OL}	I_{OL} = 1.6mA	–	–	0.4	V
	Other Outputs		I_{OL} = 1.6mA	–	–	0.4	
			I_{OL} = 3.2mA	–	–	0.6	
Output "High" Current	D_0~D_7	I_{OH}	V_{OH} = 2.4V	-205	–	–	μA
	PA_0~PA_7, CA_2		V_{OH} = 2.4V**	-200	–	–	μA
	PB_0~PB_7, CB_2		V_{OH} = 1.5V	-1.0	–	-10	mA
Output Leakage Current (Off State)	$\overline{IRQA}$, $\overline{IRQB}$	I_{LOH}	V_{OH} = 2.4V	–	–	10	μA
Power Dissipation		P_D		–	260	550	mW
Input Capacitance	PA_0~PA_7, PB_0~PB_7, CA_2, CB_2, D_0~D_7	C_{in}	V_{in} = 0V, Ta = 25°C, f = 1.0MHz	–	–	12.5	pF
	R/$\overline{W}$, $\overline{RES}$, $\overline{RS_0}$, RS_1, CS_0, CS_1, $\overline{CS_2}$, CA_1, CB_1, E			–	–	10	
Output Capacitance	$\overline{IRQA}$, $\overline{IRQB}$	C_{out}	V_{in} = 0V, Ta = 25°C, f = 1.0MHz	–	–	10	pF

* Ta = 25°C, V_{CC} = 5.0V ** HD68B21; V_{OH} = 2.2V min (PA_0~PA_7, CA_2)

• AC CHARACTERISTICS (V_{CC}=5.0V±5%, V_{SS}=0, Ta=-20~+75°C, unless otherwise noted.)

1. PERIPHERAL TIMING

Item		Symbol	Test Condition	HD6821		HD68A21		HD68B21		Unit
				min	max	min	max	min	max	
Peripheral Data Setup Time		t_{PDSU}	Fig. 1	200	–	135	–	100	–	ns
Peripheral Data Hold Time		t_{PDH}	Fig. 1	0	–	0	–	0	–	ns
Delay Time, Enable negative transition to CA_2 negative transition	Enable → CA_2 Negative	t_{CA2}	Fig. 2, Fig. 3	–	1.0	–	0.67	–	0.5	μs
Delay Time, Enable negative transition to CA_2 positive transition	Enable → CA_2 Positive	t_{RS1}	Fig. 2	–	1.0	–	0.67	–	0.5	μs
Rise and Fall Times for CA_1 and CA_2 input signals	CA_1, CA_2	t_r, t_f	Fig. 3	–	1.0	–	1.0	–	1.0	μs
Delay Time from CA_1 active transition to CA_2 positive transition	CA_1 – CA_2	t_{RS2}	Fig. 3	–	2.0	–	1.35	–	1.0	μs
Delay Time, Enable negative transition to Peripheral Data Valid	Enable→Peripheral Data	t_{PDW}	Fig. 4, Fig. 5	–	1.0	–	0.67	–	0.5	μs
Delay Time, Enable negative transition to Peripheral CMOS Data Valid	Enable → Peripheral Data PA_0~PA_7, CA_2	t_{CMOS}	V_{CC} – 30% V_{CC} Fig. 4	–	2.0	–	1.35	–	1.0	μs
Delay Time, Enable positive transition to CB_2 negative position	Enable → CB_2	t_{CB2}	Fig. 6, Fig. 7	–	1.0	–	0.67	–	0.5	μs
Delay Time, Peripheral Data Valid to CB_2 negative transition	Peripheral Data → CB_2	t_{DC}	Fig. 5	20	–	20	–	20	–	ns
Delay Time, Enable positive transition to CB_2 positive transition	Enable → CB_2	t_{RS1}	Fig. 6	–	1.0	–	0.67	–	0.5	μs
Peripheral Control Output Pulse Width, CA_2/CB_2	CA_2, CB_2	PW_{CT}	Fig. 2, Fig. 6	550	–	550	–	500	–	ns
Rise and Fall Time for CB_1 and CB_2 input signals	CB_1, CB_2	t_r, t_f	Fig. 7	–	1.0	–	1.0	–	1.0	μs
Delay Time, CB_1 active transition to CB_2 positive transition	CB_1 → CB_2	t_{RS2}	Fig. 7	–	2.0	–	1.35	–	1.0	μs
Interrupt Release Time, $\overline{IRQA}$ and $\overline{IRQB}$	$\overline{IRQA}$, $\overline{IRQB}$	t_{IR}	Fig. 9	–	1.6	–	1.1	–	0.85	μs
Interrupt Response Time	$\overline{IRQA}$, $\overline{IRQB}$	t_{RS3}	Fig. 8	–	1.0	–	1.0	–	1.0	μs
Interrupt Input Pulse Width	CA_1, CA_2, CB_1, CB_2	PWI	Fig. 8	500**	–	500**	–	500**	–	ns
Reset "Low" Time	$\overline{RES}$*	t_{RL}	Fig. 10	1.0	–	0.66	–	0.5	–	μs

* The Reset line must be "High" a minimum of 1.0μs before addressing the PIA.
** At least one Enable "High" pulse should be included in this period.

2. BUS TIMING

1) READ

Item		Symbol	Test Condition	HD6821		HD68A21		HD68B21		Unit
				min	max	min	max	min	max	
Enable Cycle Time		t_{cycE}	Fig. 11	1000	–	666	–	500	–	ns
Enable Pulse Width, "High"		PW_{EH}	Fig. 11	450	–	280	–	220	–	ns
Enable Pulse Width, "Low"		PW_{EL}	Fig. 11	430	–	280	–	210	–	ns
Enable Pulse Rise and Fall Times		t_{Er}, t_{Ef}	Fig. 11	–	25	–	25	–	25	ns
Setup Time	Address, R/$\overline{W}$–Enable	t_{AS}	Fig. 12	140	–	140	–	70	–	ns
Address Hold Time		t_{AH}	Fig. 12	10	–	10	–	10	–	ns
Data Delay Time		t_{DDR}	Fig. 12	–	320	–	220	–	180	ns
Data Hold Time		t_{DHR}	Fig. 12	10	–	10	–	10	–	ns

2) WRITE

Item		Symbol	Test Condition	HD6821		HD68A21		HD68B21		Unit
				min	max	min	max	min	max	
Enable Cycle Time		t_{cycE}	Fig. 11	1000	–	666	–	500	–	ns
Enable Pulse Width, "High"		PW_{EH}	Fig. 11	450	–	280	–	220	–	ns
Enable Pulse Width, "Low"		PW_{EL}	Fig. 11	430	–	280	–	210	–	ns
Enable Pulse Rise and Fall Times		t_{Er}, t_{Ef}	Fig. 11	–	25	–	25	–	25	ns
Setup Time		t_{AS}	Fig. 13	140	–	140	–	70	–	ns
Address Hold Time	Address, R/$\overline{W}$–Enable	t_{AH}	Fig. 13	10	–	10	–	10	–	ns
Data Setup Time		t_{DSW}	Fig. 13	195	–	80	–	60	–	ns
Data Hold Time		t_{DHW}	Fig. 13	10	–	10	–	10	–	ns

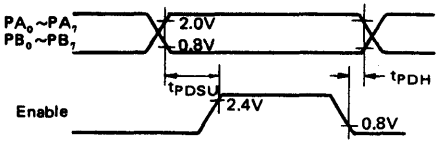
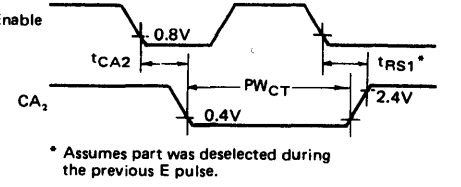


Figure 1 Peripheral Data Setup and Hold Times (Read Mode)



* Assumes part was deselected during the previous E pulse.

Figure 2 CA_2 Delay Time (Read Mode; CRA5=CRA3=1, CRA4=0)

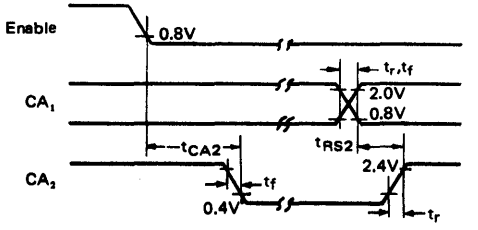


Figure 3 CA_2 Delay Time (Read Mode; CRA5=1, CRA3=CRA4=0)

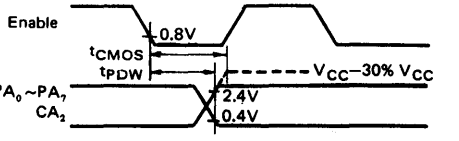
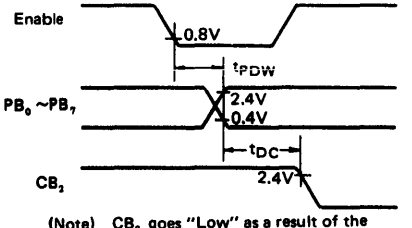
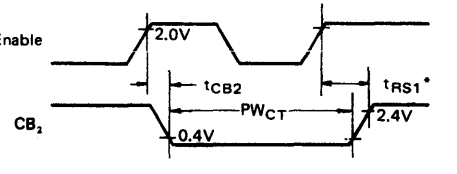


Figure 4 Peripheral CMOS Data Delay Times (Write Mode; CRA5=CRA3=1, CRA4=0)



(Note) CB_2 goes "Low" as a result of the positive transition of Enable.

Figure 5 Peripheral Data and CB_2 Delay Times (Write Mode; CRB5=CRB3=1, CRB4=0)



* Assumes part was deselected during the previous E pulse.

Figure 6 CB_2 Delay Time (Write Mode; CRB5=CRB3=1, CRB4=0)

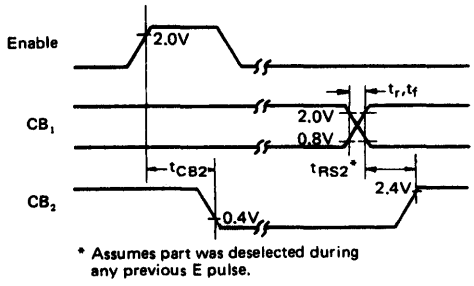


Figure 7 CB_2 Delay Time
(Write Mode; CRB5=1, CRB3=CRB4=0)

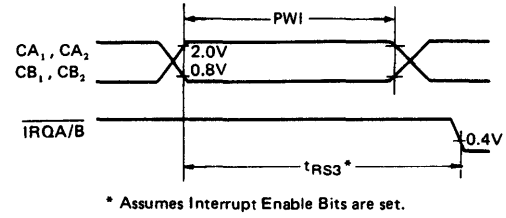


Figure 8 Interrupt Pulse Width and $\overline{IRQ}$ Response

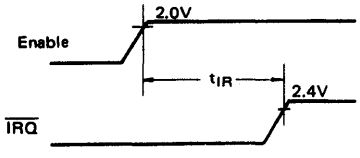


Figure 9 $\overline{IRQ}$ Release Time

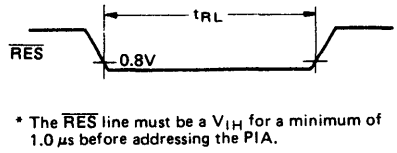


Figure 10 $\overline{RES}$ Low Time

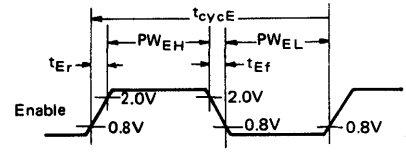


Figure 11 Enable Signal Characteristics

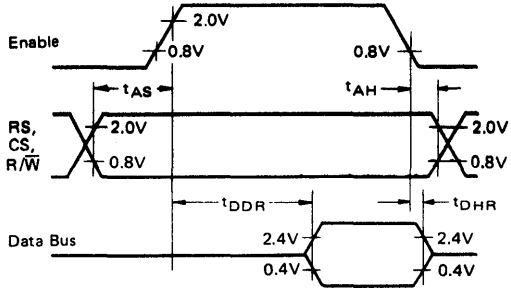


Figure 12 Bus Read Timing Characteristics
(Read Information from PIA)

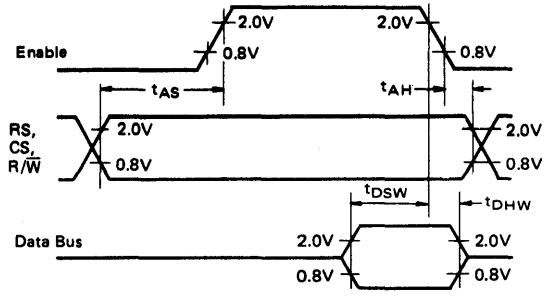
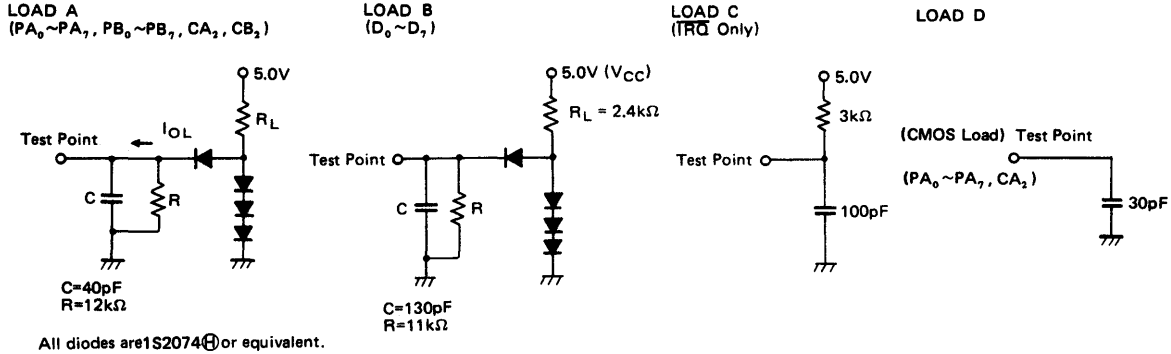


Figure 13 Bus Write Timing Characteristics
(Write Information into PIA)



Adjust R_L so that I_{OL} = 1.6mA, then test V_{OL}
Adjust R_L so that I_{OL} = 3.2mA, then test V_{OL}

Figure 14 Bus Timing Test Loads

■ PIA INTERFACE SIGNALS FOR MPU

The PIA interfaces to the HD6800 MPU with an eight-bit bi-directional data bus, three chip select lines, two register select lines, two interrupt request lines, read/write line, enable line and reset line. These signals, in conjunction with the HD6800 VMA output, permit the MPU to have complete control over the PIA. VMA should be utilized in conjunction with an MPU address line into a chip select of the PIA.

- **PIA Bi-Directional Data ($D_0 \sim D_7$)**

The bi-directional data lines ($D_0 \sim D_7$) allow the transfer of data between the MPU and the PIA. The data bus output drivers are three-state devices that remain in the high-impedance (off) state except when the MPU performs a PIA read operation. The R/$\overline{W}$ line is in the Read ("High") state when the PIA is selected for a Read operation.

- **PIA Enable (E)**

The enable pulse, E, is the only timing signal that is supplied to the PIA. Timing of all other signals is referenced to the leading and trailing edges of the E pulse. This signal will normally be a derivative of the HMCS6800 System ϕ_2 Clock. This signal must be continuous clock pulse.

- **PIA Read/Write (R/$\overline{W}$)**

This signal is generated by the MPU to control the direction of data transfers on the Data Bus. A "Low" state on the PIA line enables the input buffers and data is transferred from the MPU to the PIA on the E signal if the device has been selected. A "High" on the R/$\overline{W}$ line sets up the PIA for a transfer of data to the bus. The PIA output buffers are enabled when the proper address and the enable pulse E are present.

- **Reset ($\overline{RES}$)**

The active "Low" $\overline{RES}$ line is used to reset all register bits in the PIA to a logical zero "Low". This line can be used as a power-on reset and as a master reset during system operation.

- **PIA Chip Select (CS_0, CS_1 and $\overline{CS_2}$)**

These three input signals are used to select the PIA. CS_0 and CS_1 must be "High" and $\overline{CS_2}$ must be "Low" for selection of the device. Data transfers are then performed under the control of the E and R/$\overline{W}$ signals. The chip select lines must be stable for the duration of the E pulse. The device is deselected when any of the chip selects are in the inactive state.

- **PIA Register Select (RS_0 and RS_1)**

The two register select lines are used to select the various registers inside the PIA. These two lines are used in conjunction with internal Control Registers to select a particular register that is to be written or read.

The register and chip select lines should be stable for the duration of the E pulse while in the read or write cycle.

- **Interrupt Request ($\overline{IRQA}$ and $\overline{IRQB}$)**

The active "Low" Interrupt Request lines ($\overline{IRQA}$ and $\overline{IRQB}$) act to interrupt the MPU either directly or through interrupt priority circuitry. These lines are "open drain" (no load device on the chip). This permits all interrupt request lines to be tied together in a wire-OR configuration.

Each $\overline{IRQ}$ line has two internal interrupt flag bits that can cause the $\overline{IRQ}$ line to go "Low". Each flag bit is associated with a particular peripheral interrupt line. Also four interrupt enable bits are provided in the PIA which may be used to inhibit a particular interrupt from a peripheral device.

Servicing an interrupt by the MPU may be accomplished by a software routine that, on a prioritized basis, sequentially reads and tests the two control registers in each PIA for interrupt flag bits that are set.

The interrupt flags are cleared (zeroed) as a result of an MPU Read Peripheral Data Operation of the corresponding data register. After being cleared, the interrupt flag bit cannot be enabled to be set until the PIA is deselected during an E pulse. The E pulse is used to condition the interrupt control lines (CA_1, CA_2, CB_1, CB_2). When these lines are used as interrupt inputs at least one E pulse must occur from the inactive edge to the active edge of the interrupt input signal to condition the edge sense network. If the interrupt flag has been enabled and the edge sense circuit has been properly conditioned, the interrupt flag will be set on the next active transition of the interrupt input pin.

■ PIA PERIPHERAL INTERFACE LINES

The PIA provides two 8-bit bi-directional data buses and four interrupt/control lines for interfacing to peripheral devices.

- **Section A Peripheral Data ($PA_0 \sim PA_7$)**

Each of the peripheral data lines can be programmed to act as an input or output. This is accomplished by setting a "1" in the corresponding Data Direction Register bit for those lines which are to be outputs. A "0" in a bit of the Data Direction Register causes the corresponding peripheral data line to act as an input. During an MPU Read Peripheral Data Operation, the data on peripheral lines programmed to act as inputs appears directly on the corresponding MPU Data Bus lines.

The data in Output Register A will appear on the data lines that are programmed to be outputs. A logical "1" written into the register will cause a "High" on the corresponding data line while a "0" results in a "Low". Data in Output Register A may be read by an MPU "Read Peripheral Data A" operation when the corresponding lines are programmed as outputs. This data will be read properly if the voltage on the peripheral data lines is greater than 2.0 volts for a logic "1" output and less than 0.8 volt for a logic "0" output. Loading the output lines such that the voltage on these lines does not reach full voltage causes the data transferred into the MPU on a Read operation to differ from that contained in the respective bit of Output Register A.

- **Section B Peripheral Data ($PB_0 \sim PB_7$)**

The peripheral data lines in the B Section of the PIA can be programmed to act as either inputs or outputs in a similar manner to $PA_0 \sim PA_7$. However, the output buffers driving these lines differ from those driving lines $PA_0 \sim PA_7$. They have three-state capability, allowing them to enter a high impedance state when the peripheral data line is used as a input. In addition, data on the peripheral data lines $PB_0 \sim PB_7$ will be read properly from those lines programmed as outputs even if the voltages are below 2.0 volts for a "High". As outputs, these lines are compatible with standard TTL and may also be used as a source of up to 2.5 milliampere (typ.) at 1.5 volts to directly drive the base of a transistor switch.

- **Interrupt Input (CA_1 and CB_1)**

Peripheral Input lines CA_1 and CB_1 are input only lines that set the interrupt flags of the control registers. The active transition for these signals is also programmed by the two control registers.

- **Peripheral Control (CA_2)**

The peripheral control line CA_2 can be programmed to act as an interrupt input or as a peripheral control output. As an output, this line is compatible with standard TTL. The function of this signal line is programmed with Control Register A.

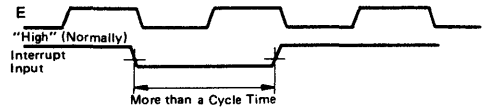
- **Peripheral Control (CB_2)**

Peripheral Control line CB_2 may also be programmed to act as an interrupt input or peripheral control output. As an input,

this line has "High" input impedance and is compatible with standard TTL. As an output it is compatible with standard TTL and may also be used as a source of up to 2.5 milliampere (typ) at 1.5 volts to directly drive the base of a transistor switch. This line is programmed by Control Register B.

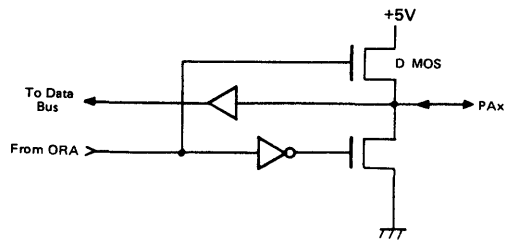
(NOTE) 1. Interrupt inputs CA_1, CA_2, CB_1 and CB_2 shall be used at normal "High" level. When interrupt inputs are "Low" at reset ($\overline{RES}$ = "Low"), interrupt flags CRA6, CRA7, CRB6 and CRB7 may be set.

2. Pulse width of interrupt inputs CA_1, CA_2, CB_1 and CB_2 shall be greater than a E cycle time. In the case that "High" time of E signal is not contained in Interrupt pulse, an interrupt flag may not be set.

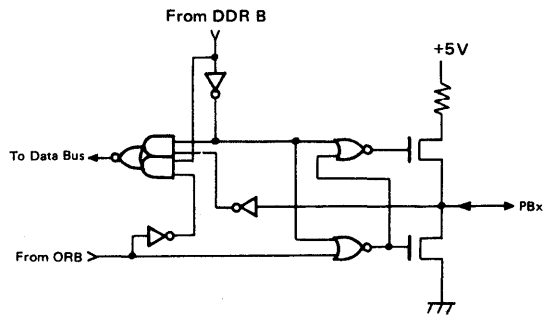


- **The equivalent Circuit of the Lines on Peripheral side**

The equivalent circuit of the lines on Peripheral side is shown in Fig. 15. The output circuits of A port is different from that of B port. When the port is used as input, the input is pullup to V_{CC} side through load MOS in A port and B port becomes "Off" (high impedance).



(a) Section A



(b) Section B

Figure 15 Peripheral Data Bus

■ INTERNAL CONTROLS

There are six locations within the PIA accessible to the MPU data bus: two Peripheral Registers, two Data Direction Registers, and two Control Registers. Selection of these locations is controlled by the RS_0 and RS_1 inputs together with bit 2 in the Control Register, as shown in Table 1.

Table 1 Internal Addressing

RS_1	RS_0	Control Register Bit CRA2	Control Register Bit CRB2	Location Selected
0	0	1	x	Peripheral Register A*
0	0	0	x	Data Direction Register A
0	1	x	x	Control Register A
1	0	x	1	Peripheral Register B*
1	0	x	0	Data Direction Register B
1	1	x	x	Control Register B

x = Don't Care
* Peripheral interface register is a generic term containing peripheral data bus and output register.

- **Initialization**

A "Low" reset line has the effect of zeroing all PIA registers. This will set PA_0~PA_7, PB_0~PB_7, CA_2 and CB_2 as inputs, and all interrupts disabled. The PIA must be configured during the restart program which follows the reset.

Details of possible configurations of the Data Direction and Control Register are as follows.

- **Data Direction Registers (DDRA and DDRB)**

The two Data Direction Registers allow the MPU to control the direction of data through each corresponding peripheral data line. A Data Direction Register bit set at "0" configures the corresponding peripheral data line as an input; a "1" results in an output.

- **Control Registers (CRA and CRB)**

The two Control Registers (CRA and CRB) allow the MPU to control the operation of the four peripheral control lines CA_1, CA_2, CB_1 and CB_2. In addition they allow the MPU to enable the interrupt lines and monitor the status of the interrupt flags. Bits 0 through 5 of the two registers may be written or read by the MPU when the proper chip select and register select signals are applied. Bits 6 and 7 of the two registers are read only and are modified by external interrupts occurring on control lines CA_1, CA_2, CB_1 or CB_2. The format of the control words is shown in Table 2.

Table 2 Control Word Format

	7	6	5	4	3	2	1	0
CRA	IRQA1	IRQA2	CA_2 Control			DDRA Access	CA_1 Control	
CRB	IRQB1	IRQB2	CB_2 Control			DDRB Access	CB_1 Control	

Data Direction Access Control Bit (CRA2 and CRB2)

Bit 2 in each Control register (CRA and CRB) allows selection of either a Peripheral Interface Register or the Data Direction Register when the proper register select signals are applied to RS_0 and RS_1.

Interrupt Flags (CRA6, CRA7, CRB6, and CRB7)

The four interrupt flag bits are set by active transitions of signals on the four Interrupt and Peripheral Control lines when those lines are programmed to be inputs. These bits cannot be set directly from the MPU Data Bus and are reset indirectly by a Read Peripheral Data Operation on the appropriate section.

Control of CA_1 and CB_1 Interrupt Lines (CRA0, CRB0, CRA1, and CRB1)

The two lowest order bits of the control registers are used to control the interrupt input lines CA_1 and CB_1. Bits CRA0 and CRB0 are used to enable the MPU interrupt signals $\overline{IRQA}$ and $\overline{IRQB}$, respectively. Bits CRA1 and CRB1 determine the active transition of the interrupt input signals CA_1 and CB_1 (Table 3)

Control of CA_2 and CB_2 Peripheral Control Lines (CRA3, CRA4, CRA5, CRB3, CRB4, and CRB5)

Bits 3, 4 and 5 of the two control registers are used to control the CA_2 and CB_2 Peripheral Control lines. These bits determine if the control lines will be an interrupt input or an output control signal. If bit CRA5 (CRB5) is "0" CA_2 (CB_2) is an interrupt input line similar to CA1 (CB1) (Table 4). When CRA5 (CRB5) is "1", CA_2 (CB_2) becomes an output signal that may be used to control peripheral data transfers. When in the output mode, CA_2 and CB_2 have slightly different characteristics (Table 5 and 6).

Table 3 Control of Interrupt Inputs CA_1 and CB_1

CRA1 (CRB1)	CRA0 (CRB0)	Interrupt Input CA_1 (CB_1)	Interrupt Flag CRA7 (CRB7)	MPU Interrupt Request $\overline{IRQA}$ ($\overline{IRQB}$)
0	0	↓ Active	Set "1" on ↓ of CA_1 (CB_1)	Disabled – $\overline{IRQ}$ remains "High"
0	1	↓ Active	Set "1" on ↓ of CA_1 (CB_1)	Goes "Low" when the interrupt flag bit CRA7 (CRB7) goes "1"
1	0	↑ Active	Set "1" on ↑ of CA_1 (CB_1)	Disabled – $\overline{IRQ}$ remains "High"
1	1	↑ Active	Set "1" on ↑ of CA_1 (CB_1)	Goes "Low" when the interrupt flag bit CRA7 (CRB7) goes "1"

(Notes)
1. ↑ indicates positive transition ("Low" to "High")
2. ↓ indicates negative transition ("High" to "Low")
3. The Interrupt flag bit CRA7 is cleared by an MPU Read of the A Peripheral Register and CRB7 is cleared by an MPU Read of the B Peripheral Register.
4. If CRA0 (CRB0) is "0" when an interrupt occurs (Interrupt disabled) and is later brought "1", $\overline{IRQA}$ ($\overline{IRQB}$) occurs after CRA0 (CRB0) is written to a "1".

Table 4 Control of CA_2 and CB_2 as Interrupt Inputs – CRA5 (CRB5) is "0"

CRA5 (CRB5)	CRA4 (CRB4)	CRA3 (CRB3)	Interrupt Input CA_2 (CB_2)	Interrupt Flag CRA6 (CRB6)	MPU Interrupt Request $\overline{IRQA}$ ($\overline{IRQB}$)
0	0	0	↓ Active	Set "1" on ↓ of CA_2 (CB_2)	Disabled – $\overline{IRQ}$ remains "High"
0	0	1	↓ Active	Set "1" on ↓ of CA_2 (CB_2)	Goes "Low" when the interrupt flag bit CRA6 (CRB6) goes "1"
0	1	0	↑ Active	Set "1" on ↑ of CA_2 (CB_2)	Disabled – $\overline{IRQ}$ remains "High"
0	1	1	↑ Active	Set "1" on ↑ of CA_2 (CB_2)	Goes "Low" when the interrupt flag bit CRA6 (CRB6) goes "1"

(Notes)
1. ↑ indicates positive transition ("Low" to "High")
2. ↓ indicates negative transition ("High" to "Low")
3. The interrupt flag bit CRA6 is cleared by an MPU Read of the A Peripheral Register and CRB6 is cleared by an MPU Read of the B Peripheral Register.
4. If CRA3 (CRB3) is "0" when an interrupt occurs (Interrupt disabled) and is later brought "1", $\overline{IRQA}$ ($\overline{IRQB}$) occurs after CRA3 (CRB3) is written to a "1".

Table 5 Control of CB_2 as an Output – CRB5 is "1"

CRB5	CRB4	CRB3	CB_2	
			Cleared	Set
1	0	0	"Low" on the positive transition of the first E pulse after MPU Write "B" Data Register operation.	"High" when the interrupt flag bit CRB7 is set by an active transition of the CB_1 signal. (See Figure 16)
1	0	1	"Low" on the positive transition of the first E pulse after an MPU Write "B" Data Register operation.	"High" on the positive edge of the first "E" pulse following an "E" pulse which occurred while the part was deselected. (See Figure 16)
1	1	0	"Low" (The content of CRB3 is output on CB_2)	
1	1	1	"High" (The content of CRB3 is output on CB_2)	

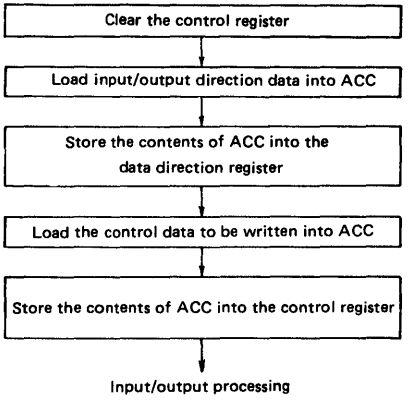
Table 6 Control of CA_2 as an Output – CRA5 is "1"

CRA5	CRA4	CRA3	CA_2	
			Cleared	Set
1	0	0	"Low" on negative transition of E after an MPU Read "A" Data Operation.	"High" when the interrupt flag bit CRA7 is set by an active transition of the CA_1 signal. (See Figure 16)
1	0	1	"Low" on negative transition of E after an MPU Read "A" Data operation.	"High" on the negative edge of the first "E" pulse which occurs during a deselect. (See Figure 16)
1	1	0	"Low" (The content of CRA3 is output on CA_2)	
1	1	1	"High" (The content of CRA3 is output on CA_2)	

■ PIA OPERATION

• Initialization

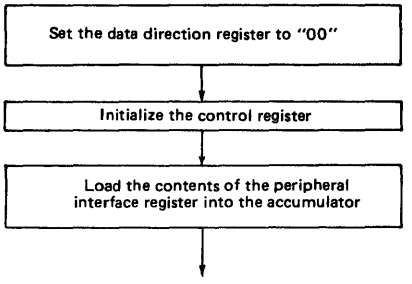
When the external reset input $\overline{RES}$ goes "Low", all internal registers are cleared to "0". Periperal data port (PA_0~PA7, PB_0~PB_7) is defined to be input and control lines (CA_1, CA_2, CB_1 and CB_2) are defined to be the interrupt input lines. PIA is also initialized by software sequence as follows.



- Program the data direction register access bit of the control register to "0" to allow to access the dada direction register.
- The data of the control line function is set into the accumulator, of which Data Direction Register Access Bit shall be programmed to "1".
- Transfer the control data from the accumulator into the control register.

• Read/Write Operation Not Using Control Lines

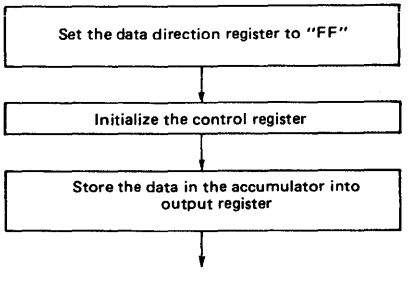
<Read Operation>




```
CLR   CRA
CLR   DDRA
LDAA  #$04
STAA  CRA
      :
      :
LDAA  PIRA
```

- Clear the DDRA access bit of the control register to "0".
- Clear all bits of the dada direction register.
- Set DDRA access bit of the control register to "1" to allow to access the peripheral interface register.

<Write Operation>




```
CLR   CRA
LDAA  #$FF
STAA  DDRB
      :
LDAA  #$04
STAA  CRB
      :
LDAA  DATA
STAA  PIRB
```

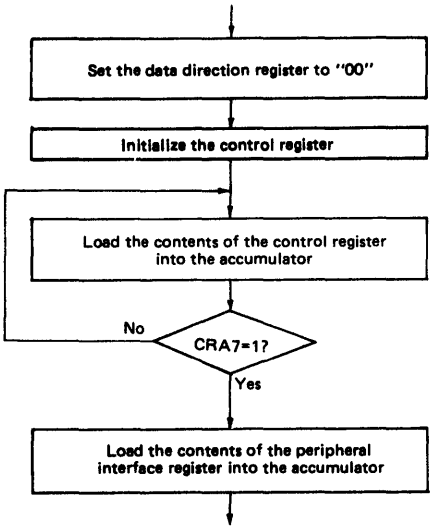
- Set DDRB access bit of the control register to "0".
- Set all bits of the data direction register to "FF".
- Set DDRB access bit of the control register to "1" to allow to access the peripheral interface register.
- Write the data into the peripheral interface register.

• **Read/Write Operating Using Control Lines**

Read/write request from peripherals shall be put into the control lines as an interrupt signal, and then MPU reads or writes after detecting interrupt request.

< Read >

The following case is that Port A is used and that the rising edge of CA_1 indicates the request for read from peripherals.



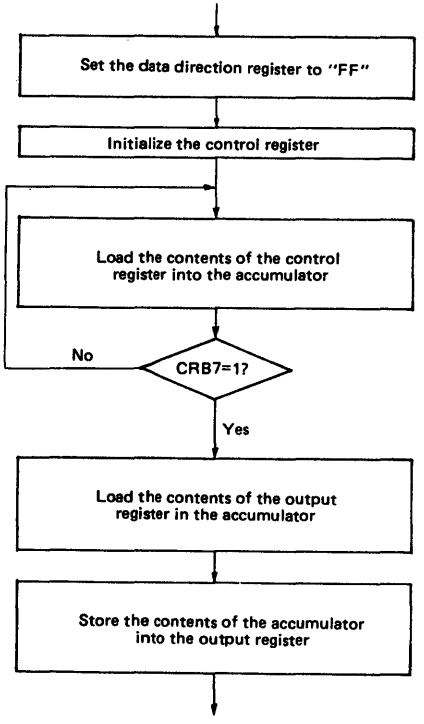

```
CLR   CRA    • Set the DDRA access bit to "0".
CLR   DDRA   • Set all bits of the data direction register to "0".

LDAA  #$06   • Program the rising edge of CA1 to be active. IRQA is masked
STAA  CRA      and DDRA access bit is set to 1.

LOOP  LDAA CRA
      BPL  LOOP   • Check whether the read request comes from peripherals
                    or not.

LDAA  PIRA   • Load the data from the peripheral interface register into the
               accumulator. CRA flag is reset after this read operation.
```

To read the peripheral data, the data is directly transfered to the data buses D_0~D_7 through PA_0~PA_7 or PB_0~PB_7 and they are not latched in the PIA. If necessary, the data should be held in the external latch until MPU completes reading it.

When initializing the control register, interrupt flag bit (CRA7, CRA6, CRB7, CRB6) cannot be written from MPU. If necessary the interrupt flag must be reset by dummy read of Peripheral Register A and B.

<Write>

Write operation using the interrupt signal is as follows. In this case, B port is used and interrupt request is input to CB_1. And the $\overline{IRQ}$ flag is set at the rising edge of CB_1.




```
        CLR   CRB      • Set DDRB access bit to "0"
        LDAA  #$FF   }
        STAA  DDRB   }  • Set all bits of DDRB to output "1".
        LDAA  #$06   }
        STAA  CRB    }  • Program the rising edge of CB1 to be active. IRQB is masked
                           and DDRB access bit is set to "1".
LOOP    LDAA  CRB    }
        BPL   LOOP   }  • Check whether the write request comes from peripherals
                           or not.
        LDAA  PIRB      • Reset the CRB7 flag by the dummy read of the
                           peripheral interface register.
        STAB  PIRB      • Store the data of the accumulator B to the peripheral
                           interface register.
```

Interrupt request flag bits (CRA7, CRA6, CRB7 and CRB6) cannot be written and they cannot be also reset by write operation to the peripheral interface register. So dummy read of peripheral interface register is needed to reset the flags.

To accept the next interrupt, it is essential to reset indirectly the interrupt flag by dummy read of peripheral interface register.

Software poling method mentioned above requires MPU to continuously monitor the control register to detect the read/write request from peripherals. So other programs cannot run at the same time. To avoid this problem, hardware interrupt may be used. The MPU is interrupted by $\overline{IRQA}$ or $\overline{IRQB}$ when the read/write request is occurred from peripherals and then MPU analyzes cause of the interrupt request during interrupt processing.

- **Handshake Mode**

The functions of CRA and CRB are similar but not identical in the hand-shake modes. Port A is used for read hand-shake operation and Port B is used for write hand-shake mode.

CA_1 and CB_1 are used for interrupt input requests and CA_2 and CB_2 are control outputs (answer) in hand-shake mode.

Fig. 16, Fig. 17 and Fig. 18 show the timing of hand-shake mode.

< Read Hand-shake Mode>

CRA5="1", CRA4="0" and CRA3="0"

(1) A peripheral device puts the 8-bit data on the peripheral data lines after the control output CA_2 goes "Low".

(2) The peripheral requests MPU to read the data by using CA_1 input.

(3) CRA7 flag is set and CA_2 becomes "High" (CA_2 automatically becomes "High" by the interrupt CA_1). This indicates the peripheral to maintain the current data and not to transfer the next data.

(4) MPU accepts the read request by $\overline{IRQA}$ hardware interrupt or CRA read. Then MPU reads the peripheral register A.

(5) CA_2 goes "Low" on the following edge of read Enable pulse. This informs that the peripheral can set the next data to port A.

<Write Hand-shake>

CRB5 = "1", CRB4 = "0" and CRB3 = "0"

(1) A peripheral device requests MPU to write the data by using CB_1 input. CB_2 output remains "High" until MPU write data to the peripheral interface register.

(2) CRB7 flag is set and MPU accepts the write request.

(3) MPU reads the peripheral interface register to reset CRB7 (dummy read).

(4) Then MPU write data to the peripheral interface register. The data is output to port B through the output register.

(5) CB_2 automatically becomes "Low" to tell the peripheral that new data is on port B.

(6) The peripheral read the data on Port B peripheral data lines and set CB_1 to "Low" to tell MPU that the data on the peripheral data lines has been taken and that next data can be written to the peripheral interface register.

<Pulse mode>

CRA5 = "1", CRA4 = "0" and CRA3 = "1"
CRB5 = "1", CRB4 = "0" and CRB3 = "1"
This mode is shown in Figure 16, Figure 19 and Figure 20.

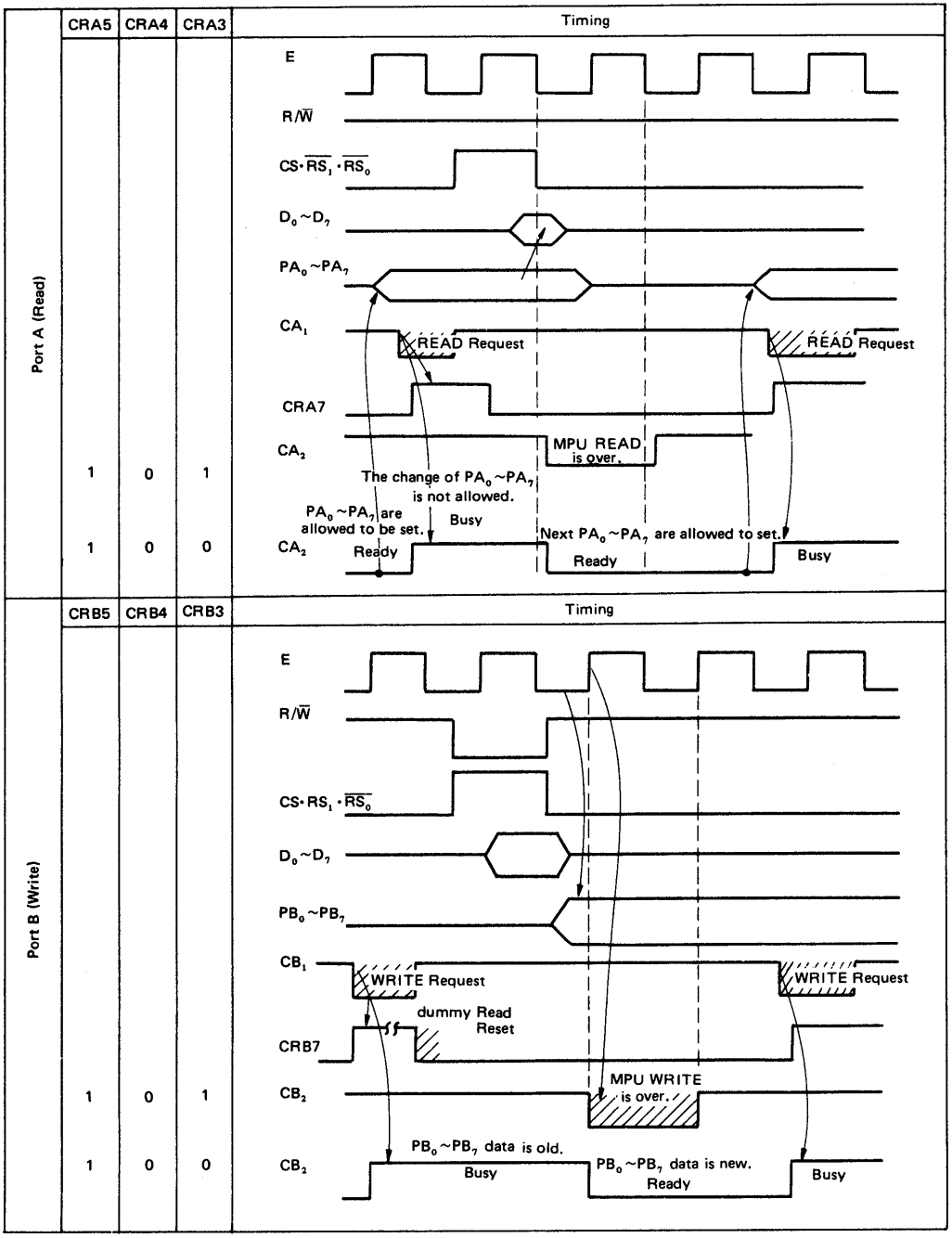


Figure 16 Timing of Hand-shake Mode and Pulse Mode

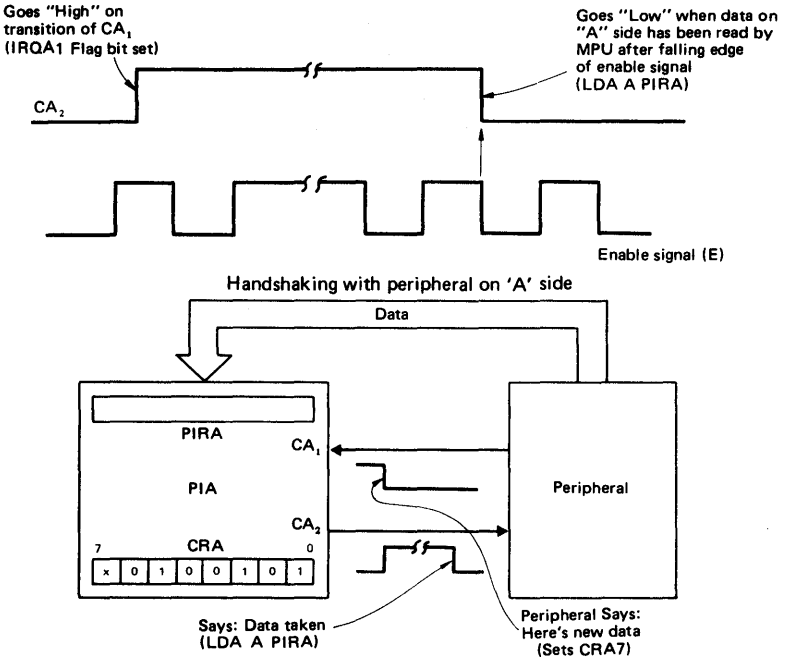


Figure 17 Bits 5, 4, 3 of CRA = 100 (Hand-shake Mode)

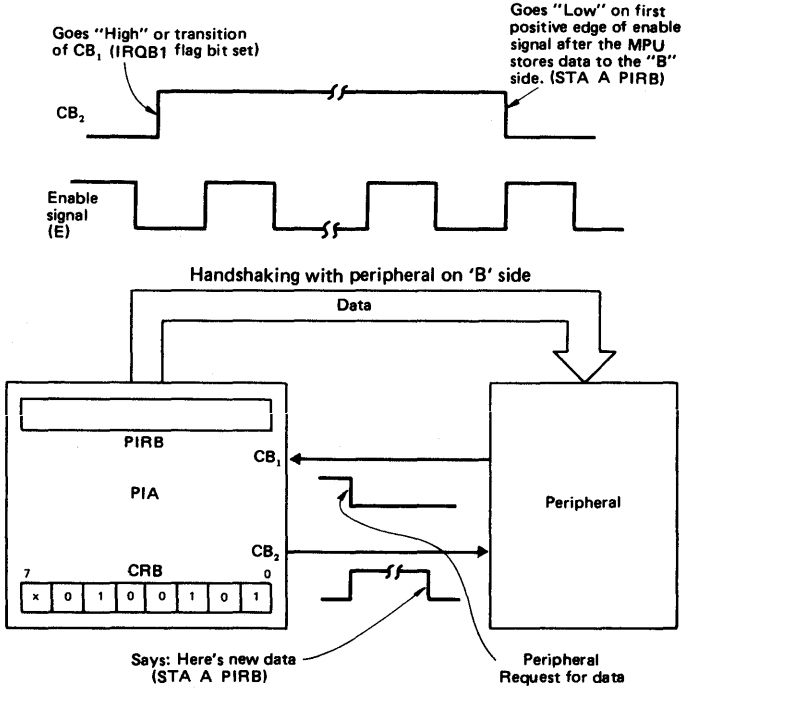


Figure 18 Bits 5, 4, 3 of CRB = 100 (Hand-shake Mode)

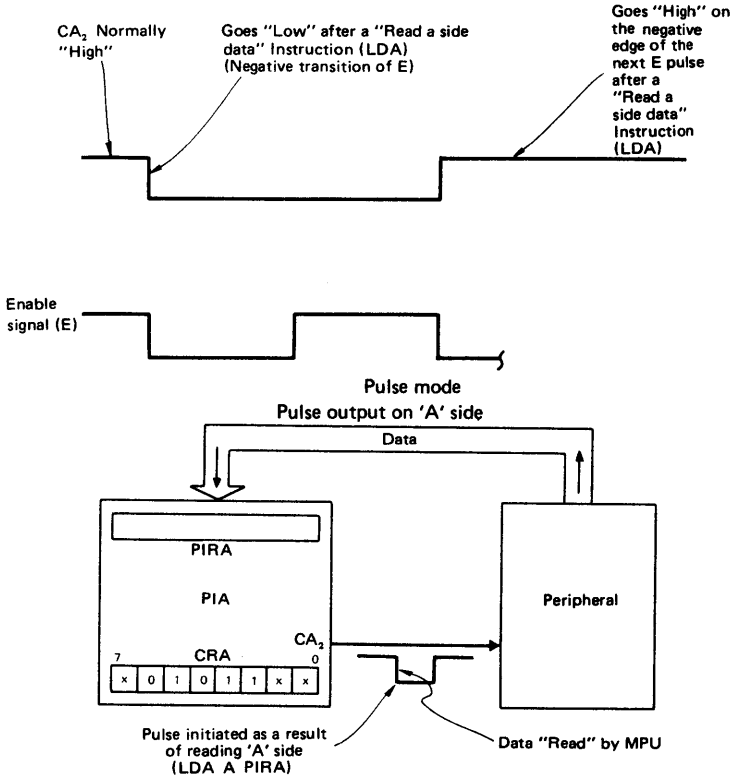


Figure 19 Bits 5, 4, 3 of CRA = 101 (Pulse Mode)

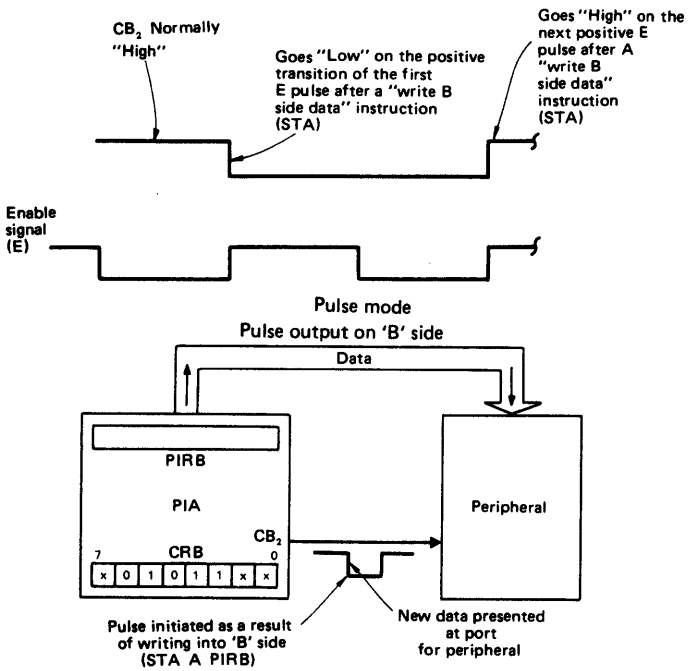


Figure 20 Bits 5, 4, 3 of CRB=101 (Pulse Mode)

■ SUMMARY OF CONTROL REGISTERS CRA AND CRB

Control registers CRA and CRB have total control of CA_1, CA_2, CB_1, and CB_2 lines. The status of eight bits of the control registers may be read into the MPU. However, the MPU can only write into Bit 0 through Bit 5 (6 bits), since Bit 6 and Bit 7 are set only by CA_1, CA_2, CB_1, or CB_2.

• Addressing PIAs

Before addressing PIAs, the data direction (DDR) must first be loaded with the bit pattern that defines how each line is to function, i.e., as an input or an output. A logic "1" in the data direction register defines the corresponding line as an output while a logic "0" defines the corresponding line as an input. Since the DDR and the peripheral interface resister have the same address, the control register bit 2 determines which register is being addressed. If Bit 2 in the control register is a logic "0", then the DDR is addressed. If Bit 2 in the control register is a logic "1", the peripheral interface register is addressed. Therefore, it is essential that the DDR be loaded first before setting Bit 2 of the control register.

<Example>

Given a PIA with an address of 4004, 4005, 4006, and 4007. 4004 is the address of the A side peripheral interface register. 4005 is the address of the A side control register. 4006 is the address of the B side peripheral interface register. 4007 is the address of the B side control register. On the A side, Bits 0, 1, 2, and 3 will be defined as inputs, while Bits 4, 5, 6, and 7 will be used as outputs. On the B side, all lines will be used as outputs.

```
PIA1AD = 4004      (DDRA, PIRA)
PIA1AC = 4005      (CRA)
PIA1BD = 4006      (DDRB, PIRB)
PIA1BC = 4007      (CRB)

1.  LDA A #%11110000   (4 outputs, 4 inputs)
2.  STA A    PIA1AD    (Loads A DDR)
3.  LDA A #%11111111   (All outputs)
4.  STA A    PIA1BD    (Loads B DDR)
5.  LDA A #%00000100   (Sets Bit 2)
6.  STA A    PIA1AC    (Bit 2 set in A control register)
7.  STA A    PIA1BC    (Bit 2 set in B control register)
```

Statement 2 addresses the DDR, since the control register (Bit 2) has not been loaded. Statements 6 and 7 load the control registers with Bit 2 set, so addressing PIA1AD or PIA1BD accesses the peripheral interface register.

• PIA Programming Via The Index Register

The program shown in the previous section can be accomplished using the Index Register.

```
1.  LDX   #$F004
2.  STX   PIA1AD     $F0→PIA1AD ;$04→PIA1AC
3.  LDX   #$FF04
4.  STX   PIA1BD     $FF→PIA1BD;$04→PIA1BC
```

Using the index register in this example has saved six bytes of program memory as compared to the program shown in the previous section.

• Active Low Outputs

When all the outputs of given PIA port are to be active "Low" (True ≦ 0.4 volts), the following procedure should be used.

a) Set Bit 2 in the control register.
b) Store all 1s ($FF) in the peripheral interface register.
c) Clear Bit 2 in the control register.
d) Store all 1s ($FF) in the data direction register.
e) Store control word (Bit 2 = 1) in control register.

<Example>

The B side of PIA1 is set up to have all active low outputs. CB_1 and CB_2 are set up to allow interrupts in the HANDSHAKE MODE and CB_1 will respond to positive edges ("Low"-to-"High" transitions). Assume reset conditions. Addresses are set up and equated to the same labels as previous example.

```
1.  LDA A #4
2.  STA A PIA1BC     Set Bit 2 in PIA1BC (control register)
3.  LDA B #$FF
4.  STA B PIA1BD     All 1s in peripheral interface register
5.  CLR   PIA1BC     Clear Bit 2
6.  STA B PIA1BD     All 1s in data direction register
7.  LDA A #$27
8.  STA A PIA1BC     00100111→→ control register
```

The above procedure is required in order to avoid outputs going "Low", to the active "Low" TRUE STATE, when all ls are stored to the data direction register as would be the case if the normal configuration procedure were followed.

• Interchanging RS_0 And RS_1

Some system applications may require movement of 16 bits of data to or from the "outside world" via two PIA ports (A side + B side). When this is the case it is an advantage to interconnect RS_1 and RS_0 as follows.

RS_0 to A1 (Address Line A1)
RS_1 to A0 (Address Line A0)

This will place the peripheral interface registers and control registers side by side in the memory map as follows.

Table	Example Address	
PIA1AD	$4004	(DDRA, PIRA)
PIA1BD	$4005	(DDRB, PIRB)
PIA1AC	$4006	(CRA)
PIA1BC	$4007	(CRB)

The index register or stackpointer may be used to move the 16-bit data in two 8-bit bytes with one instruction. As an example:

LDX PIA1AD PIA1AD →→ IX_H: PIA1BD →→ IX_L

• PIA – After Reset

When the $\overline{RES}$ (Reset Line) has been held "Low" for a minimum of one microsecond, all registers in the PIA will be cleared.

Because of the reset conditions, the PIA has been defined as

follows.

1. All I/O lines to the "outside world" have been defined as inputs.
2. CA_1, CA_2, CB_1, and CB_2 have been defined as interrupt input lines that are negative edge sensitive.
3. All the interrupts on the control lines are masked. Setting of interrupt flag bits will not cause $\overline{IRQA}$ or $\overline{IRQB}$ to go "Low".

■ SUMMARY OF CA_1-CB_1 PROGRAMMING

Bits 1 and 0 of the respective control registers are used to program the interrupt input control lines CA_1 and CB_1.

b1	b0	
0	0	b1 = Edge (0 = –, 1 = +)
0	1	b0 = Mask (0 = Mask, 1 = Allow)
1	0	
1	1	

Note that this is the same logic as Bits 4 and 3 for CA_2-CB_2 when CA_2-CB_2 are programmed as inputs.

■ SUMMARY OF CA_2-CB_2 PROGRAMMING

Bits 5, 4, and 3 of the control registers are used to program the operation of CA_2-CB_2.

	b5	b4	b3	
CA_2–CB_2 Input Mode →	0	0(–)	0	(Mask) CA_2–CB_2 Input Mode
	0	0(–)	1	(Allow) b4 = Edge (0 = –, 1 = +)
	0	1(+)	0	(Mask) b3 = Mask (0 = Mask,
	0	1(+)	1	(Allow) 1 = Allow)
CA_2–CB_2 Output Mode →	1	0	0	0 – Handshake Mode
	1	0	1	1 – Pulse Mode
	1	1	0	b3 Following Mode
	1	1	1	b3 Following Mode

I/O As Follow:
Control Lines:
CA_1 – Positive Edge, Allow Interrupt
CA_2 – Pulse Mode
CB_1 – Negative Edge, Mask Interrupt
CB_2 – Hand Shake Mode

Assume Reset Condition
PIA1AD
PIA1AC
PIA1BD
PIA1BC

PIA Configuration Solution

LDA A #$BC	10111100
STA A PIA1AD	I/O to DDRA
LDA A #$FF	1111 1111
STA A PIA1BD	I/O to DDRB
LDA A #$2F	0010 1111
STA A PIA1AC	To "A" Control
LDA A #$24	0010 0100
STA A PIA1BC	To "B" Control

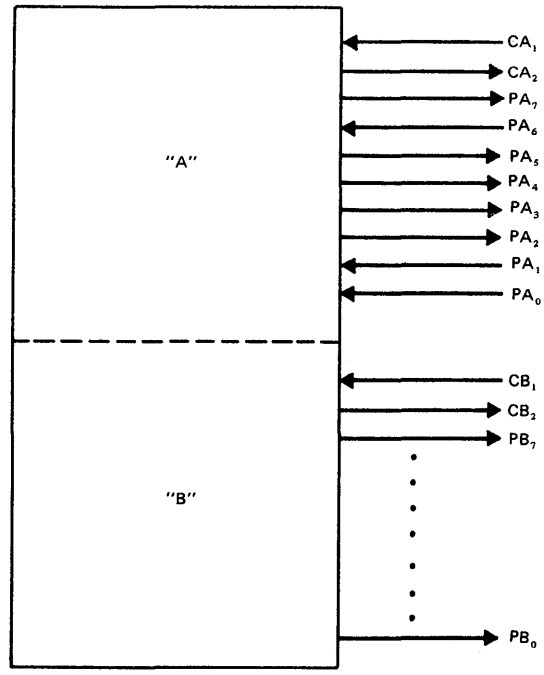


Figure 21 PIA Configuration Problem

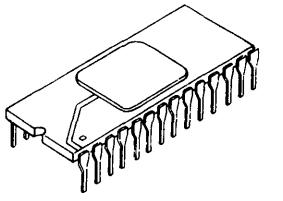
HD6840, HD68A40, HD68B40

PTM (Programmable Timer Module)

The HD6840 is a programmable subsystem component of the HMCS6800 family designed to provide variable system time intervals.

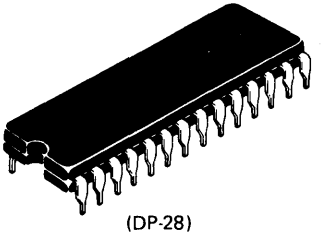
The HD6840 has three 16-bit binary counters, three corresponding control registers and a status register. These counters are under software control and may be used to cause system interrupts and/or generate output signals. The HD6840 may be utilized for such tasks as frequency measurements, event counting, interval measuring and similar tasks. The device may be used for square wave generation, gated delay signals, single pulses of controlled duration, and pulse width modulation as well as system interrupts.

HD6840, HD68A40, HD68B40



(DC-28)

HD6840P, HD68A40P, HD68B40P

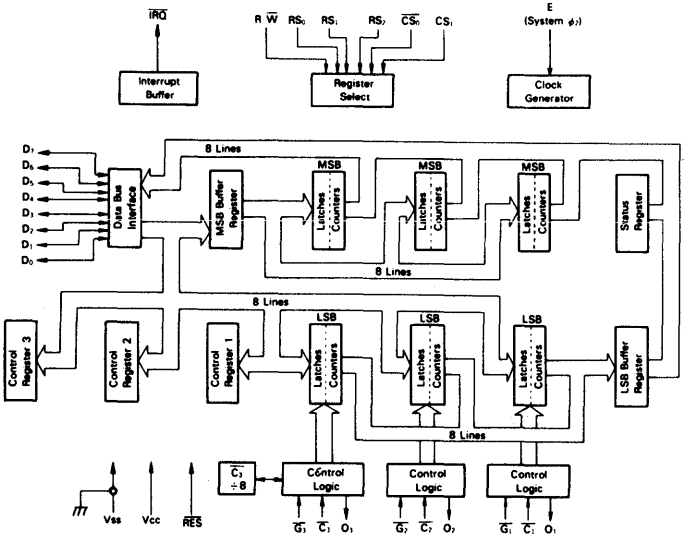


(DP-28)

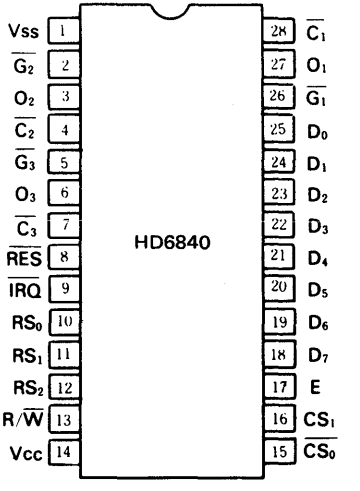
■ FEATURES

- Operates from a Single 5 Volts Power Supply
- Fully TTL Compatible
- Single System Clock Required (E)
- Selectable Prescaler on Timer 3 Capable of 4 MHz for the HD6840, 6 MHz for the HD68A40 and 8 MHz for the HD68B40
- Programmable Interrupts ($\overline{IRQ}$) Output to MPU
- Readable Down Counter Indicates Counts to Go until Time-Out
- Selectable Gating for Frequency or Pulse-Width Comparison
- $\overline{RES}$ Input
- Three Asynchronous External Clock and Gate/Trigger Inputs Internally Synchronized
- Three Maskable Outputs
- Compatible with MC6840, MC68A40 and MC68B40

■ BLOCK DIAGRAM



■ PIN ARRANGEMENT



(Top View)

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC}*	-0.3~+7.0	V
Input Voltage	V_{in}*	-0.3~+7.0	V
Operating Temperature	T_{opr}	- 20~+ 75	°C
Storage Temperature	T_{stg}	- 55~+150	°C

* With respect to V_{SS} (SYSTEM GND)

[NOTE] Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ RECOMMENDED OPERATING CONDITIONS

Item	Symbol	min	typ	max	Unit
Supply Voltage	V_{CC}*	4.75	5.0	5.25	V
Input Voltage	V_{IL}*	-0.3	–	0.8	V
	V_{IH}*	2.2	–	V_{CC}	V
Operating Temperature	T_{opr}	- 20	25	75	°C

* With respect to V_{SS} (SYSTEM GND)

■ ELECTRICAL CHARACTERISTICS

• DC CHARACTERISTICS (V_{CC} = 5V ± 5%, V_{SS} = 0V, Ta = -20 ~ +75°C, unless otherwise noted.)

Item	Symbol	Test Condition		min	typ*	max	Unit
Input "High" Voltage	V_{IH}			2.2	–	V_{CC}	V
Input "Low" Voltage	V_{IL}			-0.3	–	0.8	V
Input Leakage Current	I_{in}	V_{in} = 0 ~ 5.25V (Except D_0 ~ D_7)		-2.5	–	2.5	μA
Three-State Input Current (off-state)	I_{TSI}	V_{in} = 0.4 ~ 2.4V, V_{CC} = 5.25V (D_0 ~ D_7)		- 10	–	10	μA
Output "High" Voltage	V_{OH}	I_{LOAD} = -205 μA (D_0 ~ D_7)		2.4	–	–	V
		I_{LOAD} = -200 μA (Other Outputs)					
Output "Low" Voltage	V_{OL}	I_{LOAD} = 1.6 mA (D_0 ~ D_7)		–	–	0.4	V
		I_{LOAD} = 3.2 mA (O_1 ~ O_3, $\overline{IRQ}$)					
Output Leakage Current (off-state)	I_{LOH}	V_{OH} = 2.4V ($\overline{IRQ}$)		–	–	10	μA
Power Dissipation	P_D			–	330	550	mW
Input Capacitance	C_{in}	V_{in} = 0V, Ta = 25°C, f = 1 MHz	D_0 ~ D_7	–	–	12.5	pF
			Other Input	–	–	7.5	
Output Capacitance	C_{out}	V_{in} = 0V, Ta = 25°C, f = 1 MHz	$\overline{IRQ}$	–	–	5.0	pF
			O_1, O_2, O_3	–	–	10	

* Ta = 25°C, V_{CC} = 5.0V

• AC CHARACTERISTICS (V_{CC} = 5V ± 5%, V_{SS} = 0V, Ta = -20 ~ +75°C, unless otherwise noted.)

1. MPU READ TIMING

Item	Symbol	Test Condition	HD6840			HD68A40			HD68B40			Unit
			min	typ	max	min	typ	max	min	typ	max	
Enable Cycle Time	t_{cycE}	Fig. 1	1.0	–	10	0.666	–	10	0.5	–	10	μs
Enable "High" Pulse Width	PW_{EH}		0.45	–	4.5	0.280	–	4.5	0.22	–	4.5	μs
Enable "Low" Pulse Width	PW_{EL}		0.43	–	–	0.280	–	–	0.21	–	–	μs
Enable Rise and Fall Time	t_{Er}, t_{Ef}		–	–	25	–	–	25	–	–	25	ns
Address Set Up Time	t_{AS}		140	–	–	140	–	–	70	–	–	ns
Data Delay Time	t_{DDR}		–	–	320	–	–	220	–	–	180	ns
Data Hold Time	t_H		10	–	–	10	–	–	10	–	–	ns
Address Hold Time	t_{AH}		10	–	–	10	–	–	10	–	–	ns
Data Access Time	t_{ACC}		–	–	480	–	–	360	–	–	250	ns

2. MPU WRITE TIMING

Item	Symbol	Test Condition	HD6840			HD68A40			HD68B40			Unit
			min	typ	max	min	typ	max	min	typ	max	
Enable Cycle Time	t_{cycE}	Fig. 2	1.0	–	10	0.666	–	10	0.5	–	10	μs
Enable "High" Pulse Width	PW_{EH}		0.45	–	4.5	0.280	–	4.5	0.22	–	4.5	μs
Enable "Low" Pulse Width	PW_{EL}		0.43	–	–	0.280	–	–	0.21	–	–	μs
Enable Rise and Fall Time	t_{Er}, t_{Ef}		–	–	25	–	–	25	–	–	25	ns
Address Set Up Time	t_{AS}		140	–	–	140	–	–	70	–	–	ns
Data Set Up Time	t_{DSW}		195	–	–	80	–	–	60	–	–	ns
Data Hold Time	t_H		10	–	–	10	–	–	10	–	–	ns
Address Hold Time	t_{AH}		10	–	–	10	–	–	10	–	–	ns

3. TIMING OF PTM SIGNAL

Item			Symbol	Test Condition		HD6840		HD68A40		HD68B40		Unit
						min	max	min	max	min	max	
Input Rise and Fall Times	$\overline{C}$, $\overline{G}$, $\overline{RES}$		t_r, t_f	Fig. 3, Fig. 4		–	1.0*	–	0.666*	–	0.5*	μs
Input "Low" Pulse Width	$\overline{C}$, $\overline{G}$, $\overline{RES}$		PW_L	Fig. 3 (Asynchronous Mode)		$t_{cycE}+t_{SU}+t_{HD}$	–	$t_{cycE}+t_{SU}+t_{HD}$	–	$t_{cycE}+t_{SU}+t_{HD}$	–	ns
Input "High" Pulse Width	$\overline{C}$, $\overline{G}$		PW_H	Fig. 4 (Asynchronous Mode)		$t_{cycE}+t_{SU}+t_{HD}$	–	$t_{cycE}+t_{SU}+t_{HD}$	–	$t_{cycE}+t_{SU}+t_{HD}$	–	ns
Input Setup Time	$\overline{C}$, $\overline{G}$, $\overline{RES}$		t_{SU}	Fig. 5 (Synchronous Mode)		200	–	120	–	75	–	ns
	$\overline{C_3}$ (÷8 Prescaler Mode)					200	–	170	–	170	–	
Input Hold Time	$\overline{C}$, $\overline{G}$, $\overline{RES}$		t_{HD}	Fig. 5 (Synchronous Mode)		50	–	50	–	50	–	ns
	$\overline{C_3}$ (÷8 Prescaler Mode)					50	–	50	–	50	–	
Input Pulse Width	$\overline{C_3}$ (÷8 Prescaler Mode)		PW_L, PW_H	(Asynchronous Mode)		125	–	84	–	62.5	–	ns
Output Delay Time	$O_1 \sim O_3$	TTL	t_{co}	Fig. 6	V_{OH}=2.4V, Load B	–	700	–	460	–	340	ns
		MOS	t_{cm}		V_{OH}=2.4V, Load D	–	450	–	450	–	340	ns
		CMOS	t_{cmos}		V_{OH}=0.7×V_{CC}, Load D	–	2.0	–	1.35	–	1.0	μs
Interrupt Release Time			t_{IR}	Fig. 7		–	1.2	–	0.9	–	0.7	μs

* t_r, $t_f \leqq t_{cycE}$

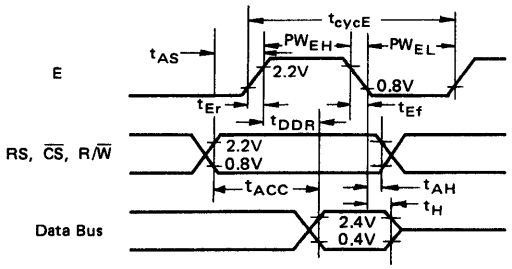


Figure 1 Bus Read Timing
(Read Information from PTM)

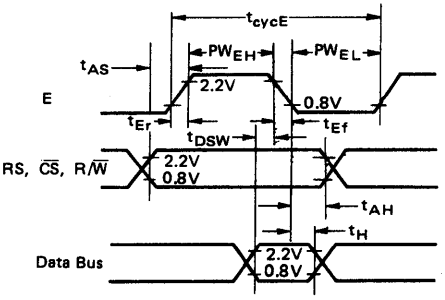


Figure 2 Bus Write Timing
(Write Information into PTM)

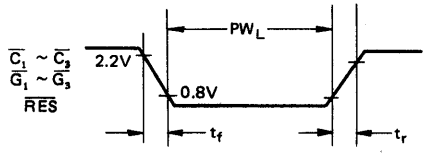


Figure 3 Input Pulse Width "Low"

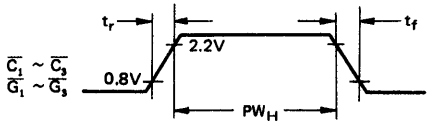


Figure 4 Input Pulse Width "High"

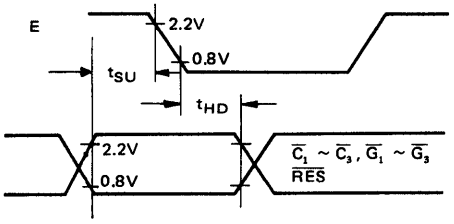


Figure 5 Input Setup and Hold Times

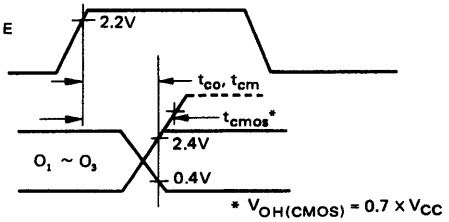


Figure 6 Output Delay

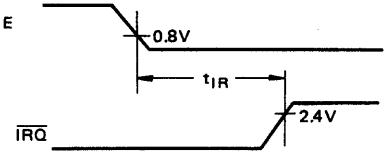


Figure 7 $\overline{IRQ}$ Release Time

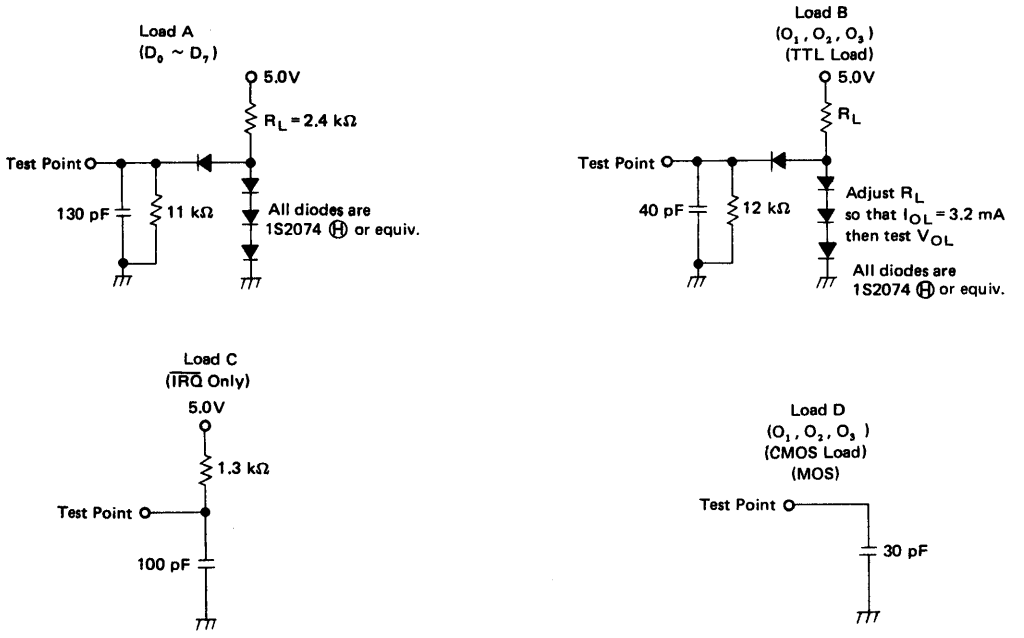


Figure 8 Test Loads

■ GENERAL DESCRIPTION

The HD6840 is part of the HMCS6800 microprocessor family and is fully bus compatible with HD6800 systems. The three timers in the HD6840 operate independently and in several distinct modes to fit a wide variety of measurement and synthesis applications.

The HD6840 is an integrated set of three distinct counter/timers. It consists of three 16-bit data latches, three 16-bit counters (clocked independently), and the comparison and enable circuitry necessary to implement various measurement and synthesis functions. In addition, it contains interrupt drivers to alert the processor that a particular function has been completed.

In a typical application, a timer will be loaded by first storing two bytes of data into an associated Counter Latch. This data is then transferred into the counter via a Counter initialization cycle. If the counter is enabled, the counter decrements on each subsequent clock period which may be an external clock, or Enable (E) until one of several predetermined conditions causes it to halt or recycle. The timers are thus programmable, cyclic in nature, controllable by external inputs or the MPU program, and accessible by the MPU at any time.

■ PTM INTERFACE SIGNALS FOR MPU

The Programmable Timer Module (PTM) interfaces to the HMCS6800 Bus with an eight-bit bidirectional data bus, two Chip Select lines, a Read/Write line, an Enable (System ϕ_2) line, an Interrupt Request line, an external Reset line, and three Register Select lines. These signals, in conjunction with the HD6800 VMA output, permit the MPU to control the PTM. VMA should be utilized in conjunction with an MPU address line into a Chip Select of the PTM, when the HD6800, HD6802 are used.

• Bidirectional Data (D_0 ~ D_7)

The bidirectional data lines (D_0~D_7) allow the transfer of data between the MPU and PTM. The data bus output drivers are three-state devices which remain in the high-impedance (off) state except when the MPU performs a PTM read operation (Read/Write and Enable lines "High" and PTM Chip Selects activated).

• Chip Select ($\overline{CS_0}$, CS_1)

These two signals are used to activate the Data Bus interface and allow transfer of data from the PTM. With $\overline{CS_0}$ = "Low" and CS_1 = "High", the device is selected and data transfer will occur.

• Read/Write (R/$\overline{W}$)

This signal is generated by the MPU to control the direction of data transfer on the Data Bus. With the PTM selected, a "Low" state on the PTM R/$\overline{W}$ line enables the input buffers and

data is transferred from the MPU to the PTM on the trailing edge of the Enable (System ϕ_2) signal. Alternately, (under the same conditions) R/$\overline{W}$ = "High" and Enable "High" allows data in the PTM to be read by the MPU.

- **Enable (E)**

This signal synchronizes data transfer between the MPU and the PTM. It also performs an equivalent synchronization function on the external clock, reset, and gate inputs of the PTM.

- **Interrupt Request ($\overline{IRQ}$)**

The active "Low" Interrupt Request signal is normally tied directly (or through priority interrupt circuitry) to the $\overline{IRQ}$ input of the MPU. This is an "open drain" output (no load device on the chip) which permits other similar interrupt request lines to be tied together in a wire-OR configuration.

The $\overline{IRQ}$ line is activated if, and only if, the Composite Interrupt Flag (Bit 7 of the Internal Status Register) is asserted. The conditions under which the $\overline{IRQ}$ line is activated are discussed in conjunction with the Status Register.

- **Reset ($\overline{RES}$)**

A "Low" level at this input is clocked into the PTM by the Enable (System ϕ_2) input. Two Enable pulses are required to synchronize and process the signal. The PTM then recognizes the active "Low" or inactive "High" on the third Enable pulse. If the $\overline{RES}$ signal is asynchronous, an additional Enable period is required if setup times are not met. The $\overline{RES}$ input must be stable "High"/"Low" for the minimum time stated in the AC Characteristics.

Recognition of a "Low" level at this input by the PTM causes the following action to occur:

a. All counter latches are preset to their maximal count values.
b. All Control Register bits are cleared with the exception of CR10 (internal reset bit) which is set.
c. All counters are preset to the contents of the latches.
d. All counter outputs are reset and all counter clocks are disabled.
e. All Status Register bits (interrupt flags) are cleared.

- **Register Select Lines (RS_0, RS_1, RS_2)**

These inputs are used in conjunction with the R/$\overline{W}$ line to select the internal registers, counters and latches as shown in Table 1.

It has been previously stated that the PTM is accessed via MPU Load and Store operations in much the same manner as a memory device. The instructions available with the HMCS6800 family of MPUs which perform operations directly on memory should not be used when the PTM is accessed. These instructions actually fetch a byte from memory, perform an operation, then restore it to the same address location. Since the PTM used the R/$\overline{W}$ line as an additional register select input, the modified data may not be restored to the same register if these instructions are used.

■ PTM ASYNCHRONOUS INPUT/OUTPUT SIGNALS

Each of the three timers within the PTM has external clock and gate inputs as well as a counter output line. The inputs are high impedance, TTL compatible lines and outputs are capable of driving two standard TTL loads.

- **Clock Inputs ($\overline{C_1}$, $\overline{C_2}$, $\overline{C_3}$)**

Input pins $\overline{C_1}$, $\overline{C_2}$, and $\overline{C_3}$ will accept asynchronous TTL voltage level signals to decrement Timers 1, 2, and 3, respectively. The "High" and "Low" levels of the external clocks must each be stable for at least one system clock period plus the sum

Table 1 Register Selection

Register Select Inputs *			Operations	
RS_2	RS_1	RS_0	R/$\overline{W}$ = "Low"	R/$\overline{W}$ = "High"
L	L	L	CR20 = "0" Write Control Register #3 CR20 = "1" Write Control Register #1	No Operation
L	L	H	Write Control Register #2	Read Status Register
L	H	L	Write MSB Buffer Register	Read Timer #1 Counter
L	H	H	Write Timer #1 Latches	Read LSB Buffer Register
H	L	L	Write MSB Buffer Register	Read Timer #2 Counter
H	L	H	Write Timer #2 Latches	Read LSB Buffer Register
H	H	L	Write MSB Buffer Register	Read Timer #3 Counter
H	H	H	Write Timer #3 Latches	Read LSB Buffer Register

* L; "Low" level, H; "High" level

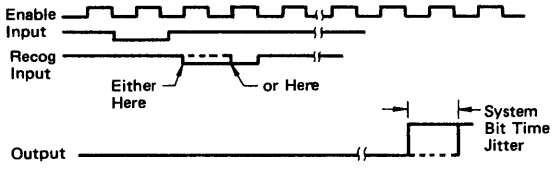
of the setup and hold times for the inputs. The asynchronous clock rate can vary from dc to the limit imposed by Enable (System ϕ_2) Setup, and Hold time.

The external clock inputs are clocked in by Enable (System ϕ_2) pulses. Three Enable periods are used to synchronize and process the external clock. The fourth Enable pulse decrements the internal counter. This does not affect the input frequency, it merely creates a delay between a clock input transition and internal recognition of that transition by the PTM. All references to $\overline{C}$ inputs in this document relate to internal recognition of the input transition. Note that a clock "High" or "Low" level which does not meet setup and hold time specifications may require an additional Enable pulse for recognition. When observing recurring events, a lack of synchronization will result in "jitter" being observed on the output of the PTM when using asynchronous clocks and gate input signals. There are two types of jitter. "System jitter" is the result of the input signals being out of synchronization with the Enable (System ϕ_2), permitting signals with marginal setup and hold time to be recognized by either the bit time nearest the input transition or the subsequent bit time.

"Input jitter" can be as great as the time between input signal negative going transitions plus the system jitter, if the first transition is recognized during one system cycle, and not recognized the next cycle, or vice versa.

External clock input $\overline{C_3}$ represents a special case when Timer

#3 is programmed to utilize its optional ÷8 prescaler mode. The maximum input frequency and allowable duty cycles for this case are specified under the AC Characteristics. The output of the ÷8 prescaler is treated in the same manner as the previously discussed clock inputs. That is, it is clocked into the counter by Enable pulses, is recognized on the fourth Enable pulse (provided setup and hold time requirements are met), and must produce an output pulse at least as wide as the sum of an Enable period, setup, and hold times.



- **$\overline{\text{Gate}}$ Inputs ($\overline{G_1}$, $\overline{G_2}$, $\overline{G_3}$)**

Input pins $\overline{G_1}$, $\overline{G_2}$, and $\overline{G_3}$ accept asynchronous TTL-compatible signals which are used as triggers or clock gating functions to Timers 1, 2, and 3, respectively. The gating inputs are clocked into the PTM by the Enable (System ϕ_2) signal in the same manner as the previously discussed clock inputs. That is, a $\overline{\text{Gate}}$ transition is recognized by the PTM on the fourth Enable pulse (provided setup and hold time requirements are met), and the "High" or "Low" levels of the $\overline{\text{Gate}}$ input must be stable for at least one system clock period plus the sum of setup and hold times. All references to $\overline{G}$ transition in this document relate to internal recognition of the input transition.

The $\overline{\text{Gate}}$ inputs of all timers directly affected the internal 16-bit counter. The operation of $\overline{G_3}$ is therefore independent of the ÷8 prescaler selection.

- **Timer Outputs (O_1, O_2, O_3)**

Timer outputs O_1, O_2, and O_3 are capable of driving up to two TTL loads and produce a defined output waveform for either Continuous or Single-Shot Timer modes. Output waveform definition is accomplished by selecting either Single 16-bit or Dual 8-bit operating modes. The single 16-bit mode will produce a square-wave output in the continuous timer mode and will produce a single pulse in the Single-Shot Timer mode. The Dual 8-bit mode will produce a variable duty cycle pulse in both the continuous and single shot Timer modes. "1" bit of each Control Register (CRX7) is used to enable the corresponding output. If this bit is cleared, the output will remain "Low" (V_{OL}) regardless of the operating mode.

If it is cleared while the output is high the output will go low during the first enable cycle following a write to the Control Register.

The Continuous and Single-Shot Timer Modes are the only ones for which output response is defined in this data sheet. Signals appear at the outputs (unless CRX7 = "0") during Frequency and Pulse Width comparison modes, but the actual waveform is not predictable in typical applications.

■ CONTROL REGISTER

Each timer in the HD6840 has a corresponding write-only Control Register. Control Register #2 has a unique address space (RS0="High", RS1="Low", RS2="Low") and therefore may be written into at any time. The remaining Control Registers (#1 and#3) share the Address Space selected by a "Low" level on all Register Select inputs.

- **CR20**

The least-significant bit of Control Register #2 (CR20) is used as an additional addressing bit for Control Registers #1 and #3. Thus, with all Register selects and R/$\overline{W}$ inputs at "Low" level. Control Register #1 will be written into if CR20 is a logic "1". Under the same conditions, control Register #3 can also be written into after a $\overline{\text{RES}}$ "Low" condition has occurred, since all control register bits (except CR10) are cleared. Therefore, one may write in the sequence CR3, CR2, CR1.

- **CR10**

The least-significant bit of Control Register #1 is used as an internal Reset bit. When this bit is a logic "0", all timers are allowed to operate in the modes prescribed by the remaining bits of the control registers. Writing a "1" into CR10 causes all counters to be preset with the contents of the corresponding counter latches, all counter clocks to be disabled, and the timer outputs and interrupt flags (Status Register) to be reset. Counter Latches and Control Registers are undisturbed by an Internal Reset and may be written into regardless of the state of CR10.

- **CR30**

The least-significant bit of Control Register #3 is used as a selector for a ÷8 prescaler which is available with Timer #3 only. The prescaler, if selected, is effectively placed between the clock input circuitry and the input to Counter #3. It can therefore be used with either the internal clock (Enable) or an external clock source.

- **CRX1 ~ CRX7 (X=1~3)**

The functions depicted in the foregoing discussions are tabulated in Table 2 for ease of reference.

Control Register Bits CR10, CR20, and CR30 are unique in that each selects a different function. The remaining bits (1 through 7) of each Control Register select common functions, with a particular Control Register affecting only its corresponding timer.

- **CRX1**

Bit 1 of Control Register #1 (CR11) selects whether an internal or external clock source is to be used with Timer #1. Similarly, CR21 selects the clock source for Timer #2, and CR31 performs this function for Timer #3. The function of each bit of Control Register "X" can therefore be defined as shown in the remaining section of Table 2.

- **CRX2**

Control Register Bit 2 selects whether the binary information contained in the Counter Latches (and subsequently loaded into the counter) is to be treated as a single 16-bit word or two 8-bit bytes. In the single 16-bit Counter Mode (CRX2=0) the counter will decrement to zero after N + 1 enabled ($\overline{G}$="Low") clock periods, where N is defined as the 16-bit number in the Counter Latches. With CRX2 = 1, a similar Time Out will occur after (L + 1)·(M + 1) enabled clock periods, where L and M, respectively, refer to the LSB and MSB bytes in the Counter Latches.

- **CRX3 ~ CRX7**

Control Register Bits 3, 4, and 5 are explained in detail in the Timer Operating Mode section. Bit 6 is an interrupt mask bit which will be explained more fully in conjunction with the Status Register, and bit 7 is used to enable the corresponding Timer Output. A summary of the control register programming modes is shown in Table 3.

■ STATUS REGISTER/INTERRUPT FLAGS

The HD6840 has an internal Read-Only Status Register which contains four Interrupt Flags. (The remaining four bits of the register are not used, and default to "0"s when being read.) Bits 0, 1, and 2 are assigned to Timers 1, 2, and 3, respectively, as individual flag bits, while Bit 7 is a Composite Interrupt Flag. This flag bit will be asserted if any of the individual flag bits is

Table 2 Control Register Bits

CONTROL REGISTER #1		CONTROL REGISTER #2		CONTROL REGISTER #3	
CR10	Internal Reset Bit	CR20	Control Register Address Bit	CR30	Timer #3 Clock Control
"0" All timers allowed to operate "1" All timers held in preset state		"0" CR #3 may be written "1" CR #1 may be written		"0" T3 Clock is not prescaled "1" T3 Clock is prescaled by ÷ 8	
CRX1*		Timer #X Clock Source			
"0"		TX uses external clock source on $\overline{CX}$ input			
"1"		TX uses Enable clock			
CRX2		Timer #X Counting Mode Control			
"0"		TX configured for normal (16-bit) counting mode			
"1"		TX configured for dual 8-bit counting mode			
CRX3 CRX4 CRX5		Timer #X Counter Mode and Interrupt Control (See Table 3)			
CRX6		Timer #X Interrupt Enable			
"0"		Interrupt Flag masked on $\overline{IRQ}$			
"1"		Interrupt Flag enabled to $\overline{IRQ}$			
CRX7		Timer #X Counter Output Enable			
"0"		TX Output masked on output OX			
"1"		TX Output enabled on output OX			

* Control Register for Timer 1, 2, or 3, Bit 1.

set while Bit 6 of the corresponding Control Register is at a logic "1". The conditions for asserting the Composite Interrupt Flag bit can therefore be expressed as:

$$INT = I_1 \cdot CR16 + I_2 \cdot CR26 + I_3 \cdot CR36$$

where INT = Composite Interrupt Flag (Bit 7)
I_1 = Timer #1 Interrupt Flag (Bit 0)
I_2 = Timer #2 Interrupt Flag (Bit 1)
I_3 = Timer #3 Interrupt Flag (Bit 2)

STATUS REGISTER

7	6	5	4	3	2	1	0
INT					I_3	I_2	I_1

An interrupt flag is cleared by a Timer Reset condition, i.e., External $\overline{RES}$ = "Low" or Internal Reset Bit (CR10) = "1". It will also be cleared by a Read Timer Counter Command provided that the Status Register has previously been read while the interrupt flag was set. This condition on the Read Status Register – Read Timer Counter (RS–RT) sequence is designed to prevent missing interrupts which might occur after the status register is read, but prior to reading the Timer Counter.

An Individual Interrupt Flag is also cleared by a Write Timer Latches (W) command or a Counter Initialization (CI) sequence, provided that W or CI affects the Timer corresponding to the individual Interrupt Flag.

■ COUNTER LATCH INITIALIZATION

Each of the three independent timers consists of a 16-bit addressable counter and 16 bits of addressable latches. The counters are preset to the binary numbers stored in the latches. Counter initialization results in the transfer of the latch contents to the counter. See notes in Table 5 regarding the binary number N, L, or M placed into the Latches and their relationship to the output waveforms and counter Time-Outs.

Since the PTM data bus is 8-bits wide and the counters are 16-bits wide, a temporary register (MSB Buffer Register) is provided. This "write only" register is for the Most Significant Byte of the desired latch data. Three addresses are provided for the MSB Buffer Register (as indicated in Table 1), but they all lead to the same Buffer. Data from the MSB Buffer will automatically be transferred into the Most Significant Byte of Timer #X when a Write Timer #X Latches Command is performed. So it can be seen that the HD6840 has been designed to allow transfer of two bytes of data into the counter latches provided that the MSB is transferred first.

In many applications, the source of the data will be as HMCS6800 MPU. It should be noted that the 16-bit store operations of the HMCS6800 microprocessors (STS and STX etc.) transfer data in the order required by the PTM. A Store Index Register Instruction, for example, results in the MSB of the X register being transferred to the selected address, then the LSB of the X register being written into the next higher location. Thus, either the index register or stack pointer may be transfered directly into a selected counter latch with a single instruction.

A logic "Low" at the $\overline{RES}$ input also initializes the counter latches. In this case, all latches will assume a maximum count of $(65{,}536)_{10}$. It is important to note that an Internal Reset (Bit 0 of Control Register 1 Set) has no effect on the counter latches.

■ COUNTER INITIALIZATION

Counter Initialization is defined as the transfer of data from the latches to the counter with subsequent clearing of the Individual Interrupt Flag associated with the counter. Counter Initialization always occurs when a reset condition ($\overline{RES}$ = "Low" or CR10 = "1") is recognized. It can also occur – depending on Timer Mode – with a Write Timer Latches command or recognition of a negative transition of the $\overline{Gate}$ input.

Counter recycling or re-initialization occurs when a negative transition of the clock input is recognized after the counter has reached an all-zero state. In this case, data is transferred from the Latches to the Counter.

■ TIMER OPERATING MODES

The HD6840 has been designed to operate effectively in a wide variety of applications. This is accomplished by using three bits of each control register (CRX3, CRX4, and CRX5) to

defined different operating modes of the Timers. These modes are divided into Wave Synthesis and Wave Measurement modes, and outlined in Table 4.

Table 4 Operating Modes

Control Register			Timer Operating Mode	
CRX3	CRX4	CRX5		
0	*	0	Continuous	Wave Synthesis
0	*	1	Single-Shot	
1	0	*	Frequency Comparison	Wave Measurement
1	1	*	Pulse Width Comparison	

* Defines Additional Timer Functions.

One of the WAVE SYNTHESIS modes is the Continuous Operating mode, which is useful for cyclic wave generation. Either symmetrical or variable duty-cycle waves can be generated in this mode. The other wave synthesis mode, the Single-Shot mode, is similar in use to the Continuous operating mode, however, a single pulse is generated, with a programmable preset width.

The WAVE MEASUREMENT modes include the Frequency Comparison and Pulse Width Comparison modes which are used to measure cyclic and singular pulse widths, respectively.

In addition to the four timer modes in Table 4, the remaining control register bit is used to modify counter initialization and enabling or interrupt conditions.

■ WAVE SYNTHESIS MODES

- **Continuous Operating Mode (Table 5)**

The continuous mode will synthesize a continuous wave with

Table 3 Control Register Programming

	Register 1	Register 2	Register 3
7 6 5 4 3 2 1 0			
X X X X X X X ‡	"0" All Timers Operate	Reg #3 May Be Written	T3 Clk ÷ 1
	"1" All Timers Preset	Reg #1 May Be Written	T3 Clk ÷ 8
7 6 5 4 3 2 1 0			
X X X X X X ‡ X	"0" External Clock ($\overline{CX}$ Input)		
	"1" Internal Clock (Enable)		
7 6 5 4 3 2 1 0			
X X X X X ‡ X X	"0" Normal (16-Bit) Count Mode		
	"1" Dual 8-Bit Count Mode		
7 6 5 4 3 2 1 0			
X X 0 0 0 X X X	Continuous Operating Mode: $\overline{Gate}$ ↓ or Write to Latches or Reset Causes Counter Initialization		
7 6 5 4 3 2 1 0			
X X 0 0 1 X X X	Frequency Comparison Mode: Interrupt if $\overline{Gate}$ ↓‾↑ is < Counter Time Out		
7 6 5 4 3 2 1 0			
X X 0 1 0 X X X	Continuous Operating Mode: $\overline{Gate}$ ↓ or Reset Causes Counter Initialization		
7 6 5 4 3 2 1 0			
X X 0 1 1 X X X	Pulse Width Comparison Mode: Interrupt if $\overline{Gate}$ ↓_↑ is < Counter Time Out		
7 6 5 4 3 2 1 0			
1 X 1 0 0 X X X	Single Shot Mode: $\overline{Gate}$ ↓ or Write to Latches or Reset Causes Counter Initialization		
7 6 5 4 3 2 1 0			
X X 1 0 1 X X X	Frequency Comparison Mode: Interrupt If $\overline{Gate}$ ↓‾↑ is > Counter Time Out		
7 6 5 4 3 2 1 0			
1 X 1 1 0 X X X	Single Shot Mode: $\overline{Gate}$ ↓ or Reset Causes Counter Initialization		
7 6 5 4 3 2 1 0			
X X 1 1 1 X X X	Pulse Width Comparison Mode: Interrupt If $\overline{Gate}$ ↓_↑ is > Counter Time Out		
7 6 5 4 3 2 1 0			
X ‡ X X X X X X	"0" Interrupt Flag Masked ($\overline{IRQ}$)		
	"1" Interrupt Flag Enabled ($\overline{IRQ}$)		
7 6 5 4 3 2 1 0			
‡ X X X X X X X	"0" Timer Output Masked		
	"1" Timer Output Enable		

(NOTE) Reset is Hardware or Software Reset ($\overline{RES}$ = "Low" or CR10 = "1").

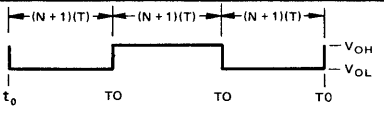
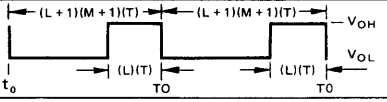
a period proportional to the preset number in the particular timer latches.

Any of the timers in the PTM may be programmed to operate in a continuous mode by writing "0"s into bits 3 and 5 of the corresponding control register. Assuming that the timer output is enabled (CRX7 = "1"), either a square wave or a variable duty cycle waveform will be generated at the Timer Output, OX. The type of output is selected via Control Register Bit 2.

Either a Timer Reset (CR10 = "1" or External $\overline{RES}$ = "Low") condition or internal recognition of a negative transition of the $\overline{Gate}$ input results in Counter Initialization. A Write Timer Latches command can be selected as a Counter Initialization signal by clearing CRX4.

The counter is enabled by an absence of a Timer Reset condition and a "Low" level at the $\overline{Gate}$ input. In the 16-bit mode, the counter will decrement on the first clock cycle during or after the counter initialization cycle. It continues to decrement on each clock signal so long as $\overline{G}$ remains "Low" and no reset condition exists. A Counter Time Out (the first clock after all

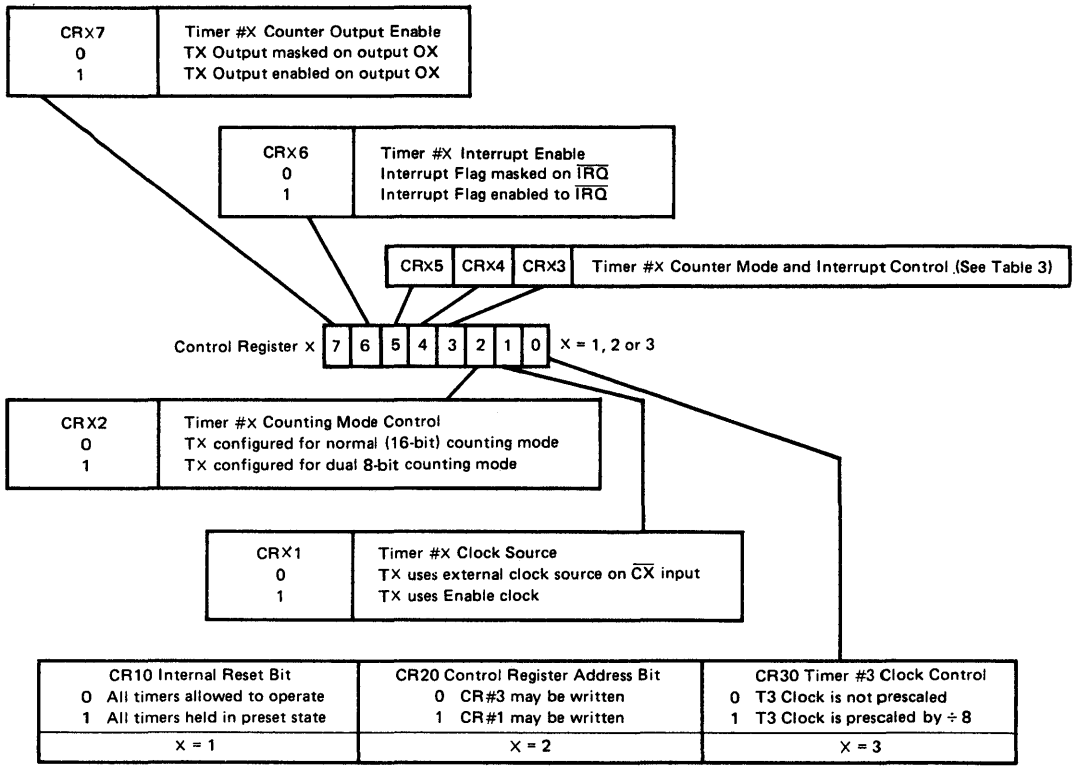
Table 5 Continuous Operating Modes

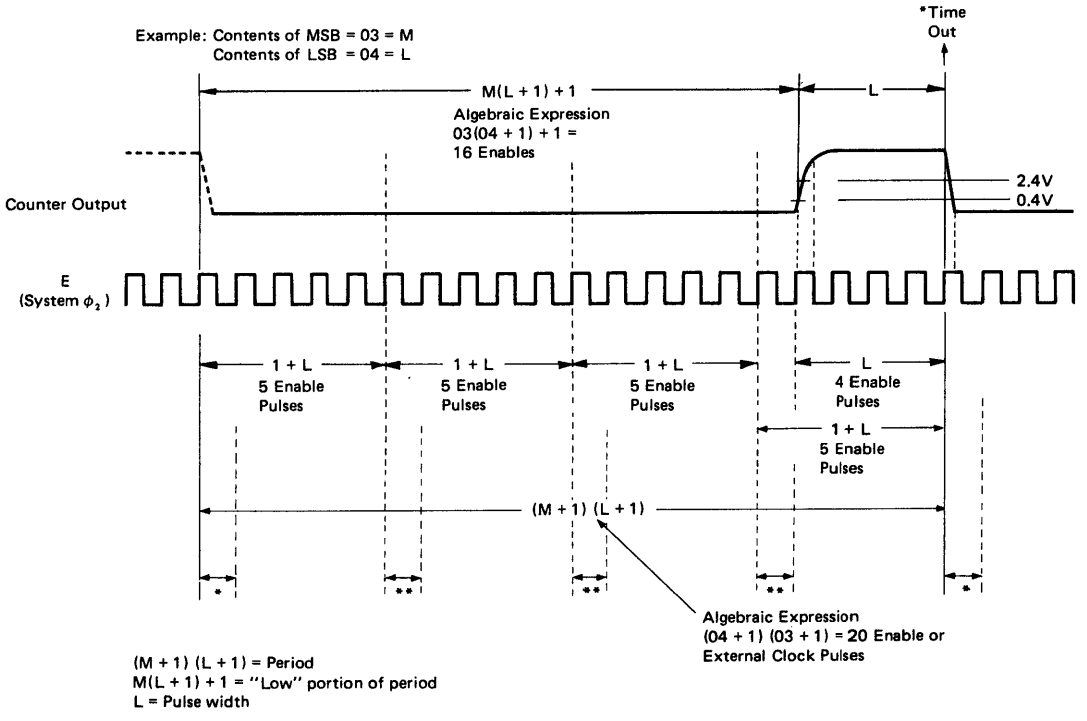
CONTINUOUS MODE (CRX3 = "0", CRX5 = "0")			
Control Register		Initialization/Output Waveforms	
CRX2	CRX4	Counter Initialization	*Timer Output (OX) (CRX7 = "1")
0	0	$\overline{G}\downarrow$+W+R	
0	1	$\overline{G}\downarrow$+R	
1	0	$\overline{G}\downarrow$+W+R	
1	1	$\overline{G}\downarrow$+R	

$\overline{G}\downarrow$ = Negative transition of $\overline{Gate}$ input.
W = Write Timer Latches Command.
R = Timer Reset (CR10 = "1" or External $\overline{RES}$ = "Low")
N = 16-Bit Number in Counter Latch.
L = 8-Bit Number in LSB Counter Latch.
M = 8-Bit Number in MSB Counter Latch.
T = Clock Input Negative Transitions to Counter.
t_0 = Counter Initialization Cycle.
TO = Counter Time Out (All Zero Condition).

* All time intervals shown above assume the $\overline{Gate}$ ($\overline{G}$) and $\overline{Clock}$ ($\overline{C}$) signals are synchronized to Enable (System ϕ_2) with the specified setup and hold time requirements.

Control Register Bits





* Preset LSB and MSB to Respective Latches on the negative transition of the E.
** Preset LSB to LSB Latches and Decrement MSB by one on the negative transition of the E.

Figure 9 Timer Output Waveform Example
(Continuous Dual 8-Bit Mode using Internal Enable)

counter bits = "0") results in the Individual Interrupt Flag being set and re-initialization of the counter.

In the dual 8-bit mode (CRX2= "1") [Refer to the example in Fig. 9] the MSB decrements once for every full countdown of the LSB + 1. When the LSB = "0", the MSB is unchanged; on the next clock pulse the LSB is reset to the count in the LSB Latches and the MSB is decremented by 1 (one). The output, if enabled, remains "Low" during and after initialization and will remain "Low" until the counter MSB is all "0"s. The output will go "High" at the beginning of the next clock pulse. The output remains "High" until both the LSB and MSB of the counter are all "0"s. At the beginning of the next clock pulse the defined Time Out (TO) will occur and the output will go "Low". In the Dual 8-bit mode the period of the output of the example in Fig. 9 would span 20 clock pulses as opposed to the 1546 clock pulses using the Normal 16-bit mode.

A special time-out condition exists for the dual 8-bit mode (CRX2 = "1") if L = "0". In this case, the counter will revert to a mode similar to the single 16-bit mode, except Time Out occurs after M+1 clock pulses. The output, if enabled, goes "Low" during the Counter Initialization cycle and reverses state at each Time Out. The counter remains cyclical (is re-initialized at each Time Out) and the Individual Interrupt Flag is set when Time Out occurs. If M = L = "0", the internal counters do not change, but the output toggles at a rate of 1/2 the clock frequency.

The discussion of the Continuous Mode has assumed that the application requires an output signal. It should be noted that the Timer operates in the same manner with the output disabled (CRX7 = "0"). A Read Timer Counter command is valid regardless of the state of CRX7.

- **Single-Shot Timer Mode**

This mode is identical to the Continuous Mode with three exceptions. The first of these is obvious from the name – the output returns to a "Low" level after the initial Time Out and remains "Low" until another Counter Initialization cycle occurs. The waveforms available are shown in Table 6.

As indicated in Table 6, the internal counting mechanism remains cyclical in the Single-Shot Mode. Each Time Out of the counter results in the setting of an Individual Interrupt Flag and re-initialization of the counter.

The second major difference between the Single-Shot and Continuous modes is that the internal counter enable is not dependent on the $\overline{\text{Gate}}$ input level remaining in the "Low" state for the Single-Shot mode.

Another special condition is introduced in the Single-Shot mode. If L = M = "0"(Dual 8-bit) or N = "0" (Single 16-bit), the output goes "Low" on the first clock received during or after Counter Initialization. The output remains "Low" until the Operating Mode is changed or nonzero data is written into the Counter Latches. Time Outs continue to occur at the end of each clock period.

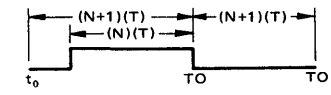
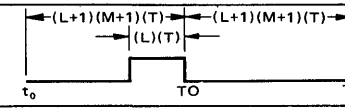
The three differences between Single-Shot and Continuous Timer Modes can be summarized as attributes of the Single-Shot

mode:

1. Output is enabled for only one pulse until it is reinitialized.
2. Counter Enable is independent of $\overline{Gate}$.
3. L = M = "0" or N = "0" disables output.

Aside from these differences, the two modes are identical.

Table 6 Single-Shot Operating Modes

Single-Shot Mode (CRX3 = "0", CRX7 = "1", CRX5 = "1")			
Control Register		Initialization/Output Waveforms	
CRX2	CRX4	Counter Initialization	Timer Output (OX)
0	0	$\overline{G}\downarrow$+W+R	
0	1	$\overline{G}\downarrow$+R	
1	0	$\overline{G}\downarrow$+W+R	
1	1	$\overline{G}\downarrow$+R	

Symbols are as defined in Table 5.

■ Wave Measurement Modes

The Wave Measurement Modes are the Frequency (period) Measurement and Pulse Width Comparison Modes, and are provided for those applications which require more flexibility of interrupt generation and Counter Initialization. Individual Interrupt Flags are set in these modes as a function of both Counter Time Out and transitions of the $\overline{Gate}$ input. Counter Initialization is also affected by Interrupt Flag status.

A timer's output is normally not used in a Wave Measurement mode, but it is defined. If the output is enabled, it will operate as follows. During the period between reinitialization of the timer and the first Time Out, the output will be a logical zero. If the first Time Out is completed (regardless of its method of generation), the output will go "High". If further TO's occur, the output will change state at each completion of a Time-Out.

The counter does operate in either Single 16-bit or Dual 8-bit modes as programmed by CRX2. Other features of the Wave Measurement Modes are outlined in Table 7.

Table 7 Wave Measurement Modes

CRX3 = "1"			
CRX4	CRX5	Application	Condition for Setting Individual Interrupt Flag
0	0	Frequency Comparison	Interrupt Generated if $\overline{Gate}$ Input Period (1/F) is less than Counter Time Out (TO)
0	1	Frequency Comparison	Interrupt Generated if $\overline{Gate}$ Input Period (1/F) is greater than Counter Time Out (TO)
1	0	Pulse Width Comparison	Interrupt Generated if $\overline{Gate}$ Input "Down Time" is less than Counter Time Out (TO)
1	1	Pulse Width Comparison	Interrupt Generated if $\overline{Gate}$ Input "Down Time" is greater than Counter Time Out (TO)

• Frequency Comparison or Period Measurement Mode (CRX3 = "1", CRX4 = "0")

The Frequency Comparison Mode with CRX5 = "1" is straightforward. If Time Out occurs prior to the first negative transition of the $\overline{Gate}$ input after a Counter Initialization cycle, an Individual Interrupt Flag is set. The counter is disabled, and a Counter Initialization cycle cannot begin until the interrupt flag is cleared and a negative transition on $\overline{G}$ is detected.

If CRX5 = "0", as shown in Table 7 and Table 8, an interrupt is generated if $\overline{Gate}$ input returns "Low" prior to a Time Out. If Counter Time-Out occurs first, the counter is recycled and continues to decrement. A bit is set within the timer on the initial Time Out which precludes further individual interrupt generation until a new Counter Initialization cycle has been completed. When this internal bit is set, a negative transition of the $\overline{Gate}$ input starts a new Counter Initialization cycle. (The condition of $\overline{G}\downarrow\cdot\overline{I}\cdot TO$ is satisfied, since a Time Out has occurred and no individual Interrupt has been generated.)

Any of the timers within the PTM may be programmed to compare the period of a pulse (giving the frequency after calculations) at the Gate input with the time period requested for Counter Time-Out. A negative transition of the $\overline{Gate}$ input enables the counter and starts a Counter Initialization cycle – provided that other conditions as noted in Table 8 are satisfied. The counter decrements on each clock signal recognized during or after Counter Initialization until an Interrupt is generated, a Write Timer Latches command is issued, or a Timer Reset condition occurs. It can be seen from Table 8 that an interrupt condition will be generated if CRX5 = "0" and the period of the pulse (single pulse or measured separately repetitive pulses) at the $\overline{Gate}$ input is less than the Counter Time Out period. If CRX5 = "1", an interrupt is generated if the reverse is true.

Assume now with CRX5 = "1" that a Counter Initialization has occurred and that the $\overline{\text{Gate}}$ input has returned "Low" prior to Counter Time Out. Since there is no Individual Interrupt Flag generated, this automatically starts a new Counter Initialization Cycle. The process will continue with frequency comparison being performed on each $\overline{\text{Gate}}$ input cycle until the mode is changed, or a cycle is determined to be above the predetermined limit.

- **Pulse Width Comparison Mode (CRX3 = "1", CRX4 = "1")**

This mode is similar to the Frequency Comparison Mode except for a positive, rather than negative, transition of the $\overline{\text{Gate}}$ input terminates the count. With CRX5 = "0", an Individual Interrupt Flag will be generated if the "Low" level pulse applied to the $\overline{\text{Gate}}$ input is less than the time period required for Counter Time Out. With CRX5 = "1", the interrupt is generated when the reverse condition is true.

As can be seen in Table 9, a positive transition of the $\overline{\text{Gate}}$ input disables the counter. With CRX5 = "0", it is therefore possible to directly obtain the width of any pulse causing an interrupt. Similar data for other Time Interval Modes and conditions can be obtained, if two sections of the PTM are dedicated to the purpose.

Table 8 Frequency Comparison Mode

CRX3 = "1", CRX4 = "0"				
Control Reg Bit 5 (CRX5)	**Counter Initialization**	**Counter Enable Flip-Flop Set (CE)**	**Counter Enable Flip-Flop Reset (CE)**	**Interrupt Flag Set (I)**
0	$\overline{G}\downarrow\cdot\overline{I}\cdot(\overline{CE}+TO)+R$	$\overline{G}\downarrow\cdot\overline{W}\cdot\overline{R}\cdot\overline{I}$	W+R+I	$\overline{G}\downarrow$ Before TO
1	$\overline{G}\downarrow\cdot\overline{I}+R$	$\overline{G}\downarrow\cdot\overline{W}\cdot\overline{R}\cdot\overline{I}$	W+R+I	TO Before $\overline{G}\downarrow$

I represents the interrupt for a given timer.

Table 9 Pulse Width Comparison Mode

CRX3 = "1", CRX4 = "1"				
Control Reg Bit 5 (CRX5)	**Counter Initialization**	**Counter Enable Flip-Flop Set (CE)**	**Counter Enable Flip-Flop Reset (CE)**	**Interrupt Flag Set (I)**
0	$\overline{G}\downarrow\cdot\overline{I}+R$	$\overline{G}\downarrow\cdot\overline{W}\cdot\overline{R}\cdot\overline{I}$	W+R+I+G	$\overline{G}\uparrow$ Before TO
1	$\overline{G}\downarrow\cdot\overline{I}+R$	$\overline{G}\downarrow\cdot\overline{W}\cdot\overline{R}\cdot\overline{I}$	W+R+I+G	TO Before $\overline{G}\uparrow$

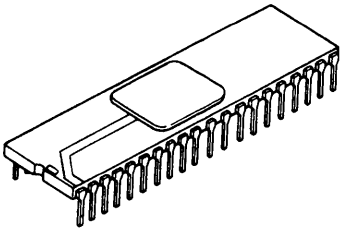
G = Level sensitive recognition of $\overline{\text{Gate}}$ input.

HD6843, HD68A43
FDC (Floppy Disk Controller)

The HD6843 Floppy Disk Controller performs the complex MPU/Floppy interface function. The FDC was designed to optimize the balance between the "Hardware/Software" in order to achieve integration of all key functions and maintain flexibility.

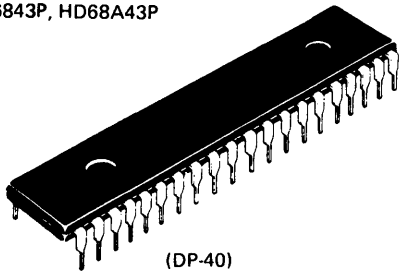
The FDC can interface a wide range of drives with a minimum of external hardware. Multiple drives can be controlled with the addition of external multiplexing rather than additional FDC's.

HD6843, HD68A43



(DC-40)

HD6843P, HD68A43P

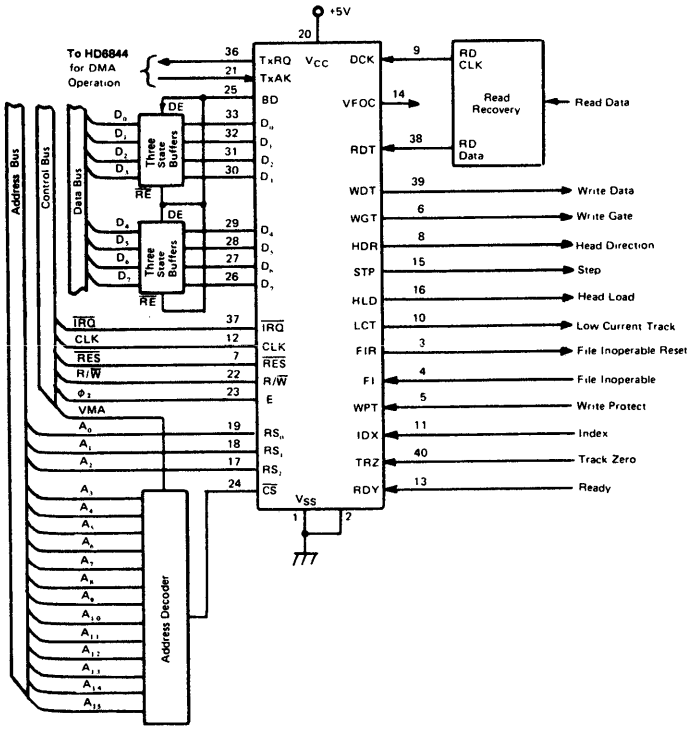


(DP-40)

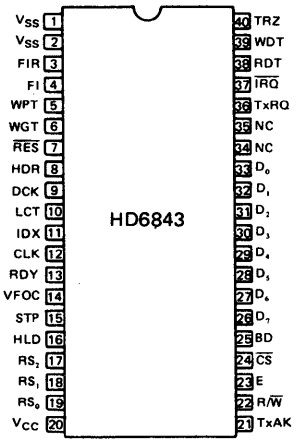
■ FEATURES

- Format compatible with IBM3740
- User Programmable read/write format
- Ten powerful macro-commands
- Macro End Interrupt allows parallel processing of MPU and FDC
- Controls multiple Floppies with external multiplexing
- Direct interface with HMCS6800
- Programmable seek and settling times enable operation with a wide range of Floppy drives
- Offers both Programmed Controlled I/O (PCIO) and DMA data transfer mode
- Free-Format read or write
- Single 5-volt power supply
- All registers directly accessible
- Compatible with MC6843

■ BLOCK DIAGRAM



■ PIN ARRANGEMENT



(Top View)

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC}*	-0.3 ~ +7.0	V
Input Voltage	V_{in}*	-0.3 ~ +7.0	V
Operating Temperature	T_{opr}	-20 ~ +75	°C
Storage Temperature	T_{stg}	-55 ~ +150	°C

* With respect to V_{SS} (SYSTEM GND)

(NOTE) Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ RECOMMENDED OPERATING CONDITIONS

Item	Symbol	min.	typ.	max.	Unit
Supply Voltage	V_{CC}*	4.75	5.0	5.25	V
Input "High" Voltage	V_{IH}*	2.0	–	V_{CC}	V
Input "Low" Voltage	V_{IL}*	-0.3	–	0.8	V
Operating Temperature	T_{opr}	-20	25	75	°C

* With respect to V_{SS} (SYSTEM GND)

■ ELECTRICAL CHARACTERISTICS

• DC CHARACTERISTICS (V_{CC}=5V±5%, V_{SS}=0V, Ta=-20~+75°C, unless otherwise noted.)

Item		Symbol	Test Condition	min.	typ.*	max.	Unit
Input "High" Voltage		V_{IH}		2.0	–	V_{CC}	V
Input "Low" Voltage		V_{IL}		-0.3	–	0.8	V
Input Leakage Current		I_{in}	V_{in}=0~5.25V	–	1.0	2.5	μA
Output "High" Voltage		V_{OH}	I_{OH}=-205μA (D_0~D_7) I_{OH}=-100μA (Others)	2.4	–	–	V
Output "Low" Voltage		V_{OL}	I_{OL}=3.2mA ($\overline{IRQ}$) I_{OL}=1.6mA (Others)	–	–	0.4	V
Three-state (off-state) Leakage Current		I_{TSI}	V_{in}=0.4~2.4V	–	2.0	10	μA
Output Leakage (off-state) Current ($\overline{IRQ}$)		I_{LOH}	V_{OH}=2.4V	–	1.0	10	μA
Power Dissipation		P_D		–	600	1000	mW
Input Capacitance	D_0~D_7	C_{in}	V_{in}=0V, T_a=25°C, f=1 MHz	–	–	12.5	pF
	Other inputs			–	–	10	pF
Output Capacitance		C_{out}	V_{in}=0V, T_a=25°C, f=1 MHz	–	–	10	pF

* V_{CC} = 5V, Ta = 25°C

- **AC CHARACTERISTICS (V_{CC}=5V±5%, V_{SS}=0V, Ta=-20~+75°C, unless otherwise noted.)**

Item	Symbol	Test Condition	HD6843			HD68A43			Unit
			min.	typ.	max.	min.	typ.	max.	
CLK Cycle Time	t_{cycC}	Figure 1	–	1.0	–	–	1.0	–	μs
CLK Pulse Width, "High"	PW_{HC}	Figure 1	0.4	–	–	0.4	–	–	μs
CLK Pulse Width, "Low"	PW_{LC}	Figure 1	0.35	–	–	0.35	–	–	μs
Rise and Fall Time of CLK	t_{Cr}, t_{Cf}	Figure 1	–	–	25	–	–	25	ns
DCK Cycle Time	t_{cycD}	Figure 2	2.6	4.0	–	2.6	4.0	–	μs
DCK Pulse Width, "High"	PW_{HD}	Figure 2	1.3	1.95	–	1.3	1.95	–	μs
DCK Pulse Width, "Low"	PW_{LD}	Figure 2	1.3	1.95	–	1.3	1.95	–	μs
Rise and Fall Time of DCK	t_{Dr}, t_{Df}	Figure 2	–	–	25	–	–	25	ns
RDT Width, "High"	t_{RDH}	Figure 2	1.0	–	–	1.0	–	–	μs
RDT Width, "Low"	t_{RDL}	Figure 2	1.0	–	–	1.0	–	–	μs
RDT~DCK Delay Time 1	t_{RDD1}	Figure 2	0.15	–	1.70	0.15	–	1.70	μs
RDT~DCK Delay Time 2	t_{RDD2}	Figure 2	0.15	–	1.70	0.15	–	1.70	μs
IDX Pulse Width, "High"	PW_{IDX}	Figure 3	20.0	–	–	20.0	–	–	μs
FIR Delay Time	t_{FIRD}	Figure 4	–	–	450	–	–	450	ns
FIR Pulse Width, "High"	PW_{FIR}	Figure 4	200	–	–	200	–	–	ns
WDT Pulse Width, "High"	PW_{WD}	Figure 7	–	1.0	–	–	1.0	–	μs
WDT Cycle Time	t_{cycW}	Figure 7	–	2.0	–	–	2.0	–	μs
STP Pulse Width, "High"	PW_{STP}	Figure 5	–	32	–	–	32	–	μs
STP Cycle Time	t_{cycS} *	Figure 5	1	–	15	1	–	15	ms
HLD Delay Time (HLD~STP)	t_{HLDD}	Figure 5	1	–	15	1	–	15	μs
HDR Set Up Time	t_{HDRS}	Figure 5	0	–	–	0	–	–	ns
HDR Hold Time	t_{HDRH}	Figure 5	32	–	–	32	–	–	μs
TxAK Set Up Time	t_{AS3}	Figure 10, 11	140	–	–	140	–	–	ns
TxAK Hold Time	t_{AH3}	Figure 10, 11	10	–	–	10	–	–	ns
TxRQ Release Time	t_{TR}	Figure 10, 11	–	–	450	–	–	240	ns
$\overline{IRQ}$ Release Time	t_{IR}	Figure 6	–	–	1.2	–	–	1.2	μs

* Cycle Time of STP changes according to the program.

- **BUS TIMING CHARACTERISTICS (V_{CC}=5V±5%, V_{SS}=0V, Ta=-20~+75°C, unless otherwise noted.)**

1 READ OPERATION SEQUENCE

Item	Symbol	Test Condition	HD6843			HD68A43			Unit
			min.	typ.	max.	min.	typ.	max.	
Enable Cycle Time	t_{cycE}	Figure 8, 10	1.0	–	–	0.666	–	–	μs
Enable Pulse Width, "High"	PW_{EH}	Figure 8, 10	0.4	–	–	0.23	–	–	μs
Enable Pulse Width, "Low"	PW_{EL}	Figure 8, 10	0.4	–	–	0.23	–	–	μs
Rise and Fall Time of Enable Input	t_{Er}, t_{Ef}	Figure 8, 10	–	–	25	–	–	25	ns
Address Set Up Time	t_{AS}	Figure 8, 10	140	–	–	140	–	–	ns
Data Delay Time	t_{DDR}	Figure 8, 10	–	–	225	–	–	200	ns
Data Access Time	t_{ACC}	Figure 8, 10	–	–	365	–	–	340	ns
Data Hold Time	t_{H}	Figure 8, 10	10	–	–	10	–	–	ns
Address Hold Time	t_{AH}	Figure 8, 10	10	–	–	10	–	–	ns
Bus Direction Delay Time	t_{DBD}	Figure 8, 10	–	–	400	–	–	400	ns

2 WRITE OPERATION SEQUENCE

Item	Symbol	Test Condition	HD6843			HD68A43			Unit
			min.	typ.	max.	min.	typ.	max.	
Enable Cycle Time	t_{cycE}	Figure 9, 11	1.0	–	–	0.666	–	–	μs
Enable Pulse Width, "High"	PW_{EH}	Figure 9, 11	0.4	–	–	0.23	–	–	μs
Enable Pulse Width, "Low"	PW_{EL}	Figure 9, 11	0.4	–	–	0.23	–	–	μs
Rise and Fall Time of Enable Input	t_{Er}, t_{Ef}	Figure 9, 11	–	–	25	–	–	25	ns
Address Set Up Time	t_{AS}	Figure 9, 11	140	–	–	140	–	–	ns
Data Set Up Time	t_{DSW}	Figure 9, 11	100	–	–	60	–	–	ns
Data Hold Time	t_H	Figure 9, 11	10	–	–	10	–	–	ns
Address Hold Time	t_{AH}	Figure 9, 11	10	–	–	10	–	–	ns
Bus Direction Delay Time	t_{DBD}	Figure 9, 11	–	–	400	–	–	400	ns

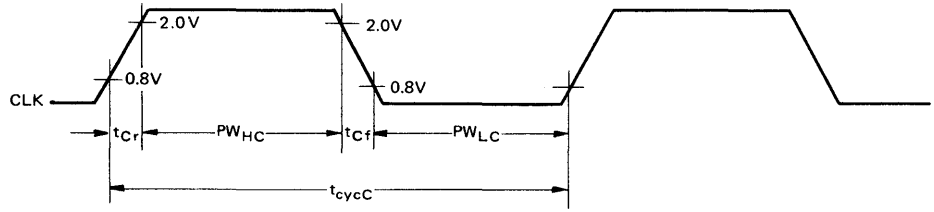


Figure 1 CLK Waveform

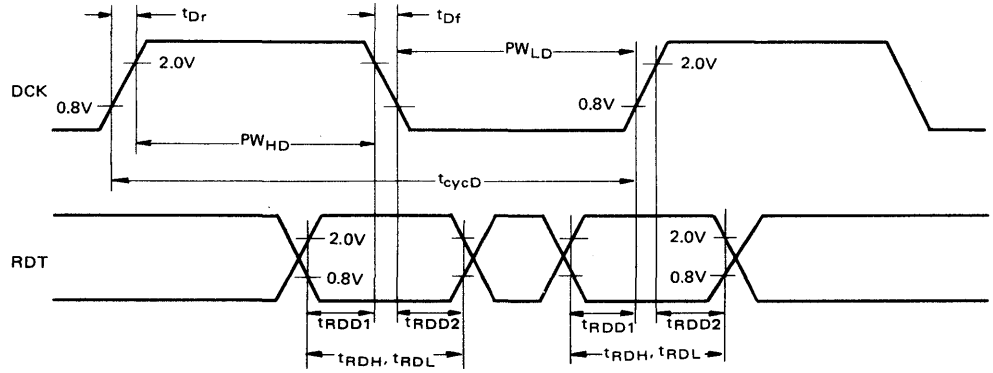


Figure 2 DCK, RDT Timing

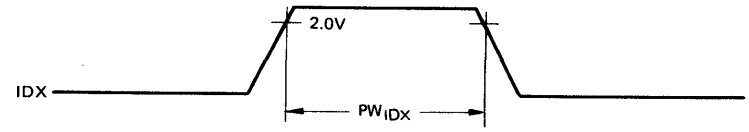


Figure 3 IDX Waveform

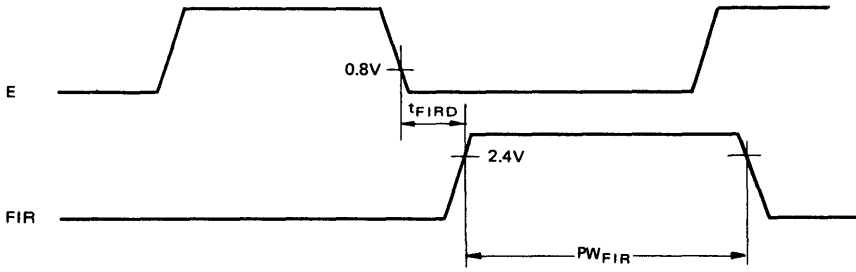


Figure 4 FIR Timing

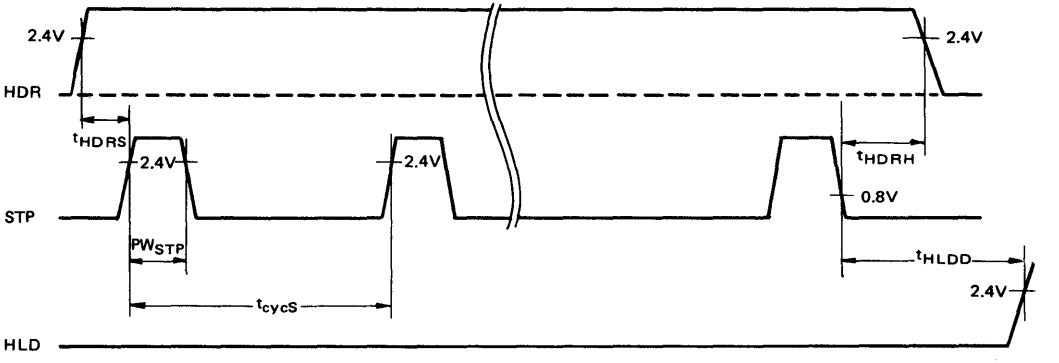


Figure 5 Seek Operation Sequence

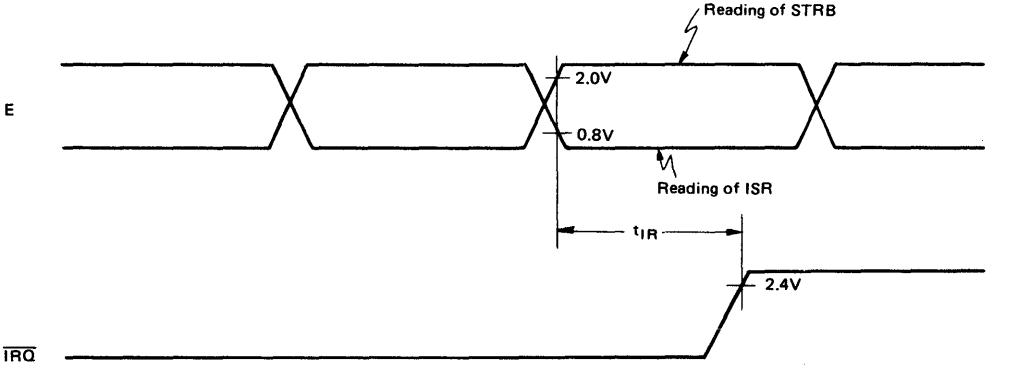


Figure 6 $\overline{IRQ}$ Release Timing

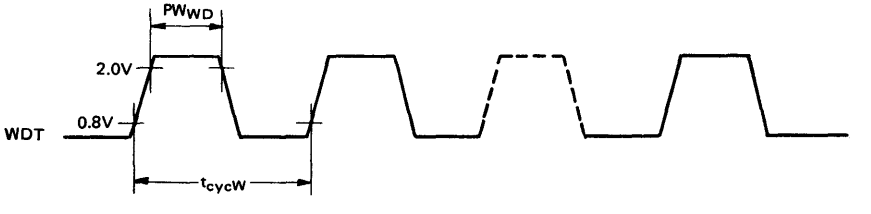


Figure 7 WDT Waveform

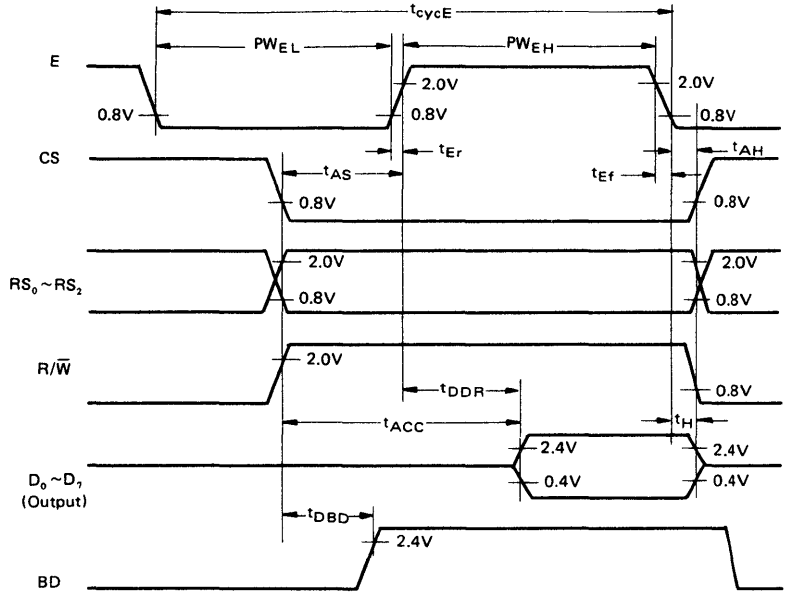


Figure 8 Read Timing

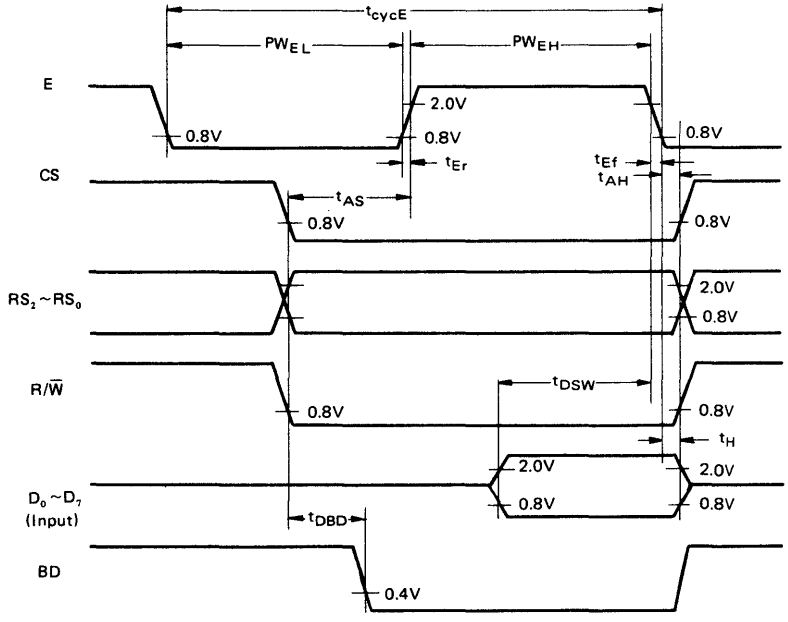


Figure 9 Write Timing

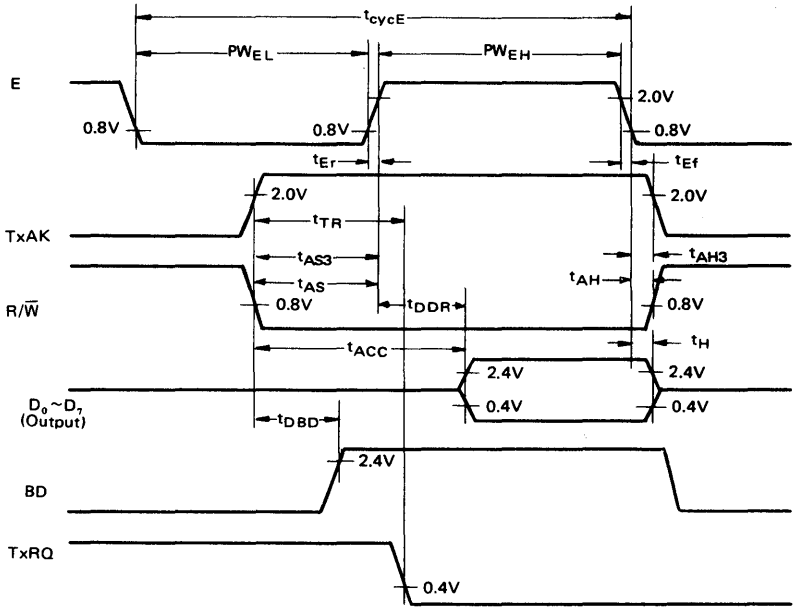


Figure 10 DMA Read Timing

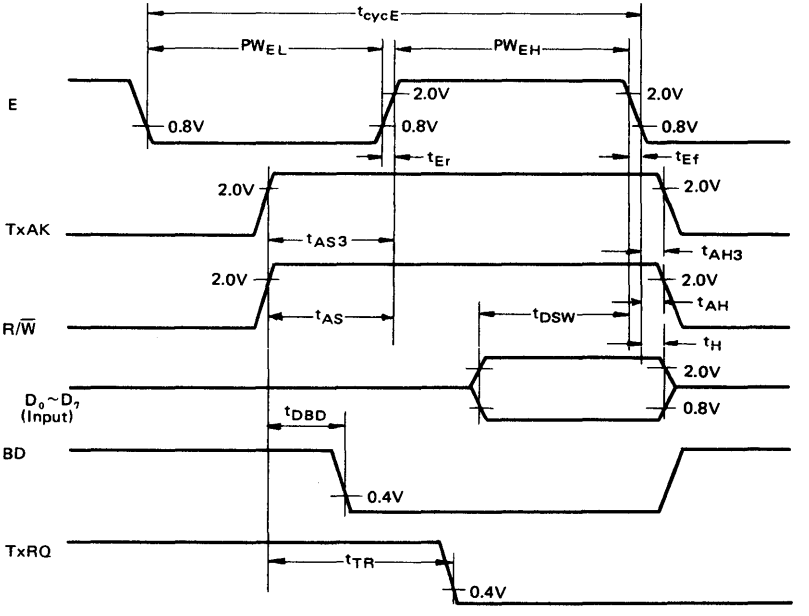


Figure 11 DMA Write Timing

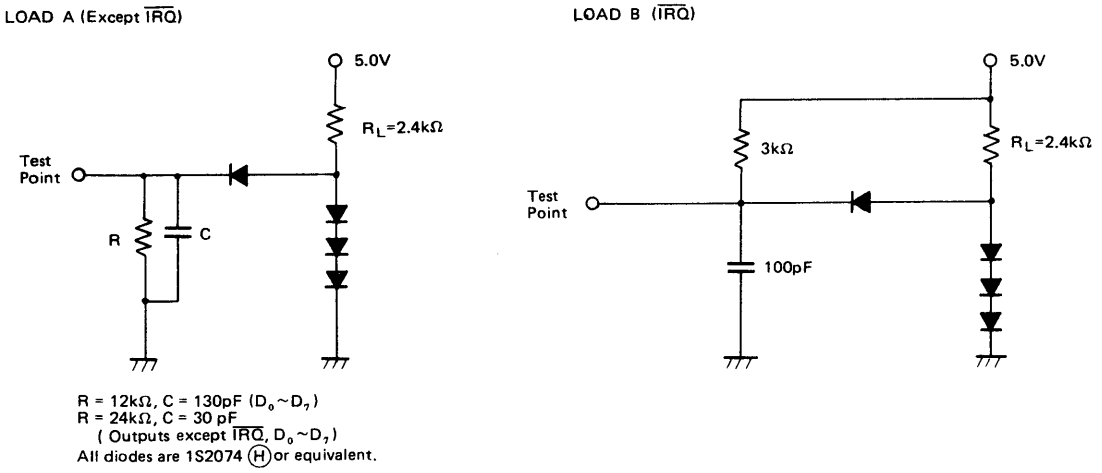


Figure 12 Load Circuit

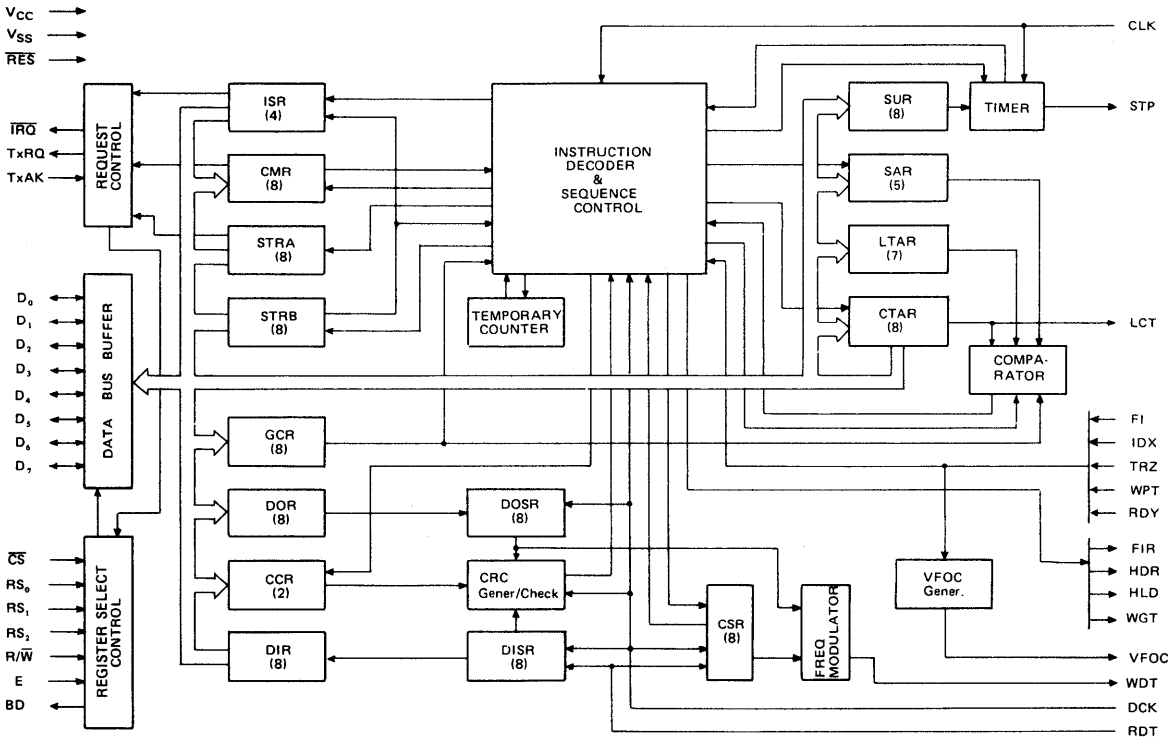


Figure 13 Block Diagram of the FDC

■ GENERAL DESCRIPTION

The HD6843 FDC consists of four primary sections; the Register, Serializing, Bus Interface, and Control sections. The following explanation of these sections can be followed in the block diagram of Figure 13.

• Register Section

The register section consists of twelve user accessible registers used for controlling a floppy disk drive. All twelve are connected by the internal data bus to allow the processor access to them.

Data Output Register (DOR)

The DOR is an 8-bit register which holds the data to be written onto the disk. The information is stored here by the bus interface.

Data Input Register (DIR)

The data words read from the disk are stored in the 8-bit DIR until read by the bus interface.

Current Track Address Register (CTAR)

CTAR is a 8-bit register containing the address of the track over which the R/W head is currently positioned.

Command Register (CMR)

The macro commands are written to the 8-bit CMR to begin their execution.

Interrupt Status Register (ISR)

The four bits of the ISR represent the four conditions that can cause an interrupt to occur.

Set-Up Register (SUR)

Variable Seek and Settling times are programmed by the SUR. Four bits are used to program the track to track seek time and four bits are used to program the head settling time for the floppy disk drive used with the FDC.

Status Register A (STRA)

The eight bits of STRA are used to indicate the state of the floppy disk interface.

Sector Address Register (SAR)

SAR contains the five bit sector address associated with the current data transfer.

Status Register B (STRB)

The eight error flags of STRB are used to signify error conditions detected by the FDC or generated by the floppy disk drive.

General Count Register (GCR)

The seven bits of GCR contain the destination track address when a SEK (seek) macro command is being executed. If a multi-sector Read or Write macro command is being executed, GCR contains the number of sectors to be read or written.

CRC Control Register (CCR)

The two bits of the CCR are used to enable the CRC and shift the CRC for the Free Format Commands.

Logical Track Address Register (LTAR)

The seven bit track address used for read and write operations is stored in the LTAR by the bus interface.

• Serializing Section

The serializing section handles the serial-to-parallel and parallel-to-serial conversions for Read/Write operations as well as CRC generation/checking and the generation/detection of the clock pattern. The Data Output Shift Register (DOSR), Data Input Shift Register (DISR), CRC Generator/Checker, and Clock Shift Register (CSR) comprise the serializing section of the FDC.

• Bus Interface

The Bus Interface section provides the timing and control logic that allows the FDC to operate with the 6800 bus, and is comprised of the Data Buffers. Request Control, and the Register Select circuitry.

• Control

The internal timing and control signals which sequence the FDC are derived from the macro instructions by the control section.

■ HD6843 PIN DESCRIPTION

• Power Pins

V_{CC}: +5 volt (±5%) power input.
V_{SS}: Power Supply Ground.

• Bus Pins

Reset ($\overline{RES}$) Input

The $\overline{RES}$ input is used to initialize the FDC. When $\overline{RES}$ becomes "Low", the state of the outputs is defined by the table below:

Output	State of Output	Output	State of Output
FIR	"Low"	HLD	"Low"
WGT	"Low"	TxRQ	"Low"
HDR	"Low"	$\overline{IRQ}$	"High"
STP	"Low"	WDT	"Low"

Registers which are affected by $\overline{RES}$ are shown in Table 7.

Interrupt Request ($\overline{IRQ}$) Output

The $\overline{IRQ}$ line is an open drain output that becomes a "Low" level (logic "0") when the FDC requests an interrupt. Interrupt requests are controlled by the interrupt enables in CMR (Command Register) with the function causing the interrupt shown in ISR (Interrupt Status Register).

Data Bus 0~Data Bus 7 (D_0~D_7) Bidirectional

The 8 bidirectional data lines allow the transfer of data between the FDC and the controlling system. The output buffers are three-state drivers that are enabled when the FDC is transferring data to the data bus.

Enable (E) Input

The E input to the FDC causes data transfers to occur between the FDC and the system controlling the FDC

(HMCS6800 MPU, DMA Controller, etc.) E must be a logic "1" ("High" level) for any transfer to be enabled on $D_0 \sim D_7$. The E input is normally connected to system ϕ_2.

Chip Select ($\overline{CS}$) Input

The $\overline{CS}$ input in conjunction with the E input, is used to enable data transfers on $D_0 \sim D_7$. E must be a "High" level and $\overline{CS}$ must be a "Low" level (logic "0") to enable the transfer. The TxAK input being a "High" level (logic "1") performs a similar function as $\overline{CS}$ being a "Low" level.

Read/Write (R/$\overline{W}$) Input

The R/$\overline{W}$ input is issued by the system controlling the FDC (HMCS6800 MPU, DMA Controller, etc.) to signify if a read or write operation is to be performed on the FDC. When TxAK is a "Low" level, R/$\overline{W}$ is used in conjunction with $\overline{CS}$ and $RS_0 \sim RS_2$ to determine which register is accessed by the bus as shown in Table 1. When TxAK is a "High" level, R/$\overline{W}$ is used to select either the DOR or DIR to the data bus (see description of TxAK input).

Register Select 0~Register Select 2 ($RS_0 \sim RS_2$) Input

$RS_0 \sim RS_2$, in conjunction with the R/$\overline{W}$ input, are used to select one of the user accessible registers in the FDC as shown in Table 1.

Transfer Request (TxRQ) Output

TxRQ is used in the DMA mode to request a data transfer from the DMAC. TxRQ is a "High" level if the FDC is in the DMA mode (CMR bit 5 is set) when a data transfer request occurs (STRA bit 0 is set). It is reset to a "Low" level (logic "0") when TxAK becomes a "High" level (logic "1"). Data transfer errors will occur if TxAK does not reset TxRQ before the next data transfer is required.

Transfer Acknowledge (TxAK) Input

TxAK is generated by the system controlling the FDC (HMCS6800 MPU, DMA Controller, etc.) and is a response to a TxRQ issued by the FDC. A "High" level (logic "1") on TxAK causes the FDC to neglect the state of $RS_0 \sim RS_2$ causing the FDC to select the DOR (Data Output Register) or DIR (Data Input Register) to the data bus ($D_0 \sim D_7$) as shown in Table 2. $\overline{CS}$ = "0" and TxAK = "1" cannot be permitted at the same time.

Table 2 Register Selection for DMA Transfers

TxAK	$RS_0 \sim RS_2$	$\overline{CS}$	R/$\overline{W}$	Register Selected
1	X	1	1	DOR
1	X	1	0	DIR

"1" ..... "High", "0" ..... "Low"

This mode of operation is normally used for DMA (Direct Memory Access) transfer with the FDC.

When TxAK is a "Low" level the registers are selected by $\overline{CS}$, R/$\overline{W}$ and $RS_0 \sim RS_2$ as shown in Table 1.

- **Bus Direction (BD) Output**

The BD output is provided to control external bidirectional buffers on the data bus ($D_0 \sim D_7$) as shown in Figure 14. Its polarity is shown by Table 3.

Table 3 Bus Direction (BD) States

TxAK	$\overline{CS}$	BD
1	1	$\overline{R/W}$
0	1	0
0	0	R/$\overline{W}$

"1" ..... "High", "0" ..... "Low"

(Operation of BD as defined by this chart allows the FDC to function with the DMA Controller HD6844.)

Table 1 Address Codes for User Accessible Registers

TxAK	$\overline{CS}$	RS_2	RS_1	RS_0	R/$\overline{W}$	Registers
0	0	0	0	0	0	DOR (Data Output Register)
					1	DIR (Data Input Register)
0	0	0	0	1	1/0	CTAR (Current Track Address Register)
0	0	0	1	0	0	CMR (Command Register)
					1	ISR (Interrupt Status Register)
0	0	0	1	1	0	SUR (Set Up Register)
					1	STRA (Status Regiser A)
0	0	1	0	0	0	SAR (Sector Address Register)
					1	STRB (Status Register B)
0	0	1	0	1	0	GCR (General Count Register)
0	0	1	1	0	0	CCR (CRC Control Register)
0	0	1	1	1	0	LTAR (Logical Track Address Register)

"1" ..... "High", "0" ..... "Low"

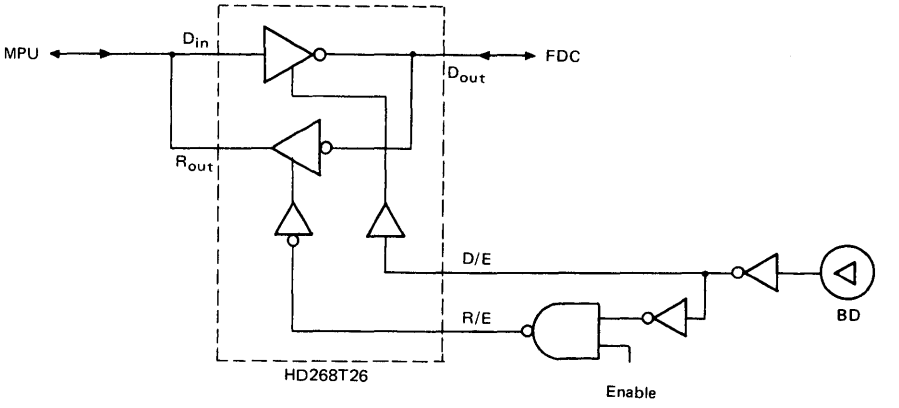


Figure 14 Bus Buffer Control

- **I/O and Control Pins**

Head Load (HLD) Output

HLD is used to notify the disk drive that the R/W head should be loaded (placed in contact with the media). When the FDC is ready for the head to load, HLD is a "High" level (logic "1"). A "Low" level (logic "0") HLD indicates the head should be unloaded.

Step (STP) Output

The STP output, in conjunction with HDR, is used to control head movement. A 32 μs wide positive (logic "1") pulse is generated on STP, to move the R/W head one track in the direction defined by the HDR output. The period of the STP signal is programmable by the SUR (Set-Up Register). The number of pulses generated on STP is the difference between the contents of the CTAR (Current Track Address Register), and the GCR (General Count Register) which contains the track address to which the head is to be moved.

Head Direction (HDR) Output

The HDR signal controls the direction of head movement. A "High" level (logic "1") signifies the head should step to the inside (toward the hub) of the disk. A "Low" level (logic "0") indicates the direction of head movement should be to the outside of the disk.

Low Current Track (LCT) Output

The LCT signal is used to control the level of write current used by the disk drive. LCT is a "Low" level (logic "0") when the write head is positioned over tracks 0~43. If it is over tracks 44~76, LCT is a "High" level (logic "1"). LCT is determined from the contents of the Current Track Address Register (CTAR).

Write Gate (WGT) Output

When a write operation is being performed, WGT is a logic "1" ("High" level). For a read operation, WGT is a "Low" level (logic "0").

File Inoperable Reset (FIR) Output

FIR is an output from the FDC to the floppy disk drive to reset it from an inoperable status. If the FI input is a "High" level, a pulse, of which width almost equals to E pulse "Low" width, is generated on the FIR output whenever Status Register B is read.

File Inoperable (FI) Input

FI is an input to the FDC from the drive. A "High" level indicates the drive is in an inoperable state. Its current state can be examined by reading bit 5 of Status Register B (STRB).

Track Zero (TRZ) Input

The TRZ input is reflected by bit 3 of STRA (Status Register A). The TRZ input must be a "High" level (logic "1") when the R/W head of the drive is positioned over track zero. A logic "1" on this input inhibits step pulses during a Seek Track Zero command.

Index (IDX) Input

The index input is received from the floppy disk drive and is used to sense the index hole in the disk media. The IDX signal is used to initialize the internal FDC timing. The state of the IDX input is reflected by bit 6 of Status Register A (STRA). A "High" level (logic "1") is to indicate the index hole is under the index sensor. The index input is used to count the number of disk revolutions while searching for the address ID field (see description of STRB bit 3).

Ready (RDY) Input

The ready input is received from the disk drive and can be read as bit 2 of STRA (Status Register A). A "High" level (logic "1") indicates the drive is ready and allows the FDC to operate the drive.

Write Protect (WPT) Input

WPT is an input indicating when the media is Write Protected. A "High" level during an FDC write operation results in a Write Error (STRB bit 6) but the FDC continues to perform the write function. The state of the WPT input can be read by examining bit 4 of the Status Register A (STRA).

Clock (CLK) Input

The CLK input is used to generate various timing sequences internal to the FDC. The head settling, seek time, step pulse width and write data pulse width, etc., are generated from the CLK input signal. The CLK is 1 MHz frequency and the duty is 50%.

- **Data Pins**

Data Clock (DCK) Input

DCK is used to clock data from the drive into the FDC. It is generated from the read data received from the drive.

Read Data (RDT) Input

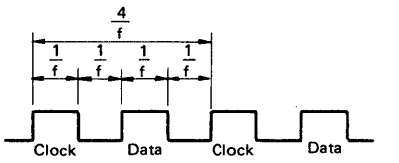
RDT is the serial data input from the drive. The data stream includes both the clock and data bits.

Write Data (WDT) Output

WDT is the double frequency modulated data output from the FDC. The time between clock bits is 4/f where f is the frequency of the clock input. The pulse width for both clock and data is 1/f (see Figure 15). For the normal clock frequency of 1 MHz the clock period is 4 μs, the clock pulse width is 1 μs and the data pulse width is 1 μs. Figure 15 shows the relationship between the WDT output and the frequency of the CLK inputs.

Variable Frequency Oscillator Control (VFOC) Output

VFOC is used as a sync signal during system diagnostics. Waveforms are shown in Figure 16.



f = Frequency of the CLK Input. To insure IBM3740 compatibility the clock frequency must be 1 MHz.

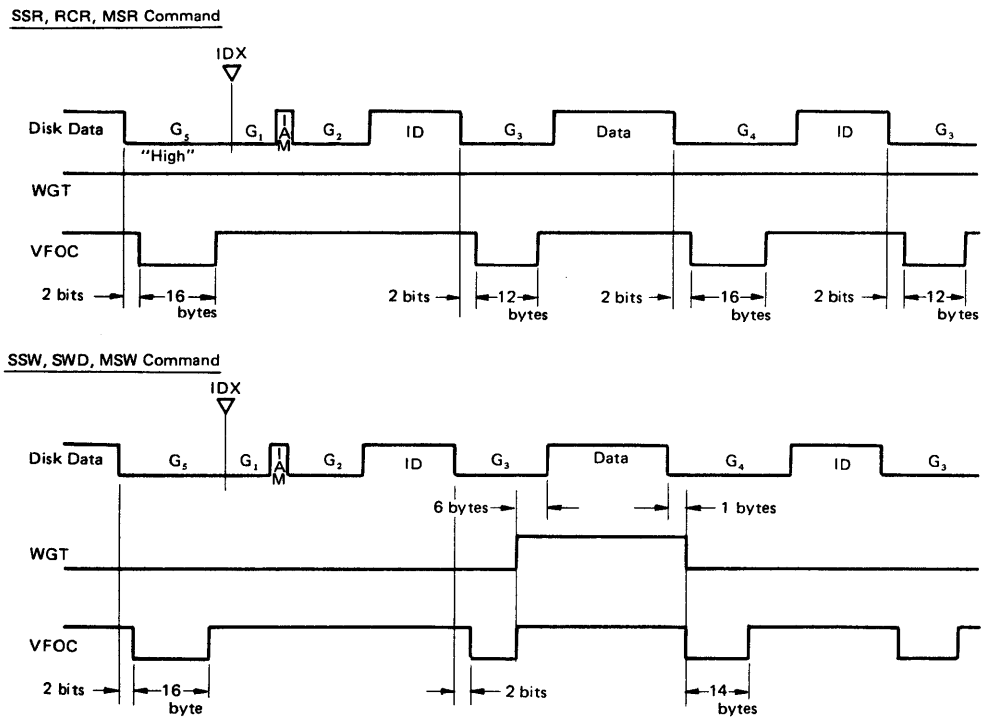
Figure 15 WDT Output Timing

■ FORMAT

The format used by the HD6843, shown in Figure 18, is compatible with the soft sector format of the IBM3740.

■ MACRO COMMAND SET

The macro command set shown in Table 4 is discussed in the following paragraphs.



In FFW Command, VFOC becomes "High" when WGT is at "High" level.
In FFR Command, VFOC remains "High".

Figure 16 Variable Frequency Oscillator Control Waveform (Relation Between WGT and VFOC)

SSW, SWD and MSW commands (Single Sector Write, Single Sector Write with Delet Data Mark, and Multi-Sector Write)

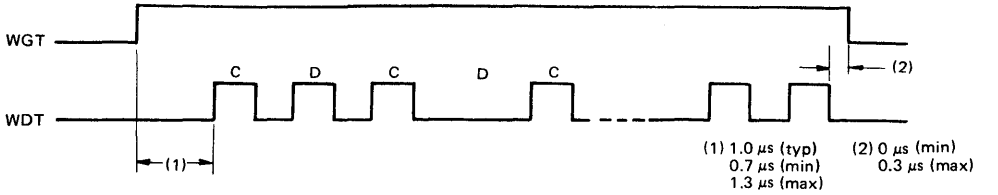


Figure 17 Write Data versus Write Gate Timing

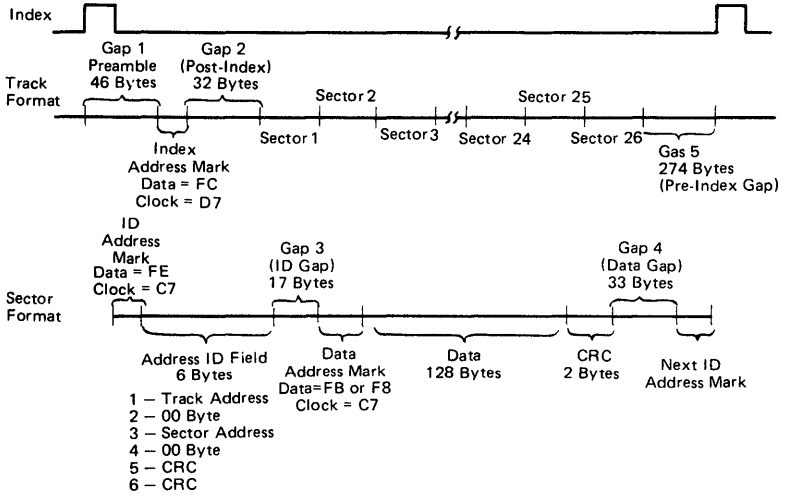


Figure 18 Soft Sector Format

Table 4 Macro Command Set

	Macro Command		CMR Bits				Hex Code
			Bit 3	Bit 2	Bit 1	Bit 0	
1	STZ	Seek Track Zero	0	0	1	0	2
2	SEK	Seek	0	0	1	1	3
3	SSR	Single Sector Read	0	1	0	0	4
4	SSW	Single Sector Write	0	1	0	1	5
5	RCR	Read CRC	0	1	1	0	6
6	SWD	Single Sector Write with Delete Data Mark	0	1	1	1	7
7	MSW	Multi Sector Write	1	1	0	1	D
8	MSR	Multi Sector Read	1	1	0	0	C
9	FFW	Free Format Write	1	0	1	1	B
10	FFR	Free Format Read	1	0	1	0	A

- **Seek Track Zero (STZ)**

The STZ command causes the R/W head to be released from the surface of the disk (HLD is reset) and positioned above track 00. The FDC issues step pulses on the STP output until the TRZ input becomes a "High" level or until 82 pulses have been sent to the drive. When the TRZ input becomes "High", the step pulses are inhibited on the STP output but the FDC remains busy until all 82 have been generated internally.

If the TRZ input remains "Low" (logic "0") after all 82 pulses have been generated, the seek error flag (STRB bit 4) is set.

After all 82 pulses have been generated, the head is loaded (HLD becomes a "High"). After the settling time specified in the SUR has expired, the Seek Command End flag is set (ISR bit 1), Busy STRA7 is reset, CTAR and GCR are cleared. The head remains in contact with the disk. A command such as RCR (Read CRC) may be issued following a STZ if the head must be released.

- **Seek (SEK)**

The SEK command is used to position the R/W head over the track on which a Read/Write operation is to be performed. The contents of the GCR are taken as the destination address and the content of the CTAR is the source address; therefore, the number of pulses (N) on the STP output are given by:

$$N = |(CTAR) - (GCR)|$$

HDR is a "High" for (GCR) > (CTAR) otherwise it is a "Low".

When a SEK command is issued, Busy is set, the head is raised from the disk, HDR is set as described above, and N number of pulses appear on the STP output. After the last step pulse is used, the head is placed in contact with the disk. Once the head settling time has expired, the Seek Command End flag (ISR bit 1) is set, Busy is reset, and the contents of the GCR are transferred to the CTAR.

■ SINGLE SECTOR READ/WRITE COMMANDS

The single sector Read/Write commands (SSR, RCR, SSW, and SWD) are used to Read/Write data from a single 128 byte sector on the disk. As shown in Figure 19 these types of instructions can be divided into two sections. The first section, which is common to all instructions, is the address search operation, while the second section is unique to the requirements of each instruction.

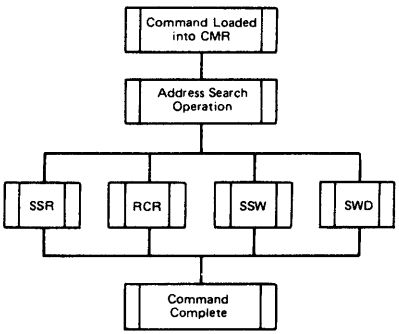


Figure 19 Basic Single Sector Command Flow Chart

- **Address Search Operation**

The flow chart of Figure 20 shows the operation of the address search operation.

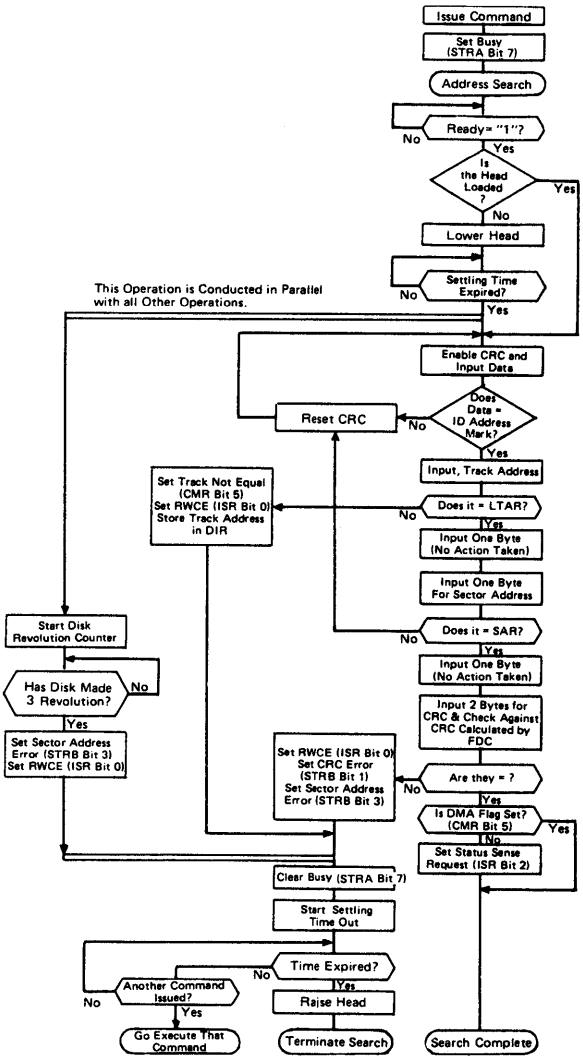


Figure 20 Operational Flow of the Address Search Sequence

- **Single Sector Read (SSR)**

The single sector read command follows the address search procedure as defined in the previous flowchart. If the search is successful, status sense request is set and the operation continues as described by the flowchart of Figure 21 .

- **Read CRC (RCR)**

The RCR command is used to verify that correct data was written on a disk. The operation is the same as for the SSR

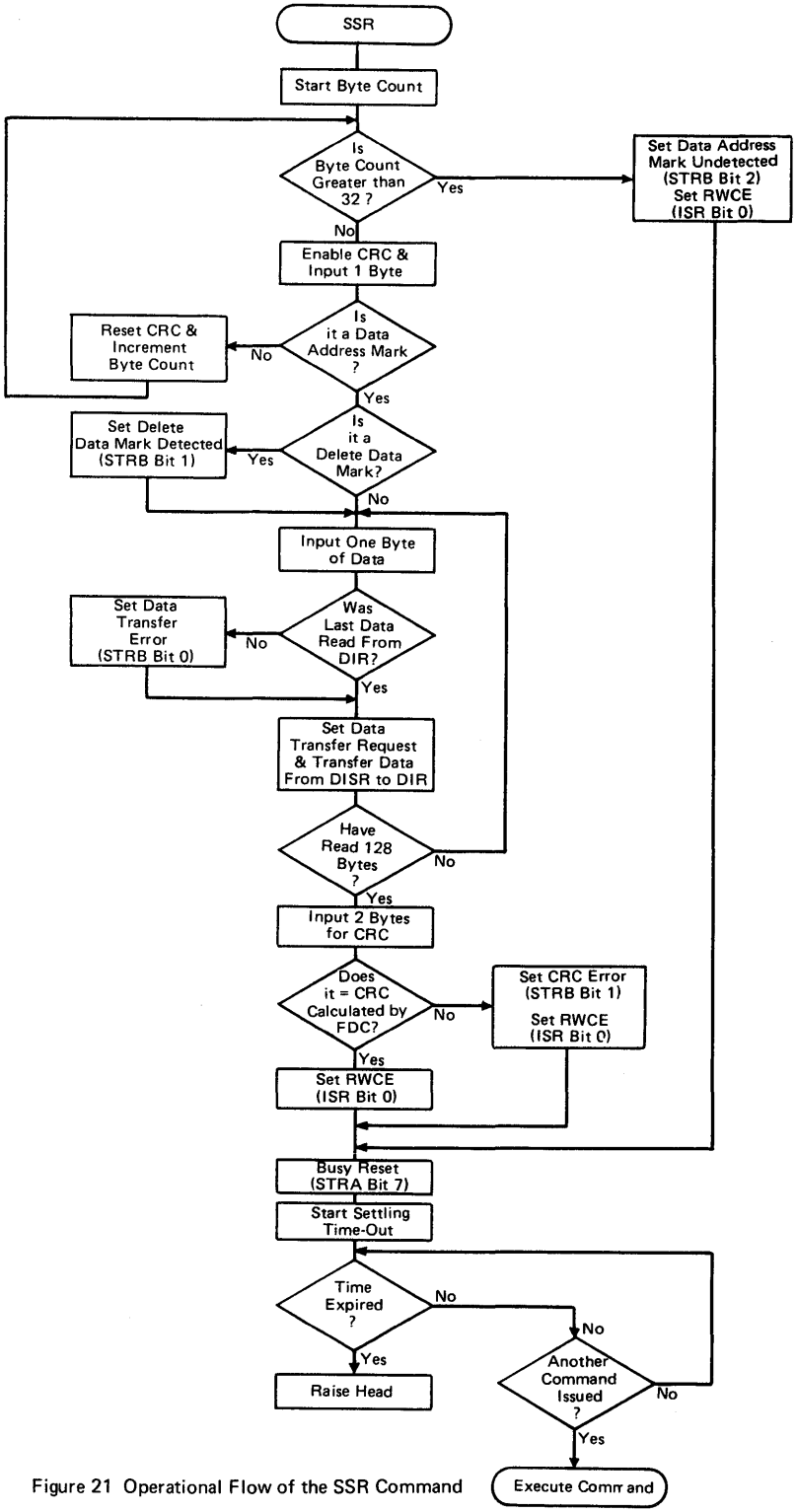


Figure 21 Operational Flow of the SSR Command

command with the exception that the data transfer request (STRA bit 0) is not set. The Status Sense Request interrupt can be disabled by using the DMA flag of CMR.

- **Single Sector Write (SSW)**

Single sector write is used to write 128 bytes of data on the disk. After the command is issued, the address search is performed. The remainder of the instruction's operation is shown in Figure 22.

- **Single Sector Write with Delete Data Mark (SWD)**

The operation flow of SWD is exactly like that of SSW. For SWD, the data pattern of the Data Address Mark becomes F8 instead of FB. The clock pattern remains C7.

- **Multi-Sector Commands (MSR/MSW)**

MSR is used for sequential reading of one or more sectors. If S sectors are to be read, S – 1 must be written into the GCR before the command is issued.

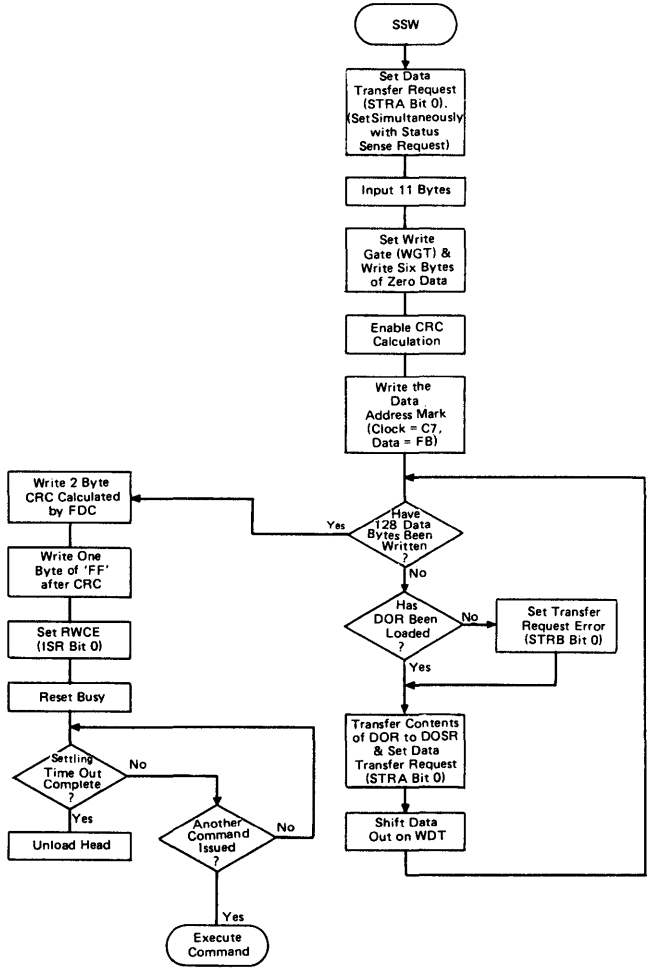


Figure 22 Operational Flow of the SSW Command

The basic operation for the MSR and MSW is the same as that for the SSR and SSW respectively. The basic operation begins with an address search operation, which is followed by a single sector read or write operation. This completes the operation on the first sector. The SAR is incremented, the GCR is decremented, and if no overflow is detected from the GCR (i.e., GCR become negative) the sequence is repeated until S number of sectors are read or written.

The completion of an MSR or MSW is like that of an SSR or SSW command. First RWCE is set and Busy is reset, after the settling time has expired, the head is released.

If a delete data mark is detected during an MSR command, STRA bit 1 (Delete Data Mark Detected) remains set throughout the commands operation.

When a multi-sector instruction is issued, the sum of the SAR and GCR must be less than 27. If SAR + GCR > 26, an address error (STRB bit 3 set) will occur after the contents of SAR becomes greater than 26.

- **Free Format Write (FFW)**

The FFW has two modes of operation which are selected by FWF (Free Format Write Flag) which is data bit 4 of the CMR.

When FWF = "0", the data bits of the DOR are written directly to the disk without first writing the preamble, address mark, etc. The contents of the DOR are FM modulated with a clock pattern of all ones.

If FWF = "1" the odd bits of the DOR are used as clock bits and even bits are used for data bits. In this mode, the DOSR clock is twice a normal write operation and one byte of DOR is one nibble (four bits of data) on the disk.

The two modes of the FFW command allow formatting a disk with either the IBM3470 format or a user defined format.

After the FFW command is loaded into the CMR, WGT becomes a "High" level, the contents of DOR are transferred to the DOSR, data transfer request (STRA bit 0) is set, and the serial bit pattern is shifted out on the WDT line. Therefore, DOR must be loaded before the FFW command is issued. Data from the DOR is continually transferred to the DOSR and shifted out on WDT until the CMR has been written with an all zero pattern. When CMR becomes zero, WGT becomes a "Low" level, but RWCE is not set and the R/W head is left in contact with the disk.

- **Free Format Read (FFR)**

FFR is used to input all data (including Address marks) from a disk. Once the FFR command is set into the CMR, the head is loaded and after the settling time has expired the serial data from the FDC is brought into the DISR. After 8 bits have accumulated, it is transferred to the DIR and Data Transfer Request (STRA bit 0) is set.

This operation continues until a zero pattern is stored in the CMR, terminating the FFR command. As in the case of the FFW command, RWCE is not set and the head remains in contact with the disk.

The first data that enters the DISR is not necessarily the first bit of a data word since the head may be lowered at any place on the disk. To prevent the FDC from remaining unsynchronized to the data, the FFR command will synchronize to an ID address mark (FE) or a Data Address mark (FB or F8) or an Index Address Mark (FC).

■ REGISTER DEFINITIONS

- **Data Output Register (DOR); Hex address 0, write only**

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
8 Bits of Data Used for a Disk Write Operation							

When one of the four write macro commands (SSW, SWD, MSW, and FFW) is executed, the information contained in the DOR is loaded into the DOSR, and is shifted out on the WDT line using a double frequency (FM) format.

- **Data Input Register (DIR); Hex address 0, read only**

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
8 Bits of Data Used for a Disk Read Operation							

One of the three read macro commands (SSR, MSR, FFR) executed, will cause the information on the RDT input to be clocked into the DISR. When 8 clock pulses have occurred, the 8 bits of information in the DISR are transferred to the DIR where it can be read by the bus interface.

- **Current Track Address (CTAR); Hex address 1, read/write**

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Track Address of Current Head position							

The address of the track over which the R/W head is currently positioned is contained in the CTAR. At the end of a SEK command, the contents of the GCR are transferred to the CTAR. CTAR is cleared at the completion of a STZ command. CTAR is a read/write register so that the head position can be updated when several drives are connected to one FDC. Bit 7 is read as a "0".

- **Command Register (CMR); Hex address 2, write only**

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3*	Bit 2*	Bit 1*	Bit 0*
Function Interrupt Mask	ISR3 Interrupt Mask	DMA Flag	FWF	Macro Command			

*Bit 0 ~ 3 are cleared by $\overline{RES}$.

The commands that control the FDC are loaded into the lower four bits of the CMR. Information that controls the data transfer mode and interrupt conditions are loaded into bits four through seven.

Bit 0~Bit 3: Macro Command

The Macro Command to be executed by the FDC is written to bits 0~3.

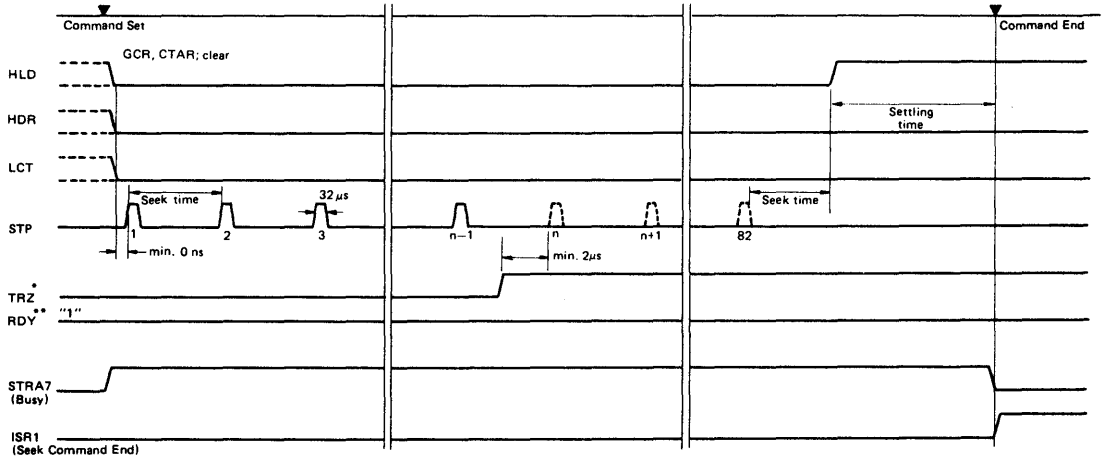
Bit 4: Free Format Write Flag (FWF)

If a Free Format Write command is issued, the state of bit 4 of the CMR determines what clock source will be used. The FWF is defined in the FFW (Free Format Write) command explanation.

Bit 5: DMA Flag

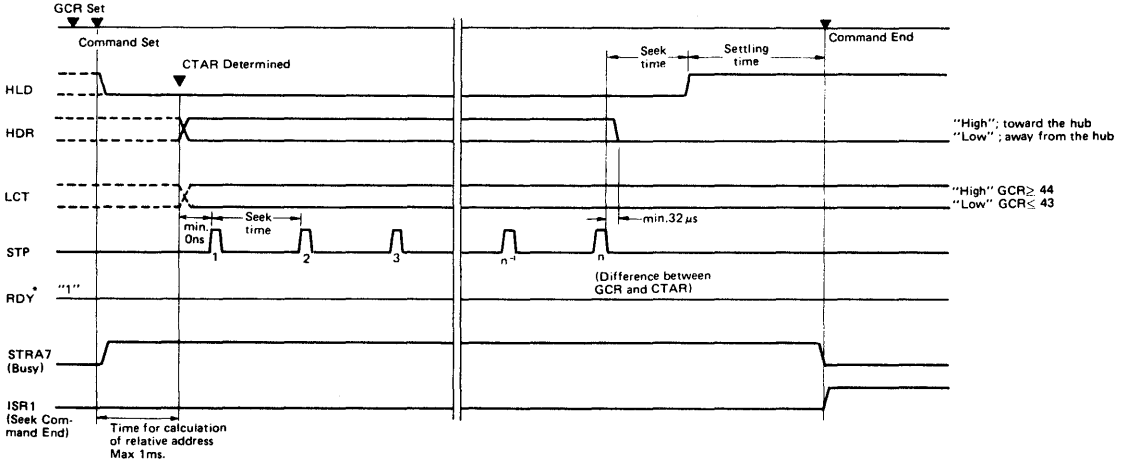
If bit 5 is a "1" the FDC is in the DMA mode. Bit 5 being a "1" inhibits setting of Status Sense Request (ISR bit 2) thereby preventing its associated interrupt. A logic "1" DMA flag also enables the TxRQ output allowing it to request DMA transfers when the Data Transfer Request flag (STRA bit 0) is set.

A logic "0" DMA flag indicates the program controlled I/O (PC I/O) mode.



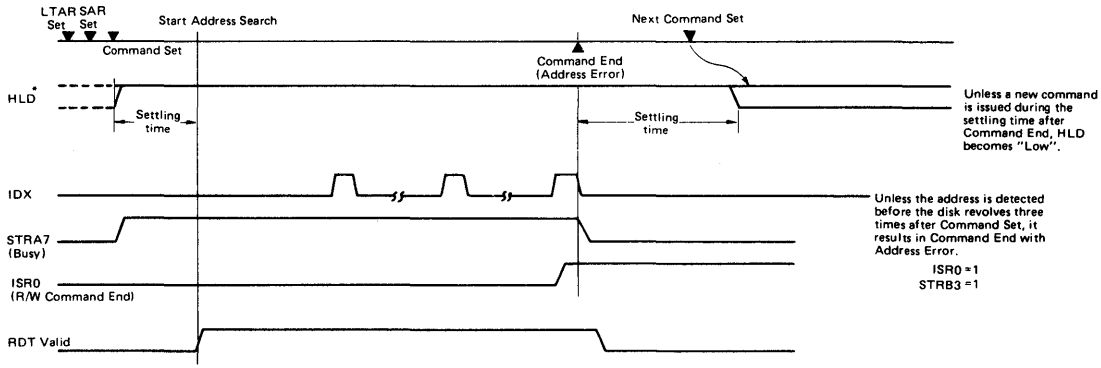
* STP output is masked when TRZ becomes "High". But if TRZ falls to "Low" again before 82 pulse outputs are all provided, STP output become available again from that time point.
** When RDY is "Low" with Command Set, the execution is postponed until RDY becomes "High".

Figure 23 Timing Sequence of STZ Command



* When RDY is "Low" with Command Set, the execution is postponed until RDY becomes "High"

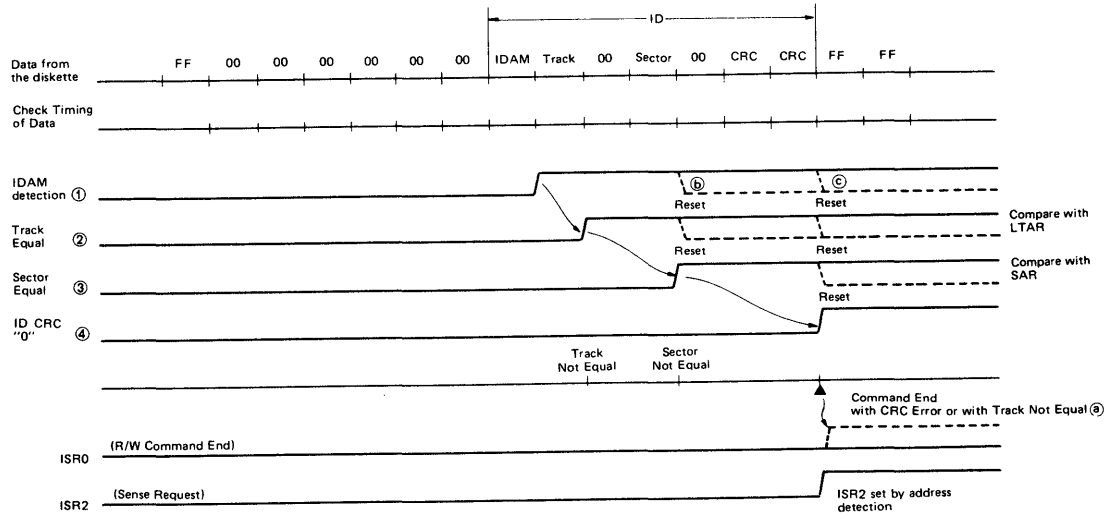
Figure 24 Timing Sequence of SEK Command



* If HLD has already been "High" when the command is set, the FDC starts the address search immediately.

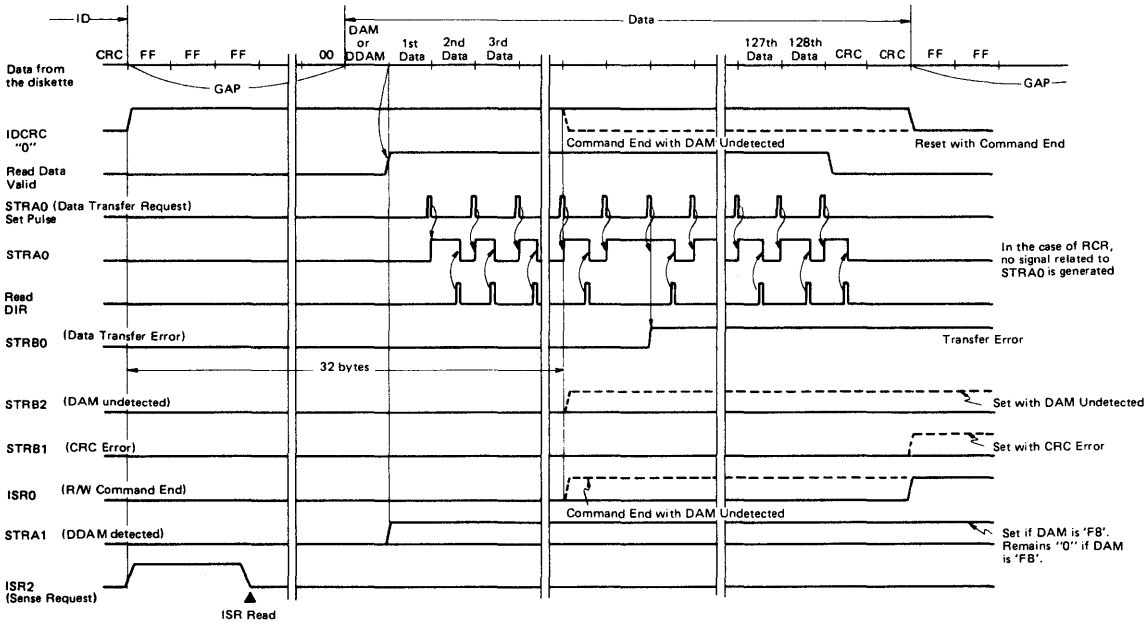
When RDY is "Low" with Command Set, the FDC waits for the execution until RDY becomes "High".

Figure 25 Timing Sequence of SSR, SSW, RCR, SWD, MSR, MSW Command (Relation with HLD and IDX)



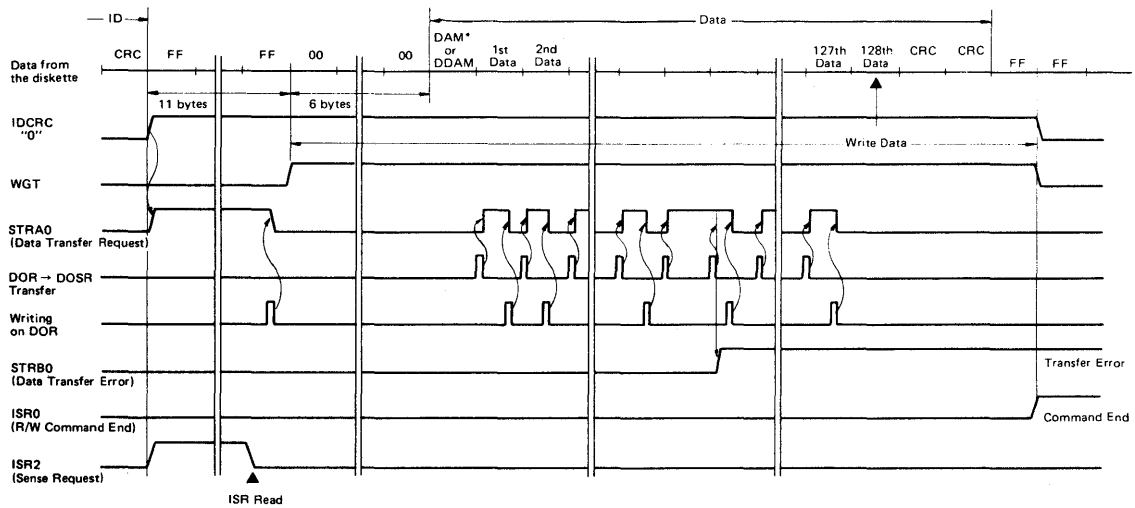
ⓐ ; In the case of Track Not Equal, ② is not set and if CRC equals to the one calculated by FDC, STRA5 is set.
ⓑ ; In the case of Sector Not Equal, ③ is not set and ① & ② are reset to search the next IDAM.
ⓒ ; In the case of CRC Error, ④ is not set and ①, ② & ③ are reset. (ISR0: Set, STRB1: Set, STRB3: Set)
When ①, ②, ③, & ④ are all set, ISR2 is Set. These four signals are reset with Command End.
When ④ is "1", go to the data transfer routine.

Figure 26 Internal Timing Sequence of Address Search Routine



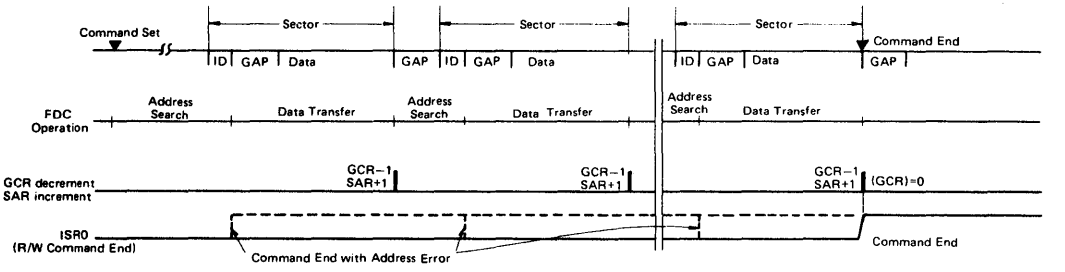
Unless DAM(FB) or DDAM(F8) is detected within 32 bytes after ID field has been detected, STRB2 is set to end the command.

Figure 27 Data Transfer Timing of SSR, RCR Command



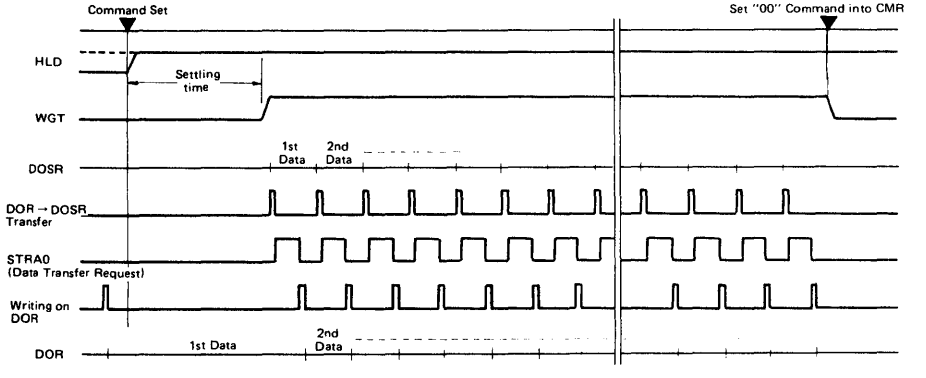
* As Data Address Mark, SSW command writes 'FB' and SWD command writes 'F8'.

Figure 28 Data Transfer Timing of SSW, SWD Command



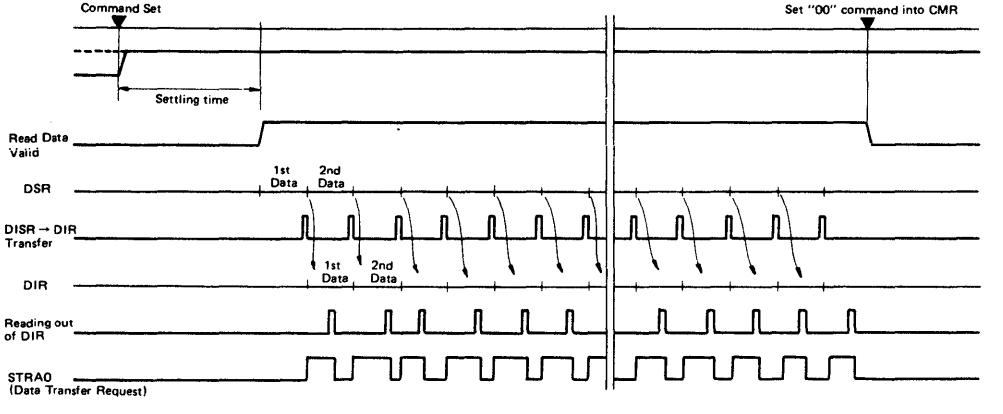
Address Search and Data Transfer in each sector is the same as those of SSR or SSW command. When Address Error occurs, it results in Command End. If an error relating to Data Transfer occurs, Error flag is set. But the command continues to be executed to shift into the next sector.

Figure 29 Timing Sequence of MSR, MSW Command



- The first one-byte data must be set into DOR before Command Set.
- If HLD has already been "High" when the command is set, WGT becomes "High" immediately.
- When '00' command is set into CMR, an interrupt of Command End is not generated.

Figure 30 Timing Sequence of FFW Command



If HLD has already been "High" when the command is set, Read operation starts immediately without waiting for the settling time. When "00" command is set into CMR, an interrupt of Command End is not generated.

Figure 31 Timing Sequence of FFR Command

Bit 6: ISR3 Interrupt Mask

CMR bit 6 (ISR3 Mask) is used to control the operation of ISR bit 3. A logic "1" in CMR bit 6 inhibits output of STRB-OR-Interrupt signal to $\overline{IRQ}$. If CMR bit 6 (ISR3 Mask) and CMR bit 7 are "0" STRB-OR-Interrupt signal will be output to $\overline{IRQ}$.

Bit 7: Function Interrupt Mask

When CMR bit 7 is a logic "1" all interrupts are inhibited.

Table 5

Causes of Interrupt	Command Register Masks That Affect Interrupts		
	CMR7 (Function Interrupt Mask)	CMR6 (ISR3 Mask)	CMR5 (DMA Flag)
ISR0 (Read write Command End)	M	X	X
ISR1 (Seek Command End)	M	X	X
ISR2 (Status Sense Request)	M	X	M
ISR3 (STRB-OR-Interrupt)	M	M	X

X = No effect
M = Bits that are used as masks

- **Interrupt Status Register (ISR); Hex address 2, read only**

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2*	Bit 1*	Bit 0*
Not Used (Read as "0")				STRB -OR	Status Sense Request	Seek Command End	Read Write Command End

* Cleared by $\overline{RES}$

Bit 0: Read Write Command End (RWCE)

When an SSR, RCR, SSW, SWD, MSR or MSW Macro Command has completed execution, bit 0 becomes set (logic "1"). If the function interrupts are enabled (bit 7 of CMR is a logic "0"), the conclusion of a Macro Command's execution will cause an interrupt.

Bit 1: Seek Command End (SCE)

Seek Command End is set on SEK and STZ commands to indicate the head has been loaded and the settling time specified in SUR has expired. Since RWCE is not set for the SEK or STZ command, SCE can be used as an interrupt to signify the SEK or STZ command has finished. SCE is not set for any of the R/W commands.

Bit 2: Status Sense Request

For an SSR, SSW, SWD, MSR, or MSW Command, Status Sense Request indicates that the specified address ID field has been detected and verified by a CRC check. This is used as an early indication that data transfers will occur after 18 more byte times. For MSR and MSW commands, it is set for each sector.

In the PC I/O mode, an interrupt occurs when Status Sense Request becomes a logic "1". In the DMA mode, (DMA flag of CMR is set) Status Sense Request is unchanged and does not generate an interrupt when the address ID field has been verified.

Bit 3: STRB-OR

STRB-OR is an "OR" of all of the bits of Status Register B.

$$\text{STRB-OR} = \text{STRB0} + \text{STRB1} + \text{STRB2} + \text{STRB3} + \text{STRB4} + \text{STRB5} + \text{STRB6} + \text{STRB7}$$

$$\text{STRB-OR-Interrupt} = \text{STRB1} + \text{STRB2} + \text{STRB3} + \text{STRB4} + \text{STRB5} + \text{STRB6} + \text{STRB7}$$

STRB-OR-Interrupt signal causes $\overline{IRQ}$. STRB-OR is read by Read ISR. STRB0 (Data Transfer Error) sets ISR Bit 3 but does not cause Interrupt.

ISR0, ISR1, and ISR2 are cleared when the Interrupt Status Register is read, but ISR3 is cleared only after Status Register B has been read except when FI input is "High".

- **Set-Up Register (SUR); Hex address 3, write only**

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Track to Track Seek Time				Head Settling Time			

The SUR is not affected by a reset operation; therefore, once it is initialized, the information remains until power is removed from the FDC.

Bit 0 ~ Bit 3: Head Settling Time

The head settling time is used to generate a delay after the head is placed in contact with the disk. This allows the head to stop bouncing before any operations are performed. The delay is programmed by bits 0~3 and is specified by the equation:

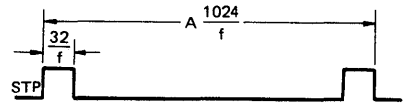
$$\text{Delay} = \frac{4096}{f} \cdot B$$

B = Number contained in bits 0~3 of SUR
f = Frequency of CLK input

For IBM3740 compatibility f = 1 MHz and the timing range is 4.096 ms for a "0001" to 61.44 ms for a "1111". A "0000" code prevents Settling Time complete from being set and the FDC must be Reset.

Bit 4 ~ Bit 7: Track to Track Seek Time

The frequency of STP is determined by bit 4~ bit 7 of SUR as shown below.



A = Number specified in bits 4~7 of SUR.
f = Frequency of CLK input.

For IBM compatible operation, f is 1 MHz. This results in an STP pulse width of 32 μs and an STP interval of 1.024 ms for a "0001" to 15.36 ms for a "1111".

- **Status Register A (STRA); Hex address 3, read only**

Bit 7*	Bit 6	Bit 5*	Bit 4	Bit 3	Bit 2	Bit 1*	Bit 0*
Busy	Index	Track Not Equal	Write Protect	Track Zero	Drive Ready	Delete Data Mark Detected	Data Transfer Request

* Cleared by $\overline{RES}$

Bit 0: Data Transfer Request

For a write operation (SSW, SWD, MSW, FFW) the transfer request bit indicates that the DOR is ready to accept the next data word to be written on the disk. If data is not written into the DOR before the last data bit in the DOSR is shifted out to the WDT line; the data transfer error bit (bit 0 of STRB) will be set. After a write command has been issued, the first transfer request occurs simultaneously with the Status Sense Request. For a write operation, transfer request is reset after the DOR has been written from the data bus.

During a read operation (SSR, MSR, FFR) the transfer request bit signifies data from the DISR has been transferred to the DIR. The DIR must be read before the DISR is full again or the data transfer error bit (bit 0 of STRB) will be set. For read operations, transfer request is reset by a read of the DIR.

Bit 1: Delete Data Mark Detected

A Single Sector Read operation that detects a delete data code (F8) instead of a general data code (FB) as a Data Address Mark will set the Delete Data Mark Detected bit. For the MSR command, bit 1 is set the first time an "F8" code is found and remains set throughout the execution of the command. Bit 1 is reset whenever an SSR, SSW, SWD, MSR, MSW, or RCR command is issued.

Bit 2: Drive Ready

The Drive Ready bit indicates the state of the Ready input from the floppy disk drive. If a command is issued with Ready at logic "0", its execution will be inhibited until Ready becomes a logic "1". If ready becomes a "0" during the execution of a command the Hard Error Flag (STRB bit 7) is set.

Bit 3: Track Zero

The state of the Track Zero input from the floppy disk drive is reflected in this bit of STRA. A logic "1" on the Track Zero input inhibits step pulses during an STZ command.

Bit 4: Write Protect

The Write Protect input from the floppy disk drive is reflected by bit 4 of STRA. A "High" level (logic "1") on the WPT input during the execution of any write command results in a write error (bit 6 of STRB set).

Bit 5: Track Not Equal

If the track address read from the address ID field does not coincide with the address in the LTAR inspite of CRC matching the one calculated by FDC, the Track Not Equal bit is set. Track Not Equal applies to all non-free format read/write commands, and is reset after a non-free format read/write command is issued.

Bit 6: Index

The state of the index input appears in bit 6 of STRA. The index input is used to count the number of disk revolutions while the FDC is looking for the address ID field (see operation of STRB bit 3) during the address search phase of a non-free format read/write command.

Bit 7: Busy

When Busy is a logic "1", the FDC is executing a command and no new commands can be issued. Busy should be confirmed to be "0" before reading ISR or STRB as well as issuing a command.

- **Sector Address Register (SAR); Hex address 4, write only**

Bit 7	Bit 6	Bit 5	Bit 4*	Bit 3*	Bit 2*	Bit 1*	Bit 0*
Not Used			5 Bit Sector Address				

* Cleared by $\overline{RES}$

Before a data transfer macro command (SSW, SWD, SSR, RCR, MSW, MSR) is issued, the address of the sector on which the operation is to be performed must be written into the SAR. The address in the sector address byte of an Address ID field of the disk is compared with the contents of the SAR. During an MSW or MSR command, the SAR is incremented after each sector is read or written. When execution is complete, the SAR contains the address of the last sector on which an operation was performed plus one.

- **Status Register B(STRB); Hex address 4, read only**

Bit 7*	Bit 6*	Bit 5	Bit 4*	Bit 3*	Bit 2*	Bit 1*	Bit 0*
Hard Error	Write Error	File Inoperable	Seek Error	Sector Address Undetected	Data Mark Undetected	CRC Error	Data Transfer Error

* Cleared by $\overline{RES}$

The bits of the STRB represent possible error conditions that may occur during execution of macro commands. Whenever STRB is reset, ISR bit 3 is also reset.

Bit 0: Data Transfer Error

Data Transfer Error indicates an underflow or overflow of data. If a Write operation is being performed, it signifies that data was not presented to the DOR before the DOSR became empty. In this case, the current contents of the DOR are transferred to the DOSR and the write operation continues. The data transfer error remains set until STRB is read, and the data transfer request remains set until data is written into the DOR. The operation of the CRC is unchanged.

For read commands, a data transfer error indicates that data in the DIR was not read before the next data word from the disk was transferred to the DIR. The read operation continues until sufficient data has been read from the disk to satisfy the requirements of the command (128 bytes for SSR). The error indication remains set until STRB is read, and the transfer request remains set until data is read from the DIR.

Bit 1: CRC Error

A CRC error occurs when the CRC read from the disk does not match that calculated by the FDC on the data it reads from the disk. A CRC error can occur in two different situations; checking the address ID field, checking the data field.

If the CRC error occurs during the check of an address ID field, Sector Address Undetected (STRB bit 3) will also be indicated (see Table 6). A CRC error of a data field is indicated by a CRC Error and no Sector Address Undetected.

Bit 2: Data Mark Undetected

If a valid data mark is not detected in the data block of a sector, it is indicated by a Data Mark Undetected error.

Bit 3: Sector Address Undetected

The Sector Address Undetected bit can be set on two conditions; not finding the sector address and a CRC error on an address ID field.

If the disk makes three revolutions during an address search operation and the sector address specified in the sector address register is not found in any of the address ID fields, a Sector Address Undetected condition is indicated.

A CRC error that occurs on an address ID field will set bit 3 also. Table 6 shows how bits 1 and 3 are related.

Table 6 Relationship of CRC Error and Sector Address Undetected

CRC Error (STRB1)	Sector Address Undetected (STRB3)	Condition
0	0	No Error
0	1	Sector Address not Detected
1	0	CRC Error on a Data Field
1	1	CRC Error on Address ID Field

Bit 4: Seek Error

An STZ (Seek Track Zero) command that never receives a track zero indication on the track zero input will result in a Seek Error (see description of STZ command).

Bit 5: File Inoperable

The state of the File Inoperable input appears in bit 5. If the File Inoperable input is a "High" level, a pulse of width equals to Enable pulse width PW_{EL} is issued on the FIR output when STRB is read. FI is not latched but the input is gated to the bus when STRB is read.

Bit 6: Write Error

If the WPT input becomes a "High" level (logic "1") during the execution of a write command the Write Error bit is set.

Bit 7: Hard Error

If the Ready input becomes a "Low" level during the operation of a command (Busy is set), a Hard Error indication will result.

- **General Count Register (GCR); Hex address 5, write only**

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Not Used	7 Bit Count for Track Number on SEK Command and Sector Count for MSR or MSW Command						

The GCR contains the destination track address for the R/W head on an SEK Macro Command. The contents of the GCR are transferred to the CTAR at the end of the SEK Command. For multi-sector read or write operations (MSR, MSW), the GCR contains the number of sectors to be read minus one. During the MSR or MSW execution the GCR is decremented after each sector is read or written.

- **CRC Control Register (CCR) ; Hex address 6, write only**

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Not Used						Shift CRC	CRC Enable

The CCR information is used only in the free format commands; for all other commands this register is masked and has no function.

Bit 0: CRC Enable

During an FFW command, CRC Enable is set by software and CRC generation takes effect on the next transfer of data from DOR to DOSR (see figure 32). The CRC generation continues until Shift CRC (CCR bit 1) is set.

For an FFR command, CRC Enable is set by software and CRC generation takes effect on the next data read from DIR. The calculation continues for all data bytes read from DIR until CRC Enable is reset. The bytes read previous to resetting CRC Enable are considered the CRC information bytes and the CRC check is made against them.

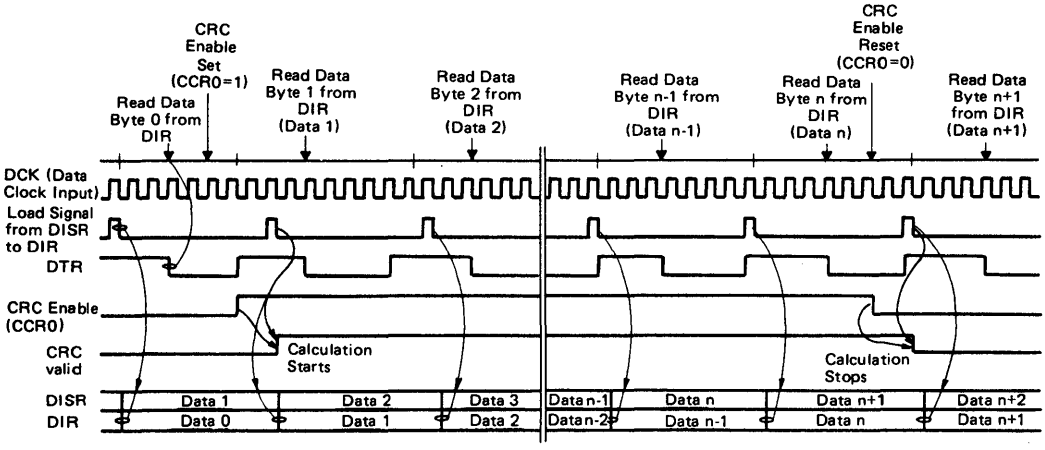
Bit 1: Shift CRC

Bit 1 is valid only for the FFW command. After setting, it takes effect on the next transfer of data from DOR to DOSR (see Figure 33). Setting Shift CRC terminates the CRC calculation and causes the CRC calculated on all the data written into DOR up to the setting of bit 1, to be shifted out the WDT output. The CRC calculation will not include any data written to DOR after Shift CRC is set.

- **LTAR (Logical Track Address); Hex address 7, write only**

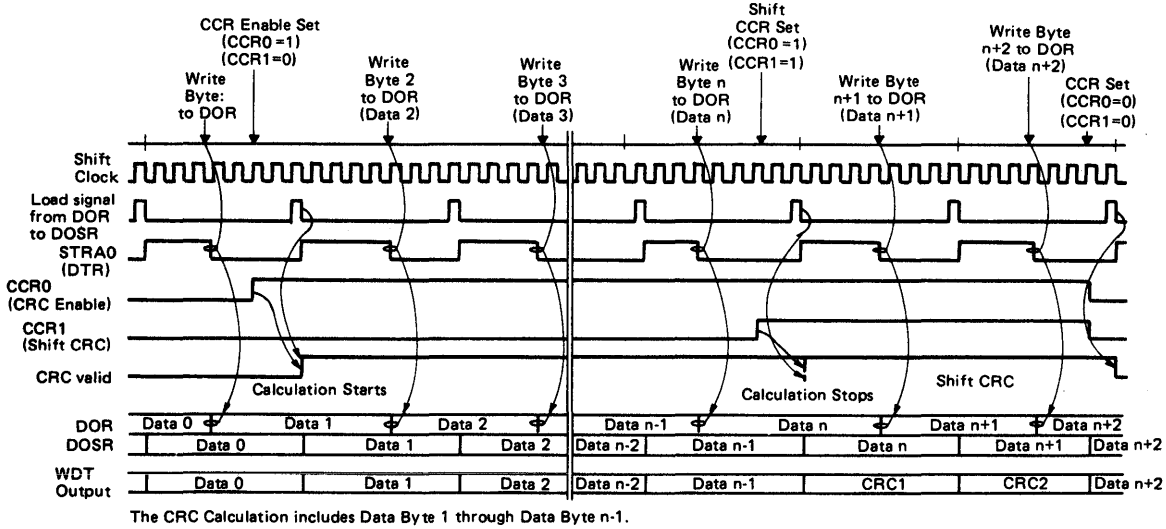
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Not Used	7 Bit Logical Track Address						

When a read or write macro command (SSW, SWD, SSR, RCR, MSW, MSR) is issued, the address of the track on which the operation is to be performed must be written into the LTAR. The address in the track address byte of an Address ID field of the disk is compared with the contents of the LTAR. The contents of LTAR are not affected by the execution of any of the commands.



CRC Calculation includes Data Byte 1 through Data Byte n.

Figure 32 CCR Control Register Timing for an FFR Command (READ)



The CRC Calculation includes Data Byte 1 through Data Byte n-1.

Figure 33 CCR Control Register Timing for an FFW Command (WRITE)

Table 7 Programming Reference Data

Table 7 is a summary of the information in the data sheet and can be used as a reference when programming the HD6843.

Registers	Hex Address	R/W Mode	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DOR	0	WO	8 Bits of Data Used for a Disk Write Operation							
DIR	0	RO	8 Bits of Data Used for a Disk Read Operation							
CTAR	1	R/W	Track Address of Current Head Position							
CMR	2	WO	Function Interrupt Mask	ISR3 Interrupt Mask	DMA Flag	FWF	Macro Command			
ISR	2	RO	Not Used				STRB -OR	Status Sense Request	Seek Command End	Read Write Command End
SUR	3	WO	Track to Track Seek Time				Head Settling Time			
STRA	3	RO	Busy	Index	Track Not Equal	Write Protect	Track Zero	Drive Ready	Delete Data Mark Detected	Data Transfer Request
SAR	4	WO	Not Used			5 Bit Sector Address				
STRB	4	RO	Hard Error	Write Error	File Inoperable	Seek Error	Sector Address Undetected	Data Mark Undetected	CRC Error	Data Transfer Error
GCR	5	WO	Not Used	7 Bit Count for Track Number on SEK or Sector Count for MSR or MSW.						
CCR	6	WO	Not Used						Shift CRC	CRC Enable
LTAR	7	WO	Not Used	7 Bit Logical Track Address						

RO – Read Only
WO – Write Only
R/W – Read/Write

* Cleared by $\overline{RES}$

MACRO COMMANDS

Hex Code	Instruction	Hex Code	Instruction
2	STZ	A	FFR
3	SEK	B	FFW
4	SSR	C	MSR
5	SSW	D	MSW
6	RCR		
7	SWD		

Table 8 Error Condition, Command Execution, Interrupt, and Head Control

Error	Flag	Set Condition	Reset Condition	Command	Command Execution	Interrupt	Head Control
Track Not Equal	STRA5	Track information of ID field is not equal to the content of LTAR.	Issuing of SSR, RCR, MSR, SSW, SWD or MSW Command	SSR, RCR, MSR SSW, SWD, MSW	The execution of a command is interrupted and R/W Command End (ISR0) is set.	Request (ISR0)	Unchanged**
Data Transfer Error	STRB0	Overrun or underflow during the data transfer	Reading of STRB	SSR, MSR, SSW, SWD, MSW, FFR FFW	Read/Write command continues to be executed.	No interrupt	Unchanged**
CRC Error	STRB1	CRC Error on ID field or Date field	Reading of STRB	SSR, RCR, MSR, SSW, SWD, MSW (FFR)	The execution of a command is interrupted and R/W Command End (ISR0) is set.	Request (ISR0, ISR3)	Unchanged**
Data Mark Undetected	STRB2	DAM or DDAM is undetected within 32 bytes after ID field has been detected.	Reading of STRB	SSR, RCR, MSR	The execution of a command is interrupted and R/W Command End (ISR0) is set.	Request (ISR0, ISR3)	Unchanged**
Sector Address Undetected	STRB3	(1) Sector Address of ID field is not equal to the content of SAR. (2) CRC Error on ID field	Reading of STRB after Busy (STRA7) is reset.	SSR, RCR, MSR SSW, SWD, MSW	The execution of a command is interrupted and R/W Command End (ISR0) is set.	Request (ISR0, ISR3)	Unchanged (Head remains loaded after settling time has expired.)
Seek Error	STRB4	TRZ signal semains "Low" level though eighty-two STP pulse outputs are provided in STZ command.	Reading of STRB	STZ	The execution of a command is interrupted and Seek Command End (ISR1) is set.	Request (ISR1, ISR3)	Unchanged
File Inoperable	STRB5	A "High" level input of FI terminal is reflected.	FI signal of the FDD is reset when "High" pulse output is provided by reading of STRB at FI="1".	All commands	The execution of a command is interrupted. If it is a Read/Write command, ISR0 is set. If it is a seek command, ISR1 is set.	Request (ISR0 or ISR1, ISR3)	Unload the head imediately (HLD="Low") Set WGT to "Low"
Write Error	STRB6	Write operation (WGT="High") is performed when the input of WPT terminal is "High" level.	Reading of STRB	SSW, SWD, MSW FFW	The execution of a command is interrupted and R/W Command End (ISR0) is set.	Request (ISR0, ISR3)	Unload the head imediately (HLD="Low" Set WGT to "Low"
Hard Error	STRB7	RDY input signal becomes "Low" level during the execution of a command (Busy="1".)	Reading of STRB	All commands	The execution of a command is interrupted. If it is a Read/Write command, ISR0 is set. If it is a seek command, ISR1 is set.	Request (ISR0 or ISR1, ISR3)	Unload the head imediately (HLD="Low") Set WGT to "Low"
Not Ready during the idling	STRA2	–	–	–	–	No interrupt	Unload the head imediately (HLD="Low")

* These errors except STRB5 and STRA2 are reset by $\overline{RES}$ inputs.

** Head is unloaded if the new command is not issued during the settling time after Read/Write command ends.

HD6844, HD68A44, HD68B44
DMAC (Direct Memory Access Controller)

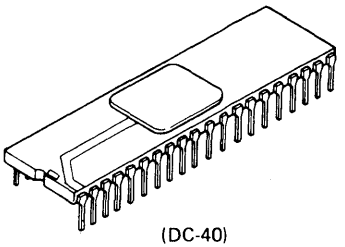
The HD6844 Direct Memory Access Controller (DMAC) performs the function of transferring data directly between memory and peripheral device controllers. It controls the address and data buses in place of the MPU in bus organized systems such as the HMCS6800 Microprocessor System.

The bus interface of the HD6844 includes select, read/write, interrupt, transfer request/grant, and bus interface logic to allow the data transfer over an 8-bit bidirectional data bus. The functional configuration of the DMAC is programmed via the data bus. The internal structure provides for control and handling of four individual channels, each of which is separately configured. Programmable control registers provide control for the transfer location and length, individual channel control and transfer mode configuration, priority of servicing, data chaining, and interrupt control. Status and control lines provide control to the peripheral controllers.

The mode of transfer for each channel can be programmed as cycle-stealing or a burst transfer mode.

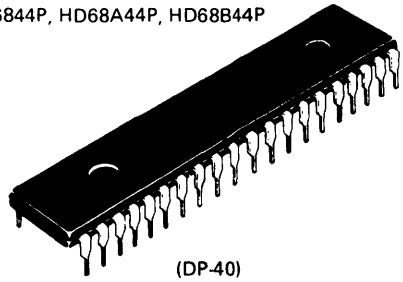
Typical applications would be with the Floppy Disk Controller (FDC), etc..

HD6844, HD68A44, HD68B44



(DC-40)

HD6844P, HD68A44P, HD68B44P

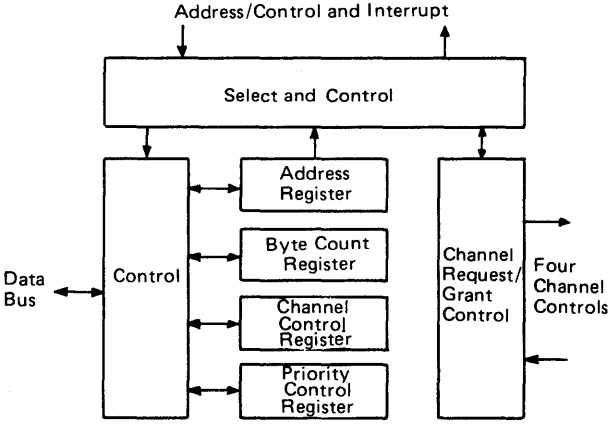


(DP-40)

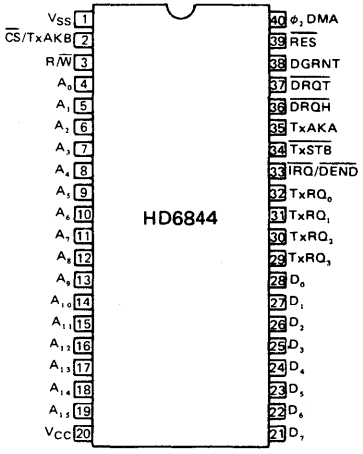
■ FEATURES

- Four DMA Channels, Each Having a 16-Bit Address Register and a 16-Bit Byte Count Register
- 1 M Byte/Sec (HD6844), 1.5 M Byte/Sec (HD68A44), 2.0 M Byte/Sec (HD68B44) Maximum Data Transfer Rate
- Selection of Fixed or Rotating Priority Service Control
- Separate Control Bits for Each Channel
- Data Chain Function
- Address Increment or Decrement Update
- Programmable Interrupts and DMA End to Peripheral Controllers
- Compatible with MC6844, MC68A44, MC68B44

■ BLOCK DIAGRAM



■ PIN ARRANGEMENT



(Top View)

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC}*	-0.3 ~ +7.0	V
Input Voltage	V_{in}*	-0.3 ~ +7.0	V
Operating Temperature	T_{opr}	-20 ~ +75	°C
Storage Temperature	T_{stg}	-55 ~ +150	°C

* With respect to V_{SS} (SYSTEM GND)

(NOTE) Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ RECOMMENDED OPERATING CONDITIONS

Item	Symbol	min	typ	max	Unit
Power Supply Voltage	V_{CC}*	4.75	5.0	5.25	V
Input Voltage	V_{IL}*	-0.3	–	0.8	V
	V_{IH}*	2.0	–	V_{CC}	V
Operating Temperature	T_{opr}	-20	25	75	°C

* With respect to V_{SS} (SYSTEM GND)

■ ELECTRICAL CHARACTERISTICS (V_{CC}=5V±5%, V_{SS}=0V, Ta=-20~+75°C, unless otherwise noted.)

● DC CHARACTERISTICS

Item		Symbol	Test Condition	min	typ*	max	Unit
Input "High" Voltage		V_{IH}		2.0	–	V_{CC}	V
Input "Low" Voltage		V_{IL}		-0.3	–	0.8	V
Input Leakage Current	TxRQ$_0$~$_3$, ϕ_2DMA, $\overline{RES}$, DGRNT	I_{in}	V_{in}=0~5.25V	-2.5	–	2.5	μA
Three-State (off state) Leakage Current	A_0~A_{15}, D_0~D_7, R/$\overline{W}$	I_{TSI}	V_{in}=0.4~2.4V	-10	–	10	μA
Output "High" Voltage	D_0~D_7	V_{OH}	I_{OH}=-205μA	2.4	–	–	V
	A_0~A_{15}, R/$\overline{W}$		I_{OH}=-145μA	2.4	–	–	
	All Other Outputs		I_{OH}=-100μA	2.4	–	–	
Output "Low" Voltage		V_{OL}	I_{OL}=1.6mA	–	–	0.4	V
Source Current	$\overline{CS}$/TxAKB	I_{CSS}	V_{in}=0V, Fig. 10	–	10	16	mA
Power Dissipation		P_D		–	500	1000	mW
Input Capacitance	ϕ_2DMA	C_{in}	V_{in}=0V, T_a=25°C f=1.0MHz	–	–	20	pF
	D_0~D_7, $\overline{CS}$, A_0~A_4, R/$\overline{W}$			–	–	12.5	
	TxRQ$_0$~$_3$, $\overline{RES}$, DGRNT			–	–	10	
Output Capacitance		C_{out}	V_{in}=0V, T_a=25°C, f=1MHz	–	–	12	pF

* V_{CC}=5.0V, T_a=25°C

- **AC CHARACTERISTICS (Load Condition Fig. 9)**

1. CLOCK TIMING

Item		Symbol	Test Condition	HD6844			HD68A44			HD68B44			Unit
				min	typ	max	min	typ	max	min	typ	max	
ϕ_2 DMA Cycle Time		$t_{cyc\phi}$	Fig. 2	1,000	–	–	666	–	–	500	–	–	ns
ϕ_2 DMA Pulse Width	"High" Level	$PW_{\phi H}$	Fig. 2	450	–	–	280	–	–	235	–	–	ns
	"Low" Level	$PW_{\phi L}$	Fig. 2	400	–	–	230	–	–	210	–	–	ns
ϕ_2 DMA Rise and Fall Time		$t_{\phi r}$, $t_{\phi f}$	Fig. 2	–	–	25	–	–	25	–	–	25	ns

2. DMA TIMING (Load Condition Fig. 9)

Item		Symbol	Test Condition	HD6844			HD68A44			HD68B44			Unit
				min	typ	max	min	typ	max	min	typ	max	
TxRQ Setup Time	ϕ_2 DMA Rising Edge	t_{TQS1}	Fig. 3	120	–	–	120	–	–	120	–	–	ns
	ϕ_2 DMA Falling Edge	t_{TQS2}		210	–	–	210	–	–	155	–	–	
TxRQ Hold Time	ϕ_2 DMA Rising Edge	t_{TQH1}		20	–	–	10	–	–	10	–	–	ns
	ϕ_2 DMA Falling Edge	t_{TQH2}		20	–	–	10	–	–	10	–	–	
DGRNT Setup Time	DGRNT	t_{DGS}	Fig. 4	155	–	–	125	–	–	115	–	–	ns
DGRNT Hold Time	DGRNT	t_{DGH}		10	–	–	10	–	–	10	–	–	
Address Output Delay Time	$A_0 \sim A_{15}$, R/$\overline{W}$, $\overline{TxSTB}$	t_{AD}	Fig. 6	–	–	270	–	–	180	–	–	160	ns
Address Output Hold Time	$A_0 \sim A_{15}$, R/$\overline{W}$	t_{AHO}	Fig. 6 Fig. 7	30	–	–	20	–	–	20	–	–	ns
	$\overline{TxSTB}$			35	–	–	35	–	–	35	–	–	
Address Three-State Delay Time	$A_0 \sim A_{15}$, R/$\overline{W}$	t_{ATSD}	Fig. 7	–	–	270	–	–	270	–	–	270	ns
Address Three-State Recovery Time	$A_0 \sim A_{15}$, R/$\overline{W}$	t_{ATSR}	Fig. 7	–	–	270	–	–	270	–	–	270	ns
Delay Time	$\overline{DRQH}$, $\overline{DRQT}$	t_{DQD}	Fig. 5	–	–	375	–	–	250	–	–	210	ns
TxAK Delay Time	ϕ_2 DMA Rising Edge	t_{TKD1}	Fig. 5	–	–	400	–	–	310	–	–	250	ns
	DGRNT Rising Edge	t_{TKD2}	Fig. 8	–	–	190	–	–	160	–	–	150	
$\overline{IRQ}$/$\overline{DEND}$ Delay Time	ϕ_2 DMA Falling Edge	t_{DED1}	Fig. 6	–	–	300	–	–	250	–	–	210	ns
	DGRNT Rising Edge	t_{DED2}	Fig. 8	–	–	190	–	–	160	–	–	125	

3. BUS TIMING

1) READ TIMING

Item		Symbol	Test Condition	HD6844			HD68A44			HD68B44			Unit
				min	typ	max	min	typ	max	min	typ	max	
Address Setup Time	$A_0 \sim A_4$, R/$\overline{W}$, $\overline{CS}$	t_{AS}	Fig. 2	140	–	–	140	–	–	70	–	–	ns
Address Input Hold Time	$A_0 \sim A_4$, R/$\overline{W}$, $\overline{CS}$	t_{AHI}		10	–	–	10	–	–	10	–	–	ns
Data Delay Time	$D_7 \sim D_7$	t_{DDR}		–	–	320	–	–	220	–	–	180	ns
Data Access Time	$D_0 \sim D_7$	t_{ACC}		–	–	460	–	–	360	–	–	280	ns
Data Output Hold Time	$D_0 \sim D_7$	t_{DHR}		10	–	–	10	–	–	10	–	–	ns

2) WRITE TIMING

Item		Symbol	Test Condition	HD6844			HD68A44			HD68B44			Unit
				min	typ	max	min	typ	max	min	typ	max	
Address Setup Time	$A_0 \sim A_4$, R/$\overline{W}$, $\overline{CS}$	t_{AS}	Fig. 2	140	–	–	140	–	–	70	–	–	ns
Address Input Hold Time	$A_0 \sim A_4$, R/$\overline{W}$, $\overline{CS}$	t_{AHI}		10	–	–	10	–	–	10	–	–	ns
Data Setup Time	$D_0 \sim D_7$	t_{DSW}		195	–	–	80	–	–	60	–	–	ns
Data Input Hold Time	$D_0 \sim D_7$	t_{DHW}		10	–	–	10	–	–	10	–	–	ns

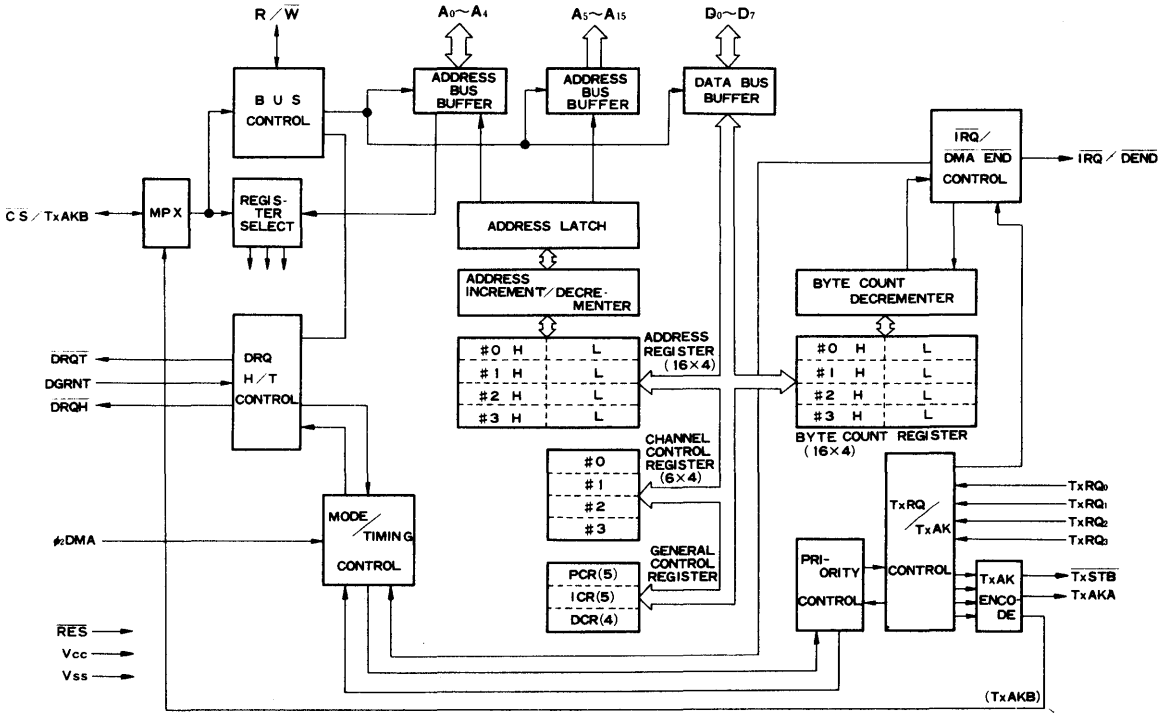


Figure 1 Expanded Block Diagram

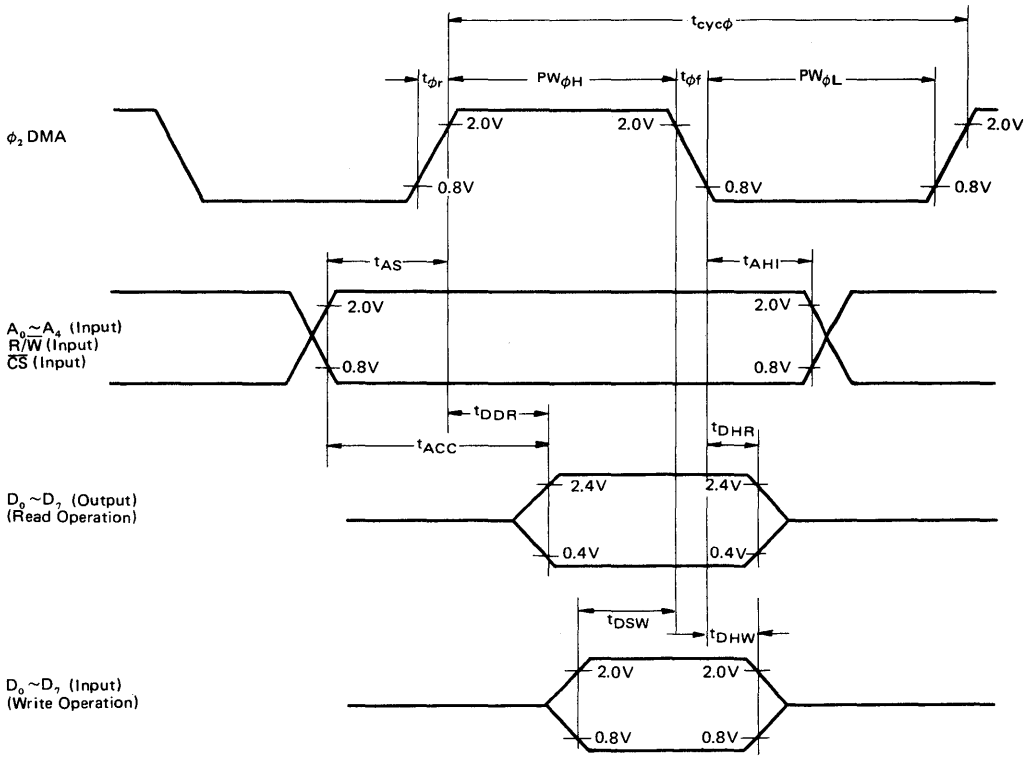


Figure 2 Read/Write Sequence

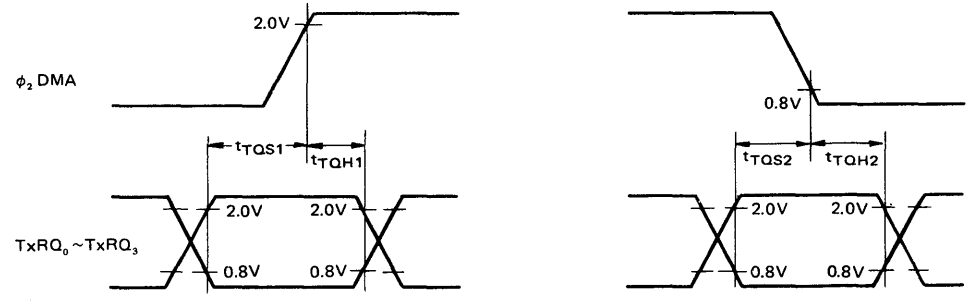


Figure 3 Timing of TxRQ Input

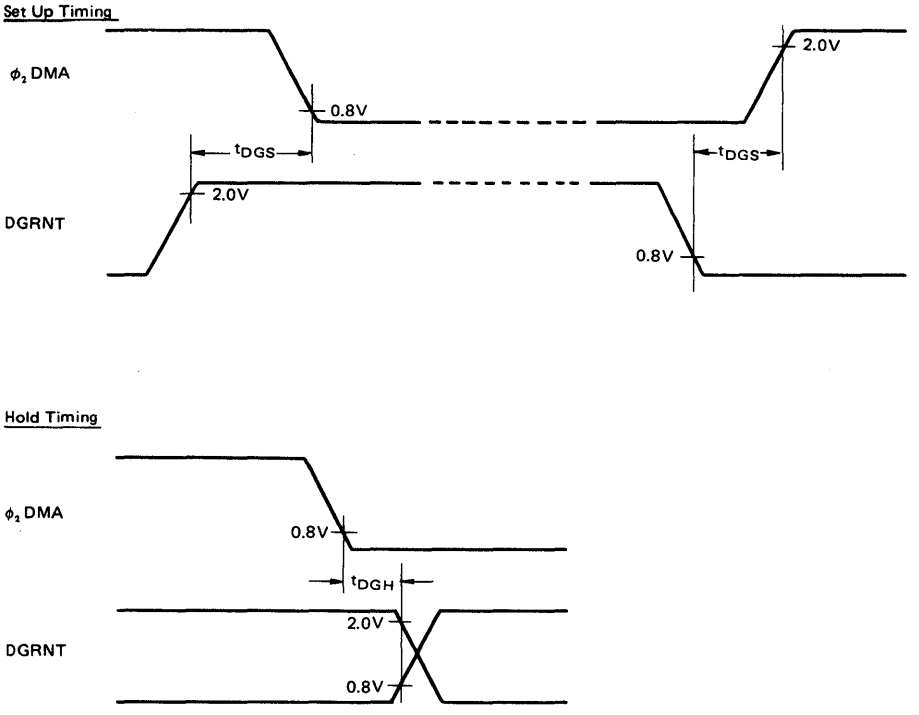


Figure 4 Timing of DGRNT Input

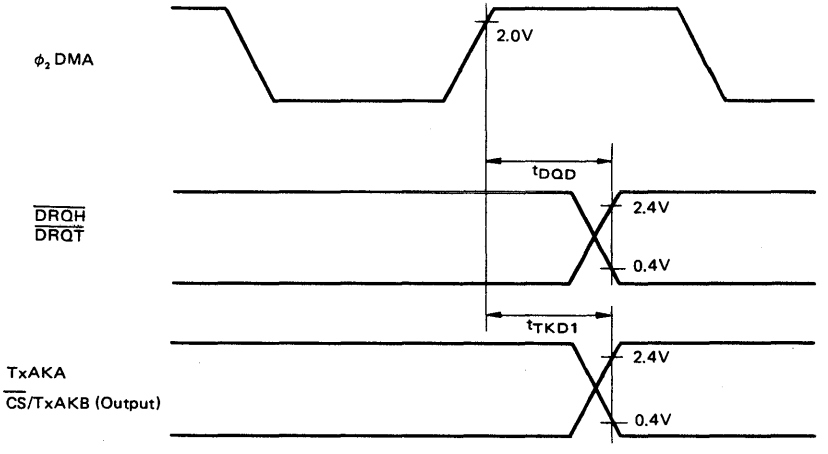


Figure 5 Timing of $\overline{DRQH}$, $\overline{DRQT}$, TxAK Outputs

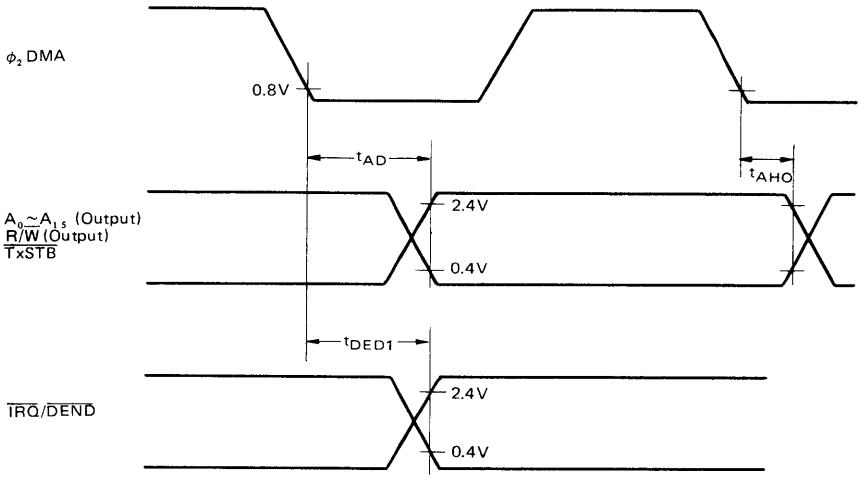
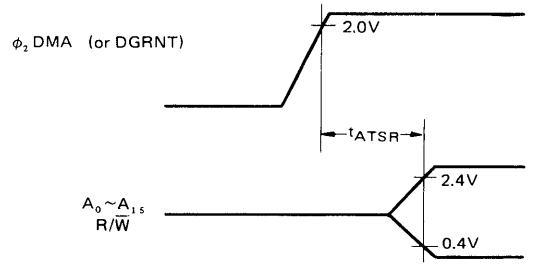


Figure 6 Timing of Address and $\overline{IRQ}/\overline{DEND}$ Outputs

Recovery Time of Address Three-state



Delay Time of Address Three-state

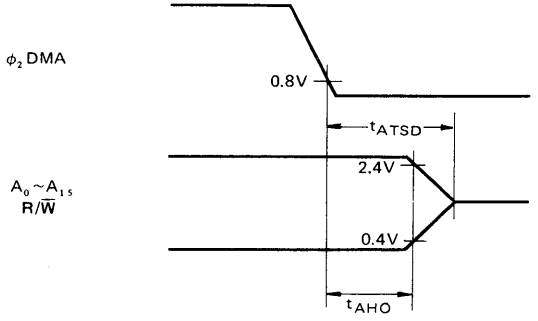


Figure 7 Timing of Address Three-state

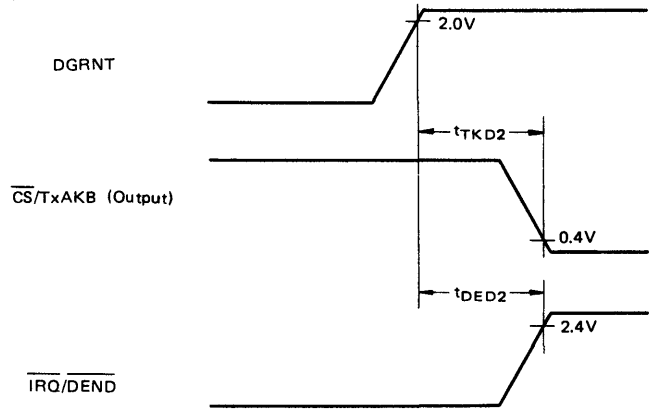
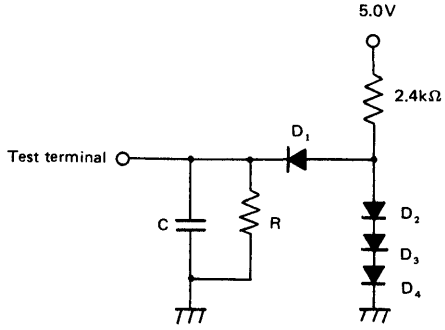


Figure 8 Timing of Synchronous DGRNT Output



Test terminal	C	R
D_0~D_7	130 pF	11 kΩ
A_0~A_{15}, R/$\overline{W}$	90 pF	16 kΩ
$\overline{CS}$/TxAKB	50 pF	24 kΩ
All other outputs	30 pF	24 kΩ

D_1~D_4 : 1S2074Ⓗ or equivalent.

Figure 9 Load Circuit

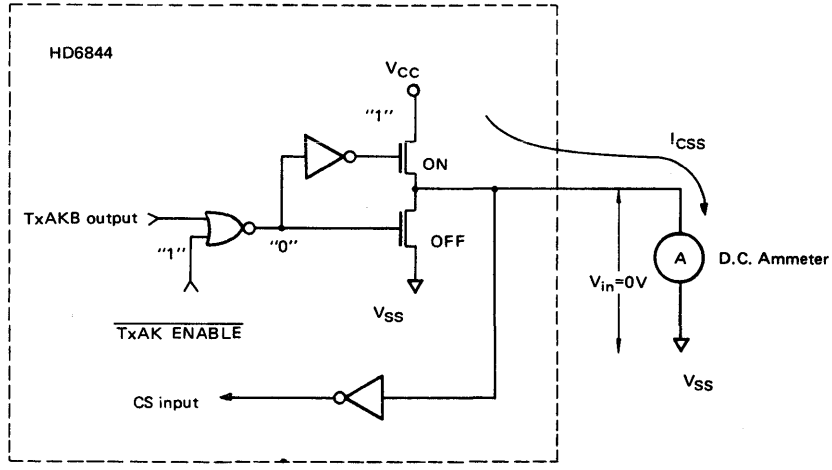


Figure 10 Source Current Measurement Circuit for $\overline{CS}$/TxAKB Terminal

■ DEVICE OPERATION

The DMAC has fifteen addressable registers, eight of them are sixteen bits in length. Each channel has a separate Address Register and a Byte Count Register, each of which is sixteen bits. There are also four Channel Control Registers. The three General Control Registers common to all four channels are the Priority Control Register, the Interrupt Control Register, and the Data Chain Register.

To prepare a channel for DMA, the Address Registers must be loaded with the starting memory address and the Byte Count Register loaded with the number of bytes to be transferred. The bits in the Channel Control Register establish the direction of the transfer, the mode, and the address increment or decrement after each cycle. Each channel can be set for one of three transfer modes: Three-State Control (TSC) Steal, Halt Steal, or Halt Burst. Two read-only status bits in the Channel Control Register indicate when the channel is busy transferring data and when the DMA transfer is completed.

The Priority Control Register enables the transfer requests from the peripheral controllers and establishes either a fixed priority or rotating priority scheme of servicing these requests.

When the DMA transfer for a channel is complete (the Byte Count Register is zero), a $\overline{\text{DMA End}}$ signal is directed to the peripheral controller and an $\overline{\text{IRQ}}$ goes to the MPU. Enabling of these interrupts is done in the interrupt Control Register. The IRQ flag bit is read from this register.

Chaining of data transfers is controlled by the Data Chain Register. When enabled, the contents of the Address and Byte Count Registers for channel #3 are put into the registers of the channel selected for chaining when its Byte Count Register becomes zero. This allows for repetitively reading or writing a block of memory.

During the DMA mode, the DMAC controls the address bus and data bus for the system as well as providing the R/$\overline{\text{W}}$ line and a signal to be used as VMA. When a peripheral device controller desires a DMA transfer, it is requested by a Transfer Request. Assuming this request is enabled and meets the test of highest priority, the DMAC will issue a DMA Request. When the DMAC receives the DMA Grant, it gives a Transfer Acknowledge to the peripheral device controller, at which time the data is transferred. When the channel's Byte Count Register equals zero, the transfer is complete and a $\overline{\text{DMA End}}$ is given to the peripheral device controller, and an $\overline{\text{IRQ}}$ is given to the MPU.

• Initialization

During a power-on sequence, the DMAC is reset via the $\overline{\text{RES}}$ input. All registers, with the exception of the Address and Byte Count Registers, are set to a logic "0" state. This disables all requests and the Data Chain function while masking all interrupts. The Address, Byte Count, and Channel Control Registers must be programmed before the respective transfer request bit is enabled in the Priority Control Register.

• Transfer Modes

There are three ways in which a DMA transfer may be done. The one used is determined by the data transfer rate required, the number of channels attached, and the hardware complexity allowable. Refer to Figures 12, 16 and 17.

Two of the modes, TSC Steal and Halt Steal, are done by cycle-stealing from the MPU. The Three-State Control (TSC) Steal mode is initiated by the DMAC bringing the $\overline{\text{DRQT}}$ line "Low". This line goes to the system clock driver which returns a "High" on DGRNT on the rising edge of the system ϕ_1 clock. The DGRNT signal must cause the address control and data lines to go to the high impedance state. The DMAC now supplies the address from the Address Register of the channel requesting. It also supplies the R/$\overline{\text{W}}$ signal as determined from the Channel Control Register. After one byte is transferred, control is returned to the MPU. This method stretches the ϕ_1 and ϕ_2 clocks while the DMAC uses the memory.

The second method of cycle-stealing is the Halt Steal mode. This method actually halts the MPU instead of stretching the ϕ_1 clock for the transfer period. This mode is initiated by the DMAC bringing the $\overline{\text{DRQH}}$ line "Low". This line connects to the MPU $\overline{\text{HALT}}$ input. The MPU Bus Available (BA) line is the DGRNT input to the DMAC. While the MPU is halted, its Address Bus, Data Bus, and R/$\overline{\text{W}}$ are in the high impedance state. The DMAC now supplies the address and R/$\overline{\text{W}}$ line. After one byte is transferred, the $\overline{\text{HALT}}$ line is returned "High" and the MPU regains control. In this mode, the MPU stops internal activity and is removed from the system while the DMAC uses the memory.

The third mode of transfer is the Halt Burst mode. This mode is similar to the Halt Steal mode, except that the transfer does not stop with one byte. The MPU is halted while an entire block of data is transferred. When the channel's Byte Count Register equals zero, the transfer is complete and control is returned to the MPU. This mode gives the highest data transfer rate, at the expense of the MPU being inactive during the transfer period.

■ INPUT/OUTPUT FUNCTIONS

• DMAC Interface Signals for the MPU

The DMAC interfaces with the HMCS6800 MPU through the eight-bit bidirectional data bus, the $\overline{\text{CS}}$ line, five address lines, an $\overline{\text{IRQ}}$ line, the Read/Write line, and the $\overline{\text{RES}}$ line. These signals, in conjunction with the HMCS6800 VMA output, permit the MPU to have access to the DMAC. Four other lines associated with the MPU and the clock driver are the $\overline{\text{DRQT}}$, $\overline{\text{DRQH}}$, DGRNT, and the ϕ_2 DMA.

Bidirectional Data (D_0~D_7)

The Bidirectional Data lines (D_0~D_7) allow for data transfer between the DMAC and the MPU. The data bus output drivers are three-state devices that remain in the high impedance state except when the MPU performs DMAC read operations.

Chip Select/Transfer Acknowledge B ($\overline{\text{CS}}$/T x AKB)

This line is multiplexed, serving both as an input and an output. $\overline{\text{CS}}$/TxAKB is an output in the four-channel mode during the DMA transfer. At all other times, it is a high impedance TTL compatible input used to address the DMAC. The DMAC is selected when $\overline{\text{CS}}$/TxAKB is "Low". VMA must be used in generating this input to insure that false selects will not occur. Transfers of data to and from the DMAC are then performed under the control of the ϕ2 DMA, Read/Write, and A_0~A_4 address lines. In the four-channel mode when TxAKB is needed, the $\overline{\text{CS}}$ gate must have an open-collector output (a pull-up resistor should not be used). In the two-channel mode, $\overline{\text{CS}}$/TxAKB is always an input.

Address Lines A_0~A_4 (A_0~A_4)

Address lines A_0~A_4 are both input and output lines. In the MPU mode, these are high impedance inputs used to address the DMAC registers. In the DMA mode, these lines are outputs which are set to the contents of the Address Register of the channel being processed.

Interrupt Request/DMA End ($\overline{IRQ}/\overline{DEND}$)

$\overline{IRQ}/\overline{DEND}$ is a TTL compatible, active "Low" output that is used to interrupt the MPU and to signal the peripheral controller that the data block transfer has ended. If the Interrupt has been enabled, the $\overline{IRQ}/\overline{DEND}$ line will go "Low" after the last DMA cycle of a transfer. An open collector gate must be connected to DGRNT and $\overline{IRQ}/\overline{DEND}$ to prevent false interrupts from the DEND signal when interrupts are not enabled. Refer to the section of "DMA End Control".

Read/Write (R/$\overline{W}$)

Read/Write is a TTL compatible line that is a high impedance input in the MPU mode and an output in the DMA mode. In the MPU mode, it is used to control the direction of data flow through the DMAC's input/output data bus interface. When Read/Write is "High" (MPU read cycle) and the chip is selected, DMAC data output buffers are turned on and a selected register is read. When it is "Low", the DMAC output drivers are turned off and the MPU writes into a selected register.

In the DMA mode, Read/Write is an output to drive the memory and peripheral controllers. Its state is determined by bit 0 of the Channel Control Register for the channel being serviced. When Read/Write is "High", the memory is read and the data from the memory is written into the peripheral controller. When it is "Low", the peripheral controller is read and its data stored in the memory. In the DMA mode, the DMAC data buffers are off.

Reset ($\overline{RES}$)

The $\overline{RES}$ input provides a means of resetting the DMAC from an external source. In the "Low" state, the $\overline{RES}$ input causes all registers, with the exception of the Address and Byte Count Registers, to be reset to the logic "0" state. This disables all transfer requests, masks all interrupts, disables the data chain function, and puts each Channel Control Register into the condition of memory write, Halt Steal transfer mode, and address increment.

- **Transfer Signals to the MPU**

Two DMA request output lines and a DMA Grant input line, together with the system clock, synchronize the DMAC with the MPU system.

DMA Request Three-State Control Steal ($\overline{DRQT}$)

This active "Low" output requests a DMA transfer for a channel configured for the TSC Steal transfer mode. This line is connected to the system clock driver, requesting a ϕ_1 clock stretch. It will remain in the "Low" state until the transfer has begun.

DMA Request Halt ($\overline{DRQH}$)

This active "Low" output requests a DMA transfer for a channel programmed for the Halt Steal or Halt Burst mode transfer. This line is connected directly to the MPU $\overline{HALT}$ input and remains "Low" until the last byte has begun to be transferred.

DMA Grant (DGRNT)

This is a high impedance input to the DMAC, giving it control of the system busses. For the TSC Steal mode, the signal comes from the system clock drive circuit (DMA Grant), indicating that the clock is being stretched. For either of the Halt modes, this signal is the Bus Available from the MPU, indicating that the MPU has halted and turned control of its busses over to the DMAC. For a design involving TSC Steal and Halt mode transfers, this input must be the OR of the clock driven DMA Grant and the MPU BA.

ϕ_2 DMA

Transferring in and out of the DMAC registers, sampling of channel request lines and gating of other control signals to the system is done internally in conjunction with the ϕ_2 DMA high impedance input. This input must be the system memory clock (non-stretched ϕ_2 clock).

- **Transfer Signals From the Peripheral Controller**

Transfer Request ($TxRQ_0 \sim TxRQ_3$)

Each of the four channels has its own high impedance input request for transfer line. The peripheral controller requests a transfer by setting its TxRQ line "High" (a logic "1"). The lines are sampled according to the priority and enabling established in the Priority Control Register. In the Steal mode and the first byte of the Halt Burst mode, the TxRQ signals are tested on the positive edge of ϕ_2 DMA and the highest priority channel is strobed. Once strobed, the TxRQs are not tested until that channel's data transfer is finished. In the succeeding bytes of the Halt Burst mode transfer, the TxRQ is tested on the negative edge of ϕ_2 DMA, and data is transferred on the next ϕ_2 DMA cycle if TxRQ is "High".

- **Transfer Signals to the Peripheral Controller**

Two encoded lines select the channel to be serviced. A strobe line acknowledges the request and performs the transfer. The $\overline{DEND}$ line signals to the peripheral controller that the DMA transfer is completed.

Transfer Acknowledge A (T x AKA)

The Transfer Acknowledge A (TxAKA) is a TTL compatible output used in conjunction with the $\overline{CS}$/TxAKB line to select the channel to be strobed for transfer and to give the DMA End Signal. In the two-channel mode, only TxAKA is used to select channel 0 or channel 1, and $\overline{CS}$/TxAKB is always an input.

Chip Select/Transfer Acknowledge B ($\overline{CS}$/TxAKB)

In the DMA mode, this dual purpose line is encoded together with TxAKA to select the channel being serviced. Table 1 shows the encoding order.

Table 1 Encoding Order

$\overline{CS}$/TxAKB	TxAKA	Channel #
0	0	0
0	1	1
1	0	2
1	1	3

Transfer Strobe ($\overline{TxSTB}$)

The $\overline{TxSTB}$ causes acknowledgement to be given to the peripheral controller and transfers the data to or from the memory. This line is also intended to be the VMA signal for the system in the DMA mode. In a one-channel system, $\overline{TxSTB}$ may be inverted and run to the peripheral controller's Acknowledge input. In a two or four-channel system, $\overline{TxSTB}$ enables the decode of TxAKA and $\overline{CS}$/TxAKB to select the device controller to be acknowledged.

Interrupt Request/DMA End ($\overline{IRQ}/\overline{DEND}$)

In the DMA mode, this dual purpose line is "Low" for the last byte of transfer, indicating a DMA End. This occurs when the Byte Count register decrements to zero.

This line, through the decode of TxAKA and $\overline{CS}$/TxAKB, can be used to strobe a DMA End to each device controller.

- **Address Lines to the Memory**

Address Lines ($A_0 \sim A_{15}$)

These output lines are in the high impedance state during the MPU mode. In the DMA mode, these lines are outputs which are set to the contents of the Address Register of the channel being processed.

■ THE DMAC REGISTERS

The HD6844 (DMAC) has Address Register (ADR), Byte Count Register (BCR), Channel Control Register (CHCR), and General Control Register (GCR).

General Control Register (GCR) is composed of Priority Control Register (PCR) that controls priority among the channels, Interrupt Control Register (ICR) that controls interrupt and Data Chain Control Register (DCR) that controls data chain function. Refer to Table 2 and Figure 1.

These are Read/Write registers and MPU can exchange the data with DMAC when $\overline{CS}$ is at "Low" level. $A_0 \sim A_4$ specifies the address of the registers. How to specify the registers is shown in Table 2.

2-byte ADR and BCR can be read or written by one instruction, using 2-byte instruction of the MPU.

- **Function of Internal Registers**

ADR (Address Register)

Each channel has 16-bit Address Register. Initial address of memory used for DMA transfer is programmed to this register. The contents of ADR are output to address bus ($A_0 \sim A_{15}$) during DMA transfer operation. When 1-byte transfer has completed, the 16-bit address is incremented or decremented by one.

The address which the MPU reads out is the renewed one, that is, the memory address for the next transfer. When 1-block transfer has completed, final memory address +1 is read out.

Table 2 Internal Registers of the DMAC

Register Name	Symbol	Channel	Address Bus Signal					Address (Hexadecimal)
			A_4	A_3	A_2	A_1	A_0	
Address Register	ADRH	0	0	0	0	0	0	00
	ADRL	0	0	0	0	0	1	01
Byte Count Register	BCRH	0	0	0	0	1	0	02
	BCRL	0	0	0	0	1	1	03
Address Register	ADRH	1	0	0	1	0	0	04
	ADRL	1	0	0	1	0	1	05
Byte Count Register	BCRH	1	0	0	1	1	0	06
	BCRL	1	0	0	1	1	1	07
Address Register	ADRH	2	0	1	0	0	0	08
	ADRL	2	0	1	0	0	1	09
Byte Count Register	BCRH	2	0	1	0	1	0	0A
	BCRL	2	0	1	0	1	1	0B
Address Register	ADRH	3	0	1	1	0	0	0C
	ADRL	3	0	1	1	0	1	0D
Byte Count Register	BCRH	3	0	1	1	1	0	0E
	BCRL	3	0	1	1	1	1	0F
Channel Control Register	CHCR	0	1	0	0	0	0	10
	CHCR	1	1	0	0	0	1	11
	CHCR	2	1	0	0	1	0	12
	CHCR	3	1	0	0	1	1	13
Priority Control Register	PCR	–	1	0	1	0	0	14
Interrupt Control Register	ICR	–	1	0	1	0	1	15
Data Chain Control Register	DCR	–	1	0	1	1	0	16

(NOTE) 1) All the registers can be accessed by R/W operation. Unused bit of the register is read out "0".
2) H/L of ADR and BCR means the higher (H) 8 bits/the lower (L) 8 bits of a 16-bit register.
3) 16-bit ADR and BCR can be read or written by one instruction, using MPU's 2-byte LOAD/STORE instruction.

Register Address
e.g. LDX $\$ \underset{\text{Address of DMAC}}{*\,*}\,0C$ ..... [(ADRH 3) → (Index Register H)] [(ADRL 3) → (Index Register L)]

BCR (Byte Count Register)

Each channel has a 16-bit Byte Count Register. Number of DMA transfer words is programmed into this register. The content of the Byte Count Register is decremented by one everytime one-byte transfer has completed. When it becomes "0", $\overline{DEND}$ output goes "Low" level and informs I/O controller of the end of one-block DMA transfer. When $\overline{IRQ}$ is not masked, $\overline{IRQ}$ output goes "Low" level and MPU is interrupted to be informed of the end of DMA transfer. Moreover, $\overline{IRQ}$ and $\overline{DEND}$ signals are output, multiplexed with $\overline{IRQ}/\overline{DEND}$ pin.

CHCR (Channel Control Register)

Each channel has Channel Control Register. This register is used to program the control information of its corresponding channel. Structure of CHCR is shown in Table 3.

(1) R/W Control (specifies the direction of transfer)
Bit – CHCR Bit 0

This bit controls the direction of DMA transfer. When it is at "1", R/$\overline{W}$ signal of DMAC goes "High" level during DMA transfer operation. This means to read out memory and write into I/O controller, that is, data is transferred from memory to I/O controller.

When it is at "0", R/$\overline{W}$ output goes "Low" level and data is transferred from I/O controller to memory.

Table 3 Bit Structure of CHCR (Channel Control Register)

Bit No.	Name	R/W	Function	
			"1"	"0"
0	R/W	R/W	Transfer from memory to I/O controller (R/$\overline{W}$ output = "High")	Transfer from I/O controller to memory (R/$\overline{W}$ output = "Low")
1	Burst/Cycle Steal	R/W	Burst Mode	Cycle Steal Mode*
2	TSC/HALT	R/W	TSC Mode	HALT Mode*
3	Address down/up	R/W	Address: -1	Address: +1
4	Not used	–	–	–
5	Not used	–	–	–
6	Busy/Ready Flag	R	Busy (DMA Transfer Operation)	Ready (No DMA Transfer Operation)
7	DEND Flag	R	DMA End & Interrupt	No Interrupt

* Burst•TSC mode is prohibited.

Note that during DMA tranfer operation, the function of R/$\overline{W}$ signal is accommodated to the memory Read/Write operation. Therefore, on the side of I/O device during DMA transfer operation, R/$\overline{W}$ input should be interpreted in inverse of the MPU Read/Write. That is, data should be output when R/$\overline{W}$ input is at "Low" level (In the case of MPU's read operation, I/O device outputs the data when it is at "High" level).

This arises from that during DMA transfer operation, I/O side performs data transfer independently instead of MPU. Moreover, such family LSI as HD6843 (FDC), etc. has this function and R/$\overline{W}$ signal is automatically interpreted inversely.

(2) Burst/Cycle Steal Bit – CHCR Bit 1

This bit is used to decide that DMA transfer should be performed in burst mode or cycle steal mode. When it is at "1", it specifies burst mode. That is, once DMA transfer is performed, MPU remains stopped until one-block data transfer is completed.

When this bit is "0", it specifies cycle steal mode. That is, everytime one-byte transfer has completed, MPU takes back the bus control, and DMA transfer and MPU operation are performed in time sharing.

(NOTE) Only in the case of HALT mode, burst mode can be specified. When TSC mode is specified, burst mode cannot be specified.

(3) TSC/HALT Mode Bit – CHCR Bit 2

This bit is used to decide that DMA transfer should be performed by using MPU's TSC function or HALT function. When it is at "0", $\overline{DRQH}$ output of DMAC is connected to $\overline{HALT}$ input of MPU and DMA transfer is performed by using MPU's HALT function.

When it is at "1", DMA transfer is performed by using MPU's TSC function. That is, $\overline{DRQT}$ output is connected to HD26501 (CPG) and MPU's clock ϕ_1 is extended. Then MPU's TSC input becomes "High" level and the bus gets into high impedance state to perform DMA transfer.

(4) Address down/up Bit – CHCR Bit 3

This bit is used to decide that the address of memory region used for DMA transfer should be renewed up (increment of address) or down (decrement of address). When it is at "1", the address is decremented by one after one-byte transfer. When it is at "0", the address is incremented by one.

(5) Busy/Ready Flag Bit – CHCR Bit 6

This bit is a status flag to indicate whether its corresponding channel is performing DMA transfer or not. (READ only)

When it receives the first TxRQ of its corresponding channel, it goes to "1". When one-block transfer is completed and BCR becomes "0", it is reset to "0".

(6) DEND Flag Bit – CHCR Bit 7

This bit is an interrupt flag to indicate that one-block DMA transfer of its corresponding channel has completed.

(READ only).

When one-block transfer of its corresponding channel is completed and BCR becomes "0", it goes to "1". As soon as this flag is read out, i.e. CHCR of this channel is read out, it is reset to "0".

Moreover, this bit is connected to $\overline{IRQ}$ output. When it is at "1" and IRQ enable bit (within ICR register described later) is at "1", $\overline{IRQ}$ output goes "Low" level.

PCR (Priority Control Register)

Priority Control Register is a 5-bit register to decide the operation mode of priority control circuit. Structure of PCR is shown in Table 4.

Table 4 Bit Structure of PCR (Priority Control Register)

Bit No.	Name	R/W	Function	
			"1"	"0"
0	TxRQ Enable #0 ($TxEN_0$)	R/W	TxRQ of Channel 0 is accepted.	TxRQ of Channel 0 is not accepted.
1	TxRQ Enable #1 ($TxEN_1$)	R/W	TxRQ of Channel 1 is accepted.	TxRQ of Channel 1 is not accepted.
2	TxRQ Enable #2 ($TxEN_2$)	R/W	TxRQ of Channel 2 is accepted.	TxRQ of Channel 2 is not accepted.
3	TxRQ Enable #3 ($TxEN_3$)	R/W	TxRQ of Channel 3 is accepted.	TxRQ of Channel 3 is not accepted.
4	Not used	–	–	–
5		–	–	–
6		–	–	–
7	Rotate Control	R/W	Rotate Mode	The order of priority is fixed at numerical order.

(1) TxRQ Enable Bit ($TxEN_0$~$TxEN_3$) – PCR Bit 0~3

Each channel has this TxRQ Enable bit. When it is at "1", TxRQ input of its corresponding channel is accepted to perform DMA transfer. When it goes to "0", TxRQ of its corresponding channel is masked not to be received and TxAK is not output. During DMA transfer operation, when this bit goes to "0" before BCR becomes "0", following TxRQ input is not accepted and DMA transfer is interrupted. Then contents of ADR and BCR remain unchanged. When it rises to "1" again, DMA transfer is reopened. Therefore, in the case of cycle steal DMA, it is possible for the program to change the priority of the specific channel temporarily by manipulating this bit.

(2) Rotate Control Bit – PCR Bit 7

When this bit is at "0", the order of priority among DMA channels is fixed at numerical order. That is, Channel 0 is given a first priority and then is followed by Channel $1 \rightarrow 2 \rightarrow 3$.

When this bit is at "1", priority control is due to rotate mode. That is, the channel that ended in the first time is given a first priority and the channel ended in the last time is controlled to be given a last priority.

ICR (Interrupt Control Register)

Interrupt Control Register is a 5-bit register to control $\overline{IRQ}$ output. Its structure is shown in Table 5.

(1) IRQ Enable Bit – ICR Bit 0~3

Each channel has IRQ Enable Bit. When this bit is at "1" and DEND Flag of its corresponding channel is set to "1", $\overline{IRQ}$ output goes "Low" level. But when it is at "0", $\overline{IRQ}$ output is masked not to be output even if DEND Flag is set to "1".

These bits enable to control to output only a necessary channel to $\overline{IRQ}$.

(2) IRQ Flag – ICR Bit 7

This is a read-only bit and the status of $\overline{IRQ}$ output is directly reflected on it. That is, when $\overline{IRQ}$ output goes to "Low" level, it becomes "1".

$\overline{IRQ}$ output of DMAC is output as logical OR of 4-channel DEND Flag according to the following equation.

$$IRQ = (DEND_0 \cdot IRQ\ Enable_0) + (DEND_1 \cdot IRQ\ Enable_1) + (DEND_2 \cdot IRQ\ Enable_2) + (DEND_3 \cdot IRQ\ Enable_3)$$

DCR (Data Chain Control Register)

Data Chain Control Register is a 4-bit register and three of those bits are used to control data chain function. Remaining one bit is used to specify 2-channel/4-channel mode.

Structure of DCR is shown in Table 6.

(1) Data Chain Enable Bit – DCR Bit 0

When this bit is at "1", data chain function of DMAC is enabled. That is, when DMA transfer of a specified channel has completed and BCR goes to "0", the contents of ADR and BCR of Channel #3 are automatically transferred to ADR and BCR of the specified channel.

(2) Data Chain Channel Bit – DCR Bit 1~2

These bits are used to specify which channel should be used for the data chain. How to specify the channel is shown in Table 7. Data Chain Channel bit specifies the channel to which data should be transfered from Channel #3. Channel #3 contains the data for replacement. Channel #3 is fixed and cannot be changed.

(3) 2/4-channel Mode Bit – DCR Bit 3

This bit has no relation to the data chain function.

It is used to specify whether $\overline{CS}$/TxAKB is used for only input pin or I/O pin. When this bit is "0", $\overline{CS}$/TxAKB becomes $\overline{CS}$ input pin in 2-channel mode since TxAKB output is not necessary for application up to 2-channel.

When this bit is "1", $\overline{CS}$/TxAKB becomes I/O pin in 4-channel mode (See Fig. 11).

Table 5 ICR (Interrupt Control Register)

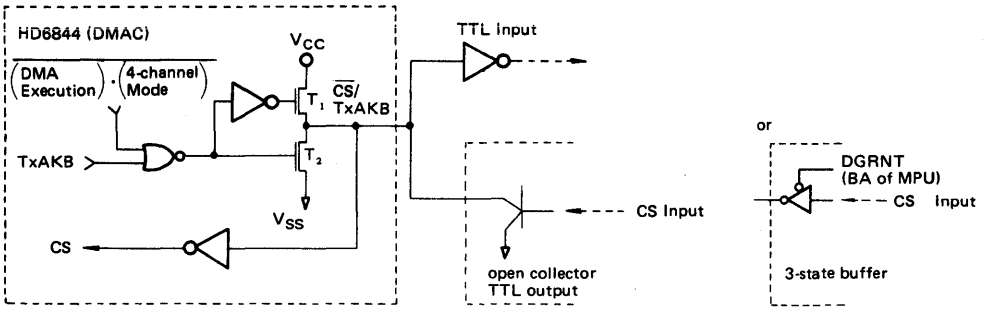
Bit No.	Name	R/W	Function	
			"1"	"0"
0	IRQ Enable #0	R/W	$\overline{IRQ}$ of Channel 0 is able to be output.	$\overline{IRQ}$ output of Channel 0 is masked.
1	IRQ Enable #1	R/W	$\overline{IRQ}$ of Channel 1 is able to be output.	$\overline{IRQ}$ output of Channel 1 is masked.
2	IRQ Enable #2	R/W	$\overline{IRQ}$ of Channel 2 is able to be output.	$\overline{IRQ}$ output of Channel 2 is masked.
3	IRQ Enable #3	R/W	$\overline{IRQ}$ of Channel 3 is able to be output.	$\overline{IRQ}$ output of Channel 3 is masked.
4	Not used	–	–	–
5		–	–	–
6		–	–	–
7	IRQ Flag	R	$\overline{IRQ}$ output "Low"	$\overline{IRQ}$ output "High" (off state)

Table 6 Bit Structure of DCR (Data Chain Control Register)

Bit No.	Name	R/W	Function	
			"1"	"0"
0	Data Chain Enable	R/W	Data Chain is performed.	Data Chain is not performed.
1	Data Chain Channel	R/W	The channel which performs Data Chain is specified. (The channel where contents of ADR and BCR of Channel #3 are loaded.)	
2		R/W		
3	2/4-Channel Mode	R/W	4-Channel Mode ($\overline{CS}$/TxAKB is I/O pin.)	2-Channel Mode ($\overline{CS}$/TxAKB is designated to only input pin.)
4	Not used	–	–	–
5		–	–	–
6		–	–	–
7		–	–	–

Table 7 How to specify Data Chain Channel

DCR Bit 1	DCR Bit 2	Specified Channel
0	0	Channel #0
1	0	Channel #1
0	1	Channel #2
1	1	–



In $\overline{CS}$ input mode T1 turns ON and T2 turns OFF. T1 functions as pull-up resistance.

Figure 11 How to Use CS/TxAKB Pin

■ OPERATION OF THE DMAC

• Transfer Mode of the DMAC

There are three DMA transfer modes such as HALT Cycle Steal, HALT Burst and TSC Cycle Steal. Operation in each mode is explained in the following.

HALT Cycle Steal Mode

This is a basic DMA transfer mode. In this mode, everytime 1-byte transfer has completed, MPU takes back the bus control and executes Instruction cycle. That is, DMA transfer and MPU operation are performed in time sharing.

Timing chart is shown in Fig. 12 and flow chart is shown in Fig. 13. Procedure of transfer operation is the following. (No. ① ~ ⑪ in Fig. 12 correspond to the following items.)

① $TxRQ_0 \sim TxRQ_3$ input is checked at the rising edge of $\phi 2$DMA. When it is at "High" level, it gets into the following operation.
② $\overline{DRQH}$="Low" is output and MPU is requested to stop its operation.
③ TxAKA is driven (Level output).
④ MPU stops its operation and DMAC waits until DGRNT goes to "High" level.
⑤ When DGRNT goes to "High" level, DMAC drives TxAKB, $A_0 \sim A_{15}$ and R/$\overline{W}$ lines.
⑥ $\overline{TxSTB}$ is given to perform DMA transfer.
⑦ Address is incremented by one and number of transfer words is decremented by one.
⑧ When $\overline{DRQH}$ rises to "High" level, MPU gets into Instruction Cycle again.
⑨ TxRQ falls to "Low" level.
⑩ $A_0 \sim A_{15}$ and R/$\overline{W}$ get into high impedance state again.
⑪ DGRNT falls to "Low" level.

[Note] $TxRQ_0 \sim TxRQ_3$ input is, in principle as shown in Fig. 12, set to "High" on account of I/O request. When $\overline{TxSTB}$ of the DMAC is driven, it is reset to "Low". Take care not to be against this principle, or the following states may happen.

(1) In the case where TxRQ becomes "High", but it is reset to "Low" before DGRNT becomes "High". In this case, the DMAC is in the wait state without sending out $\overline{TxSTB}$ until TxRQ rises to "High" again. As $\overline{DRQH}$ remains "Low" the MPU is forced to be stopped, and the system is in dead lock state until TxRQ rises to "High" again (Fig. 14).

(2) In the case where TxRQ is not reset to "Low" though $\overline{TxSTB}$ has been driven.
In this case, unless TxRQ returns to "Low" by the time ϕ_2DMA rises after $\overline{TxSTB}$ has risen to "High", it is considered as a new I/O request, which leads the above-mentioned operation ①,② ⟶ .
If TxRQ falls to "Low" immediately after that, the same state as (1) happens (Fig. 15).

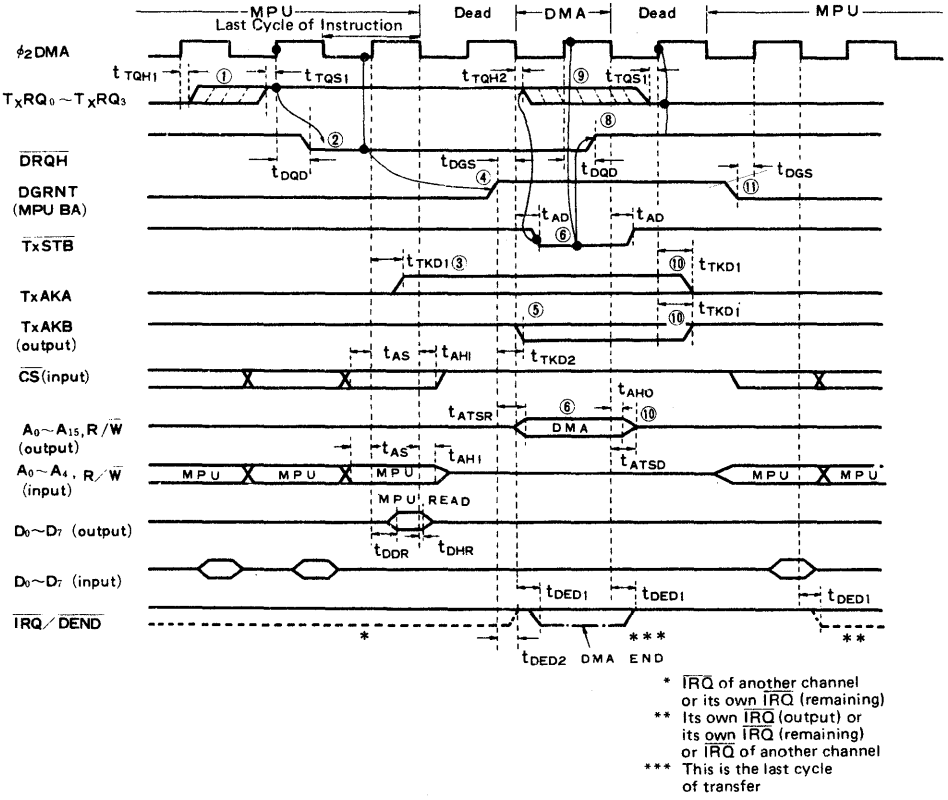


Figure 12 HALT Cycle Steal Mode

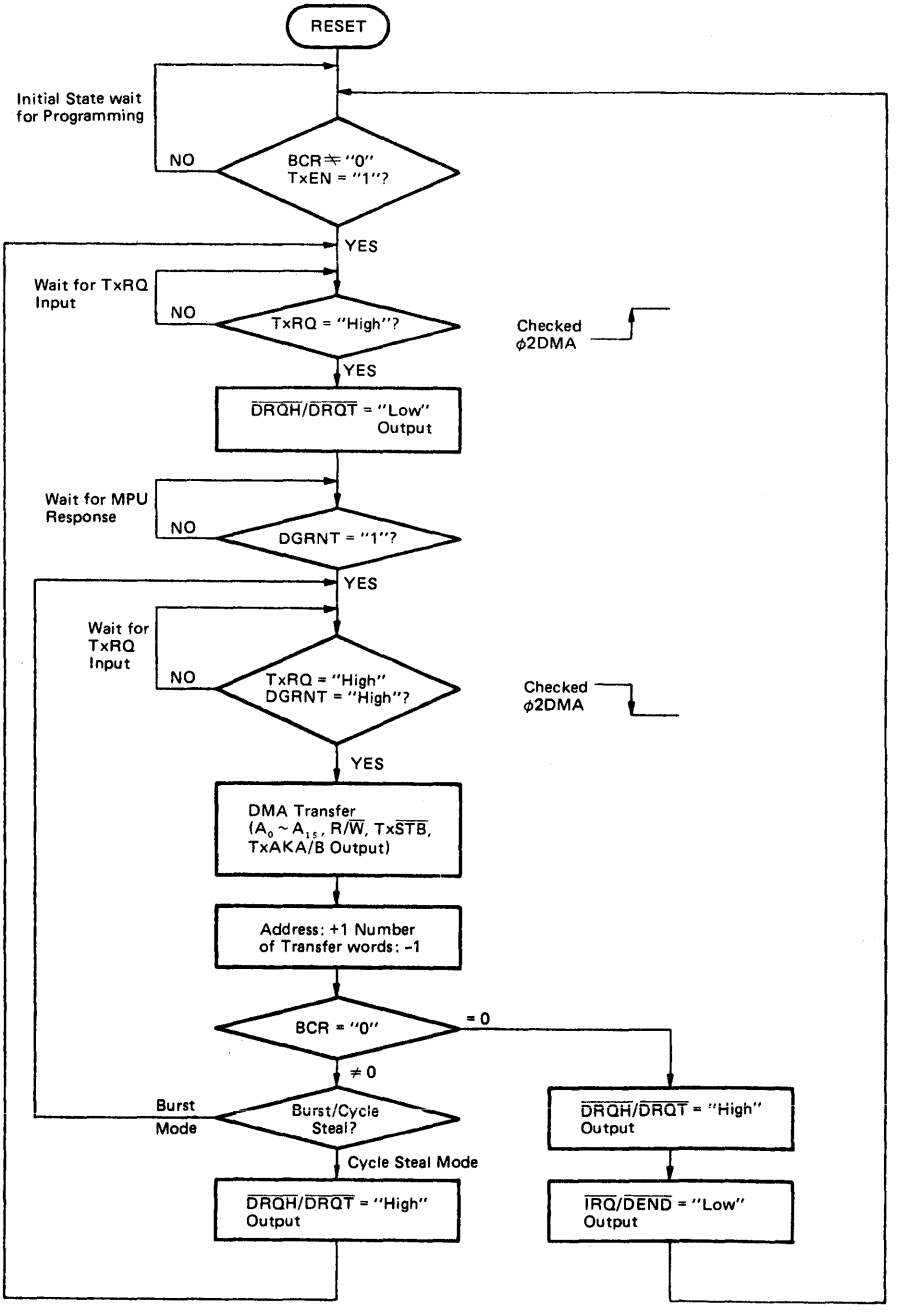


Figure 13 Flow Chart of DMAC Operation

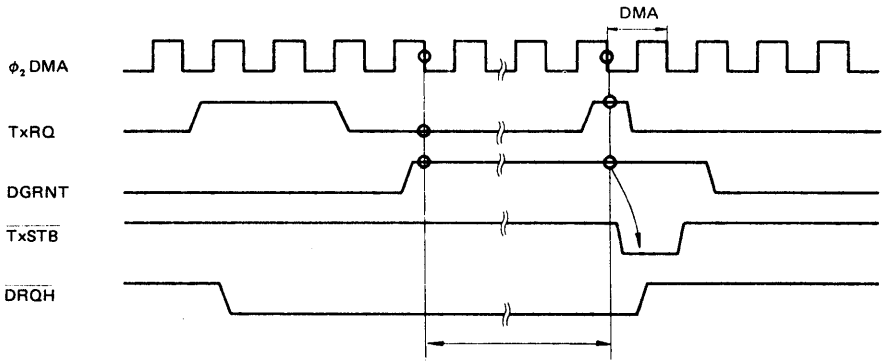


Figure 14 Extraordinary TxRQ Input (1)

(In the case where TxRQ is reset to "Low" before the transfer)

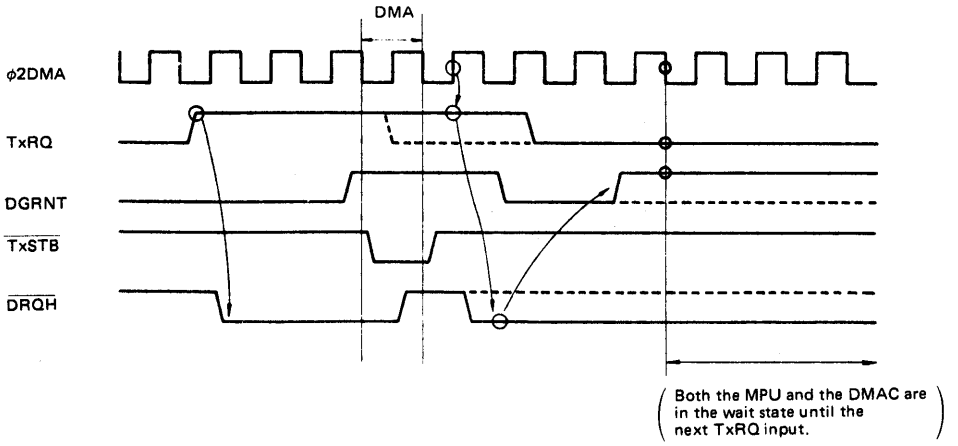


Figure 15 Extraordinary TxRQ Input (2)

(In the case where TxRQ doesn't fall to "Low" after the transfer has been completed.)

HALT Burst Mode

In the case of cycle steal mode, MPU gets into Instruction Cycle everytime 1-byte transfer has completed. But in the case of burst mode, MPU remains stopped until 1-block transfer is finished. That is, $\overline{DRQH}$ continues to be output "Low" level until BCR becomes "0".

Its timing chart and flow chart are shown in Fig. 16 and Fig. 13 respectively. Procedure of transfer is the following (No. ① ~ ⑭ in Fig. 16 correspond to the following items).

① TxRQ input is checked at the rising edge of ϕ_2 DMA. When it is at "High" level, it gest into the following operation.
② $\overline{DRQH}$="Low" level is given and MPU is requested to stop its operation.
③ TxAKA is driven.
④ MPU stops and DMAC waits for DGRNT rising "High" level.
⑤ When DGRNT rises "High" level, DMAC drives TxAKB, A_0 ~A_{15}, and R/$\overline{W}$ lines.
⑥ $\overline{TxSTB}$ is sent out to perform DMA transfer.
⑦ Address is incremented by one and number of transfer words is decremented by one.
⑧ TxRQ falls to "Low" level.
⑨ When number of transfer words is 0, from ⑪ to ⑭ operations are performed.
⑩ When BCR is not "0", TxRQ is checked at the falling edge of ϕ_2 DMA.When TxRQ is at "High" level, DMA transfer is performed through ⑥ ~ ⑧ again. When TxRQ is not at "High" level, DMAC waits for becoming "High" level.
⑪ $\overline{IRQ/DEND}$ output goes to "Low" level.
⑫ $\overline{DRQH}$ output rises to "High" level and MPU gets into Instruction Cycle again.
⑬ A_0~A_{15} and R/$\overline{W}$ get into high impedance state.
⑭ DGRNT falls to "Low" level.

The transfer of the first byte (①~ ⑥) is performed in the same way as that in cycle steal mode. But in the second-byte and subsequent transfer, TxRQ is checked at the falling edge of ϕ_2 DMA and if TxRQ is at "High" level, DMA transfer is performed at the following cycle. Therefore, a high-speed response which is not exceeding 2-clock-cycle is feasible.

In burst mode, TxRQ should be also, in principle, set to "High" when I/O request is asserted, and reset to "Low" when $\overline{TxSTB}$ goes to "Low". If TxRQ is asserted as level input without being reset, DMA transfer is performed at all cycles of ϕ_2 DMA since TxRQ is always at "High" level at the falling edge of ϕ_2 DMA. Its example is shown in the second-byte and the third-byte transfer in Fig. 16.

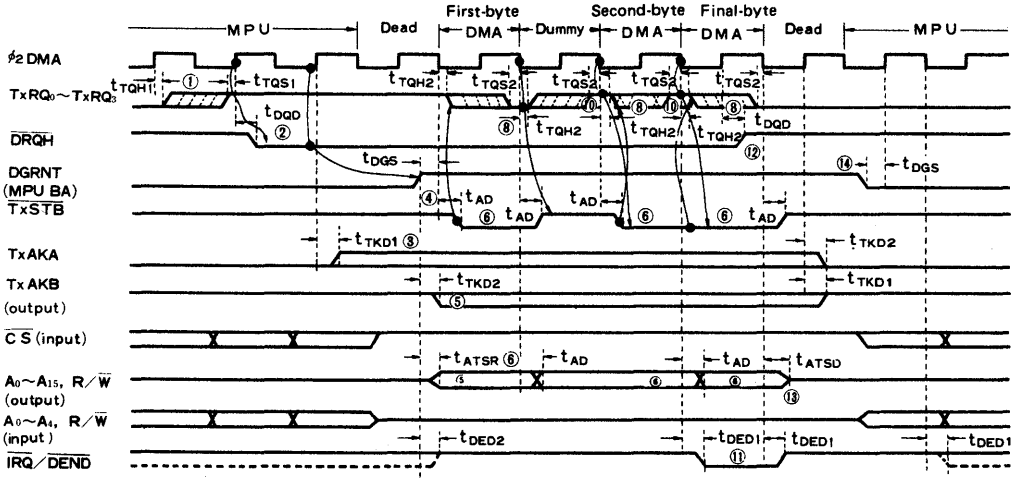


Figure 16 HALT Burst Mode

TSC Cycle Steal Mode

In the above-mentioned modes, DMA is performed by using the HALT function of the MPU. In TSC cycle steal mode, DMA is performed by using the TSC function of the MPU.

Its timing chart and flow chart are shown in Fig. 17 and Fig. 13 respectively.

Basic operation of the DMAC is the same as that in HALT cycle steal mode, but the detailed timing is different. The difference is explained in the following.

(1) $\overline{DRQT}$ is used instead of $\overline{DRQH}$.
(2) $\overline{DRQT}$ is connected to the CPG instead of the MPU. When $\overline{DRQT}$ goes to "Low", MPU (ϕ_1, ϕ_2) clock gets into an extended state.
(3) DGRNT is connected to DGRNT of the CPG. DGRNT timing is different from that in HALT mode. (DGRNT is connected to BA of the MPU.) (The response time is quick. It is set at half-clock before BA and is reset at 1-clock before BA.)

More detailed timing of DGRNT of the CPG shall be shown in the manual of the CPG.

In TSC mode, there isn't a burst mode. Because the MPU clock cannot be extended for a long time. When TSC mode is specified, $\overline{DRQT}$ returns to "High" and the MPU gets into the Instruction Cycle everytime 1-byte transfer has finished.

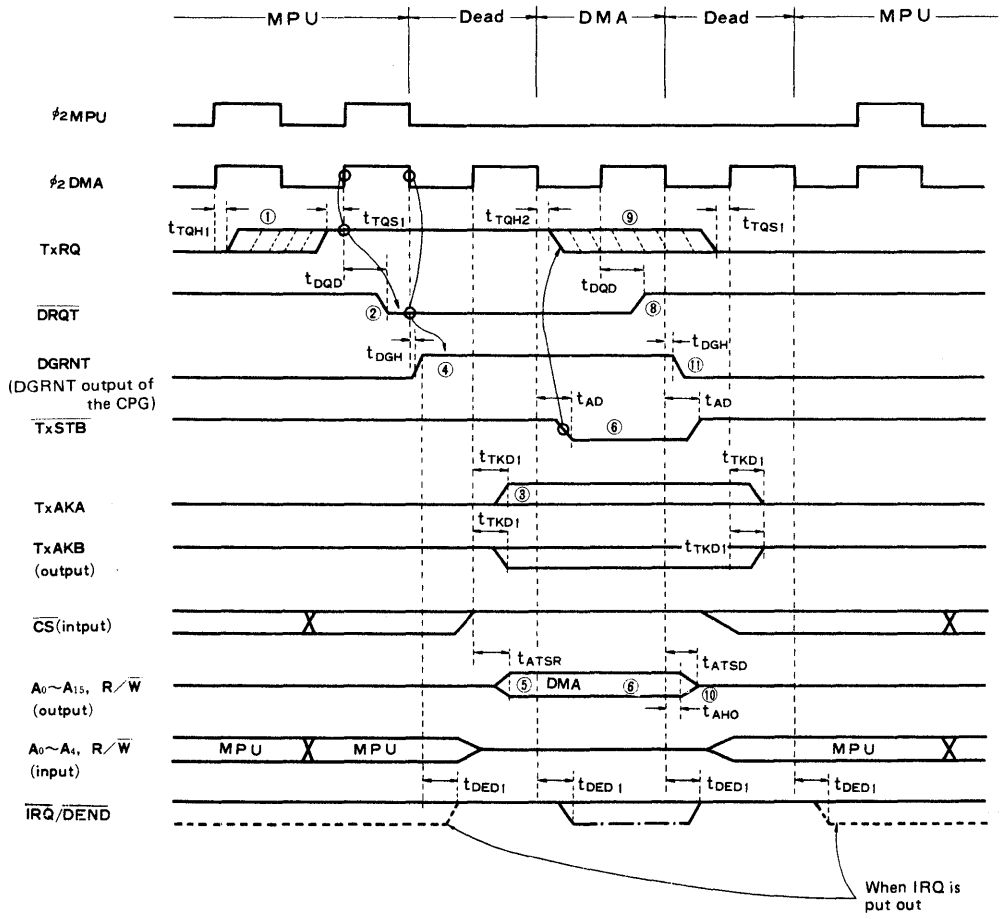


Figure 17 TSC Cycle Steal Mode

- **Priority Control**

Basic priority Control

There are two kinds of the DMAC priority control function. One is to mask TxRQ on each channel by TxRQ Enable bit. The other is priority-order-determining-circuit which the DMAC has as a hardware.

Moreover, the priority-order-determining-circuit has two operation modes (the rotate mode and the normal mode).

Structure of the priority control circuit is shown in Fig. 18. As shown in Fig. 18, TxRQ of the channel whose TxRQ Enable bit is at "1" level becomes an input of the priority-order-determining-circuit. Then it is checked whether TxRQ is at "High" level or not.

(Note) In this case, ZERO flag needs to be at "1" level. ZERO flag will be described later.

If one of $TxRQ_0 \sim TxRQ_3$ is at "High" level, its channel is selected, being given a first priority. Then it is latched by an executing-channel-number-latch-circuit to perform DMA transfer. Once an executing channel is determined and latched, it is unchanged until its DMA transfer has been completed. That is, the channel number strobe signal doesn't go to "1" and the contents of the channel-number-latch-circuit are unchanged. In the cycle steal mode, the channel is fixed until 1-byte transfer has completed. In the burst mode, it is fixed until BCR becomes "0".

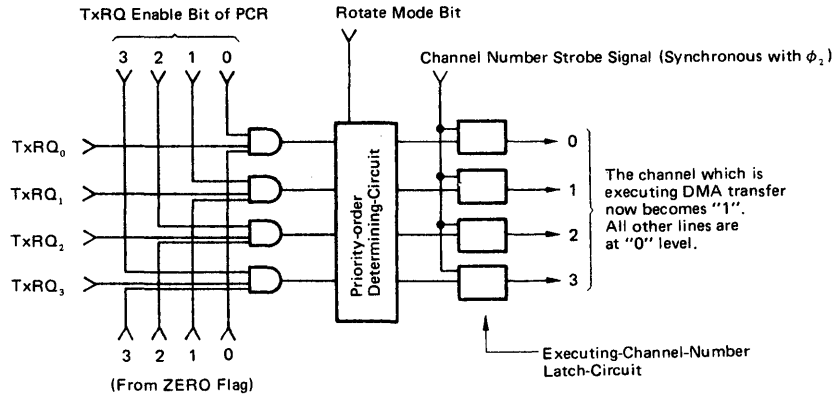


Figure 18 Structure of Priority Control Circuit

Therefore, once a long-period DMA transfer of a channel is performed in the burst mode, other channels need to wait until it has completed even if they have higher priority than the channel. Take much care to this point in designing response time to TxRQ of DMA channel.

(Note) As explained above, TxRQ input is latched internally. So once it is accepted and latched, the channel number cannot be changed even though it returns to "Low". But as explained in HALT Cycle Steal Mode, DMA transfer is not performed unless TxRQ rises to "High" again.

Strobe timing of executing-channel-number-latch-circuit is shown in Fig. 19.

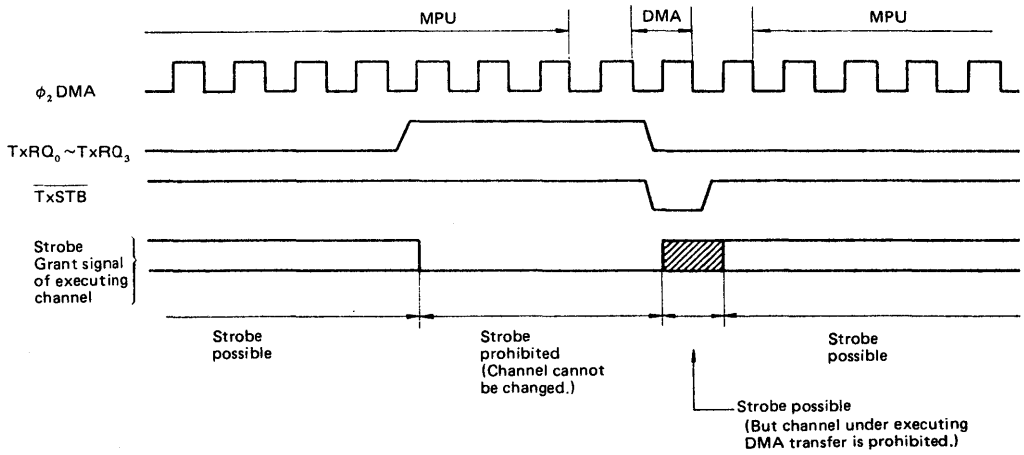


Figure 19 Strobe Timing of Executing-Channel-Number-Latch-Circuit (the cycle steal mode)

But, as shown in Fig. 19, only the channel under executing DMA transfer is prohibited to accept TxRQ during DMA transfer operation, in order that one more byte transfer may not be performed when the reset timing of TxRQ is delayed. Strobe timing in the burst mode is shown in Fig. 20.

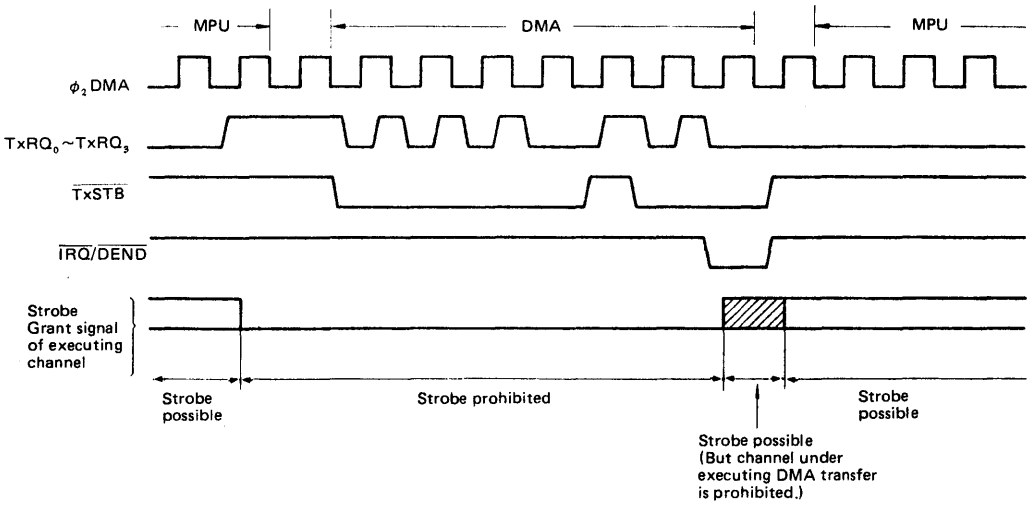


Figure 20 Strobe Timing of Executing-Channel-Number-Latch-Circuit (the burst mode)

Rotate Mode

There are two operation modes in priority-order-determining circuit. These are Normal Mode and Rotate Mode. In the normal mode, the order of priority is fixed at numerical order. (Channel 0 is given a first priority and then is followed by Channel 1 → 2 → 3.) In the rotate mode, the channel next to the channel with which DMA was executed in the last sequence, is given a first priority and the channel in the last sequence is given a last priority. But immediately after it gets into the reset state, the order of priority is the following: Channel 0 → 1 → 2 → 3.

An example of the rotate mode is shown in Fig. 21.

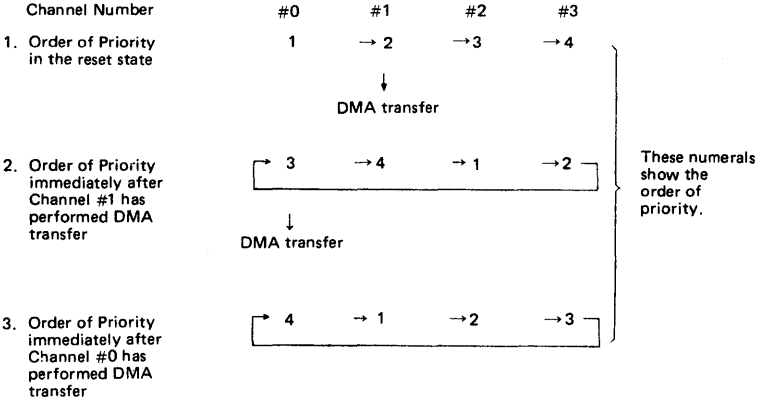
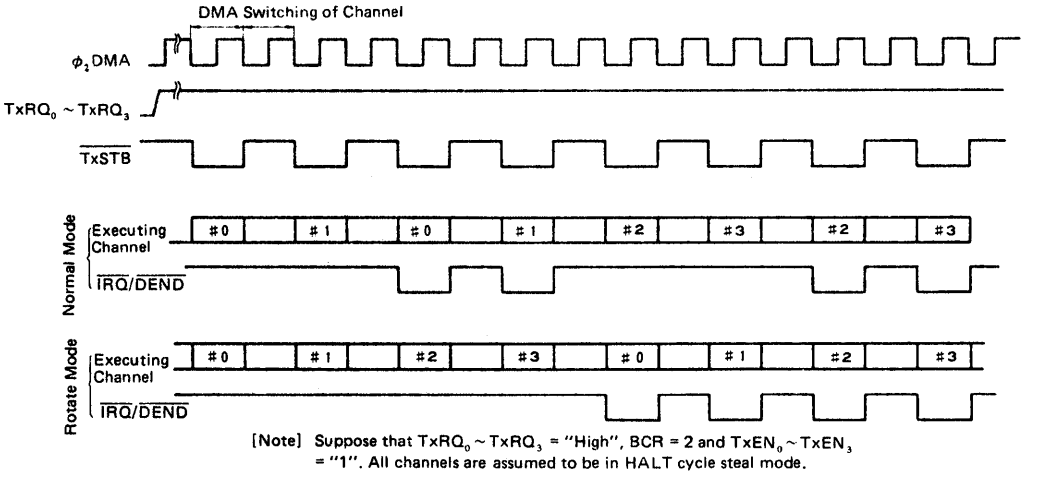


Figure 21 Example of Operation in the Rotate Mode

Next, Fig. 22 shows an example of the difference between the operation in the rotate mode and that in the normal mode. In this example, TxRQ of all channels is always at "High" level. Moreover, BCR=2 and TxEN=1 are assumed. As a transfer mode, HALT cycle steal mode is used.



[Note] Suppose that TxRQ$_0$ ~ TxRQ$_3$ = "High", BCR = 2 and TxEN$_0$ ~ TxEN$_3$ = "1". All channels are assumed to be in HALT cycle steal mode.

Figure 22 Difference between the operation in the rotate mode and that in the normal mode

The reason why the order of priority is not #0 → #0 → #1 → #1 → ——— in the normal mode is that during DMA transfer operation, TxRQ of an executing channel is prohibited from being accepted.

DMA Operation Timing with priority control

When more than 2 channels perform DMA transfer in parallel, the abovementioned priority-order-determining-circuit is used to determine the priority. The channel with lower priority waits until the channel with higher priority completes the transfer. Then it gets into DMA transfer operation. In this case, The following combinations of transfer modes are conceivable.

(1) From HALT mode to HALT mode (Fig. 23)
(2) From TSC mode to TSC mode (Fig. 24)
(3) From HALT mode to TSC mode
(4) From TSC mode to HALT mode } (Fig. 25)

In changing from HALT mode to HALT mode, only one dead cycle is intervened. That is, even in the cycle steal mode, DMA transfer of the next channel is performed without returning the bus control to the MPU ($\overline{DRQH}$ remains "Low").

In changing from TSC mode to TSC mode, DMA transfer of the next channel is performed, after returning the bus control to MPU for one cycle.

In the case of HALT → HALT, it doesn't return the bus control to MPU in order not to increase the response time of DMA transfer and dead cycles of the system.

On the other hand, in the case of TSC → TSC mode, same mean cannot be applicable because MPU clock cannot remain stopped for a long time as in the case of HALT mode.

Both in the case of HALT → TSC mode and in the case of TSC → HALT mode, DMA operation timing is based on the same idea as the above two kinds of mode change. (In detail, see Fig. 25).

The timing in the case where the next byte is transfered without changing the channel is shown in Fig. 26. This is the case of HALT → HALT mode. In this case, the bus control returns to MPU, before the next byte is transfered. In the case of TSC → TSC mode, its timing is almost the same as than in Fig. 24, that is, after 1-byte transfer has completed, MPU executes the Instruction Cycle for one clock and then DMAC executes 1-byte transfer again.

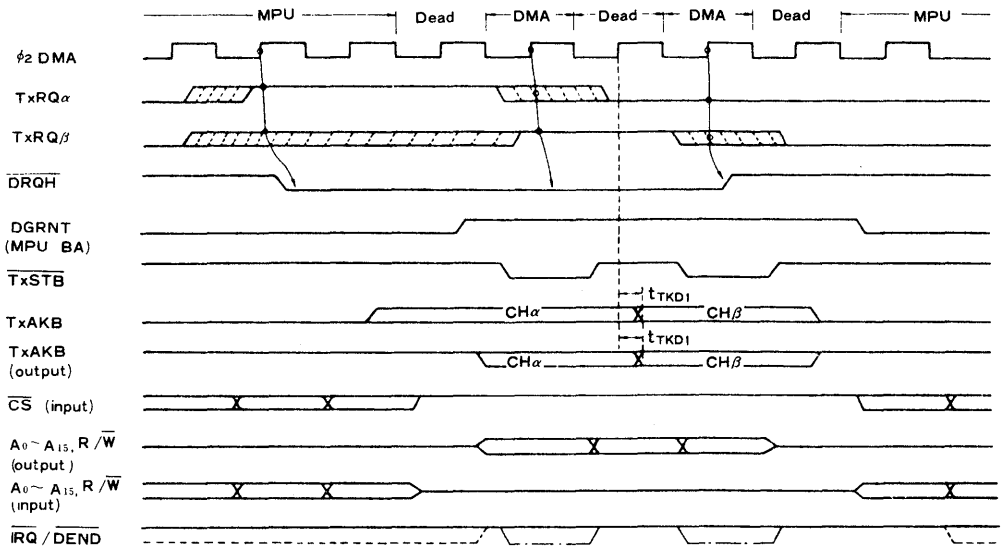


Figure 23 Channel Change (HALT Mode → HALT Mode)

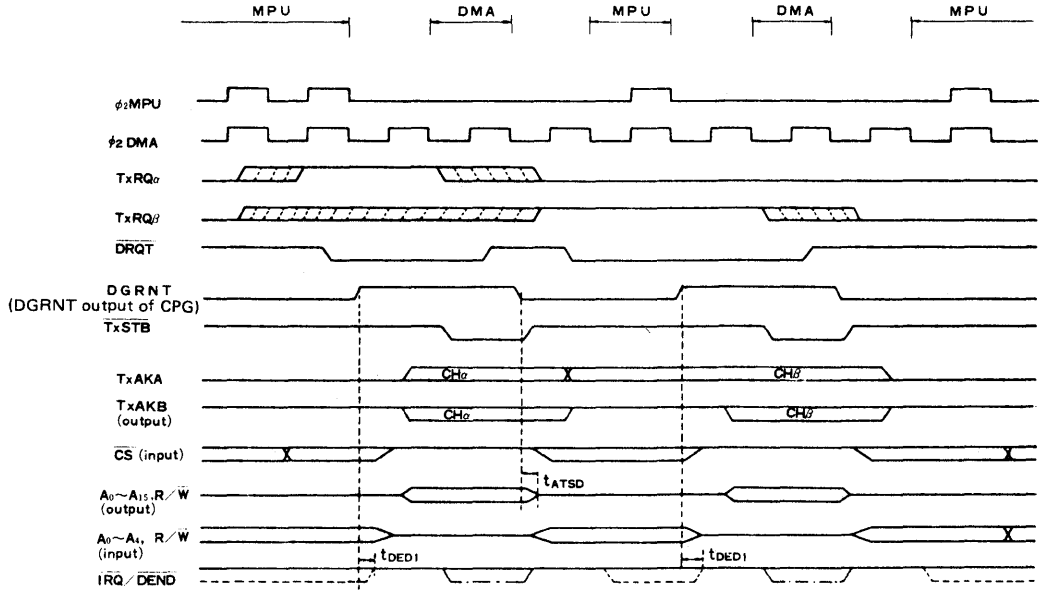


Figure 24 Channel Change (TSC Mode → TSC Mode)

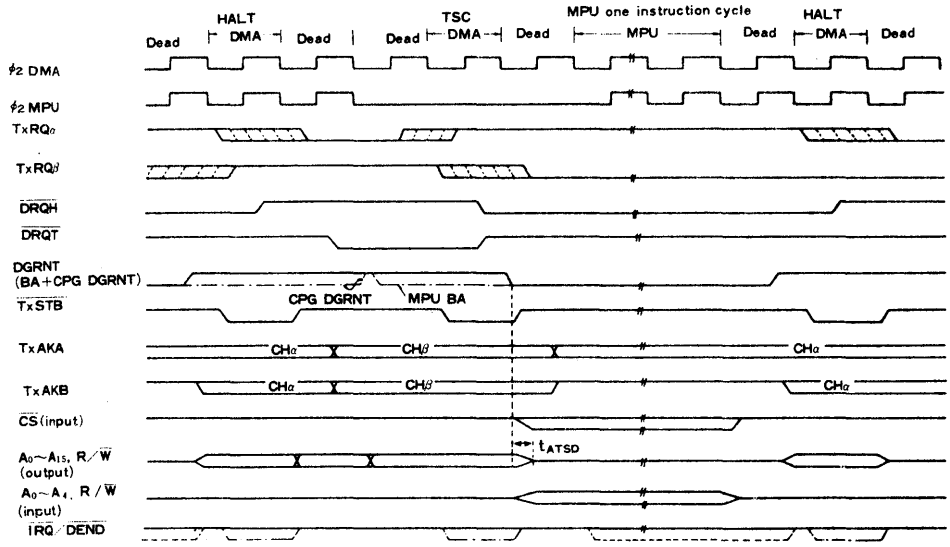
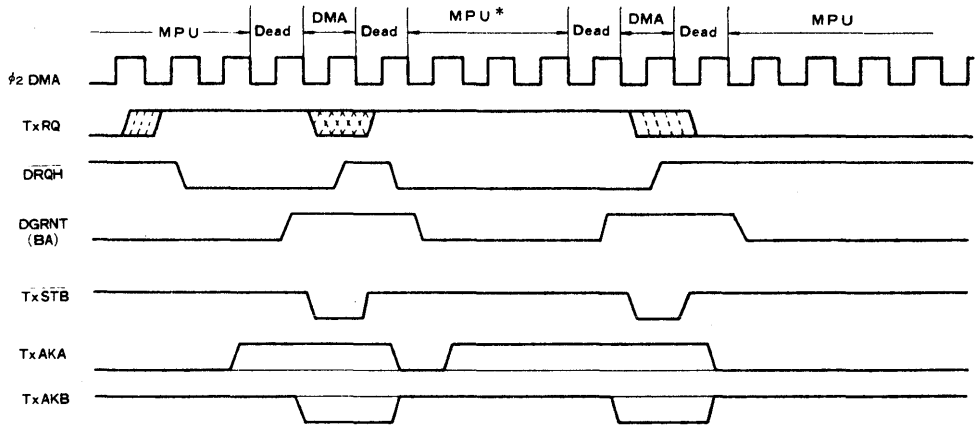


Figure 25 Channel Change (HALT Mode → TSC Mode → HALT Mode)



* Executing Period of One Instruction

Figure 26 Successive 2-byte Transfer of One Channel (HALT Cycle Steal Mode)
HALT → HALT (by one channel)

- **Status Flag**

DMAC has BUSY Flag, DEND Flag and ZERO Flag on each channel. The former two of these flags can be read out by MPU, but ZERO Flag cannot be read out. Set and reset timing of each flag are shown in Fig. 27.

BUSY/READY Flag

This flag is set to "1" when it accepts the first-byte TxRQ of its corresponding channel. After 1-block transfer has completed and BCR becomes "0", it is reset to "0". Therefore, while this flag is "1", that is, its corresponding channel is being used, the next block transfer cannot be performed.

DEND Flag

This is the interrupt flag to indicate the end of DMA transfer of its corresponding channel. After 1-block transfer has completed and BCR becomes "0", this flag is set to "1". This flag is reset to "0" immediately after the Channel Control Register having this flag is read out.

ZERO Flag

This is the internal flag to indicate whether the data stored in the BCR is "0" or not (It cannot be read out).

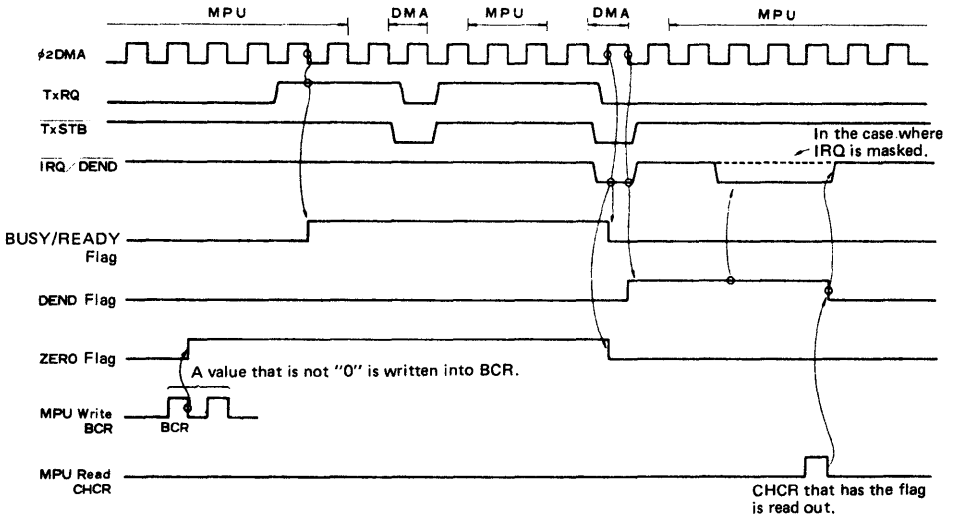


Figure 27 Timing of Status Flag (Suppose that BCR is 2 in the initial state)

When BCR is "0", ZERO Flag is "0". When BCR is not "0", it is "1".

In the reset state, this flag is "0". If data that is not "0" is written into BCR, this flag is set to "1". When BCR becomes "0" after 1-block data transfer has completed, or MPU writes "0" into BCR, this flag is reset to "0".

The function of ZERO Flag is to prohibit accepting TxRQ of its corresponding channel while this flag is "0" (that is, BCR is "0") (See Fig. 18). While ZERO Flag is "0", TxRQ is not accepted even if TxEN is "1". This function avoids an false operation even if "High" input is provided to TxRQ before the initialization of the register.

When $\overline{RES}$ pin goes to "Low", this flag becomes "0", but the number in BCR is not reset to "0". Therefore, the state of this flag and BCR are not the same. In this case new data should be written into BCR (Then ZERO Flag becomes "1").

- **DMA End Control**

Function of $\overline{IRQ}/\overline{DEND}$ Pin

DMAC has $\overline{IRQ}$ output and $\overline{DEND}$ output to perform DMA End Control. These are multiplexed outputs to $\overline{IRQ}/\overline{DEND}$ pin.

The function of $\overline{DEND}$ output is to inform I/O controller of the end of 1-block transfer. After 1-block transfer has been completed and BCR becomes "0", $\overline{DEND}$ output provides "Low" pulse whose cycle is one clock, being synchronous with the final 1-byte data transfer. 4 channels have only one $\overline{DEND}$ output in common, so each channel determines whether $\overline{DEND}$ output is its own output or not, combining with TxAK signal. When $\overline{TxAK}$ of the channel is "Low" and $\overline{DEND}$ is "Low", it shows that the cycle is the last one of DMA (See Fig. 29 and 30).

The function of $\overline{IRQ}$ output is to inform MPU of the end of 1-block transfer by interrupting it. As shown in Fig. 28, $\overline{IRQ}$ output is logical AND-OR of the interrupt flag (DEND Flag) and IRQ Enable bit of each channel.

$\overline{IRQ}$ and $\overline{DEND}$ outputs are multiplexed. $\overline{IRQ}/\overline{DEND}$ pin is used as $\overline{DEND}$ output during DMAC cycle and $\overline{IRQ}$ output during MPU cycle. Moreover, DGRNT signal separates $\overline{DEND}$ and $\overline{IRQ}$ by its "High" or "Low". In detail, see Fig. 29 and Fig. 30.

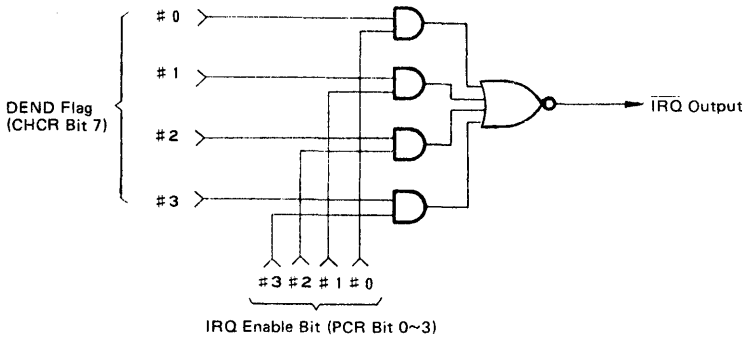


Figure 28 Logic of $\overline{IRQ}$ Output

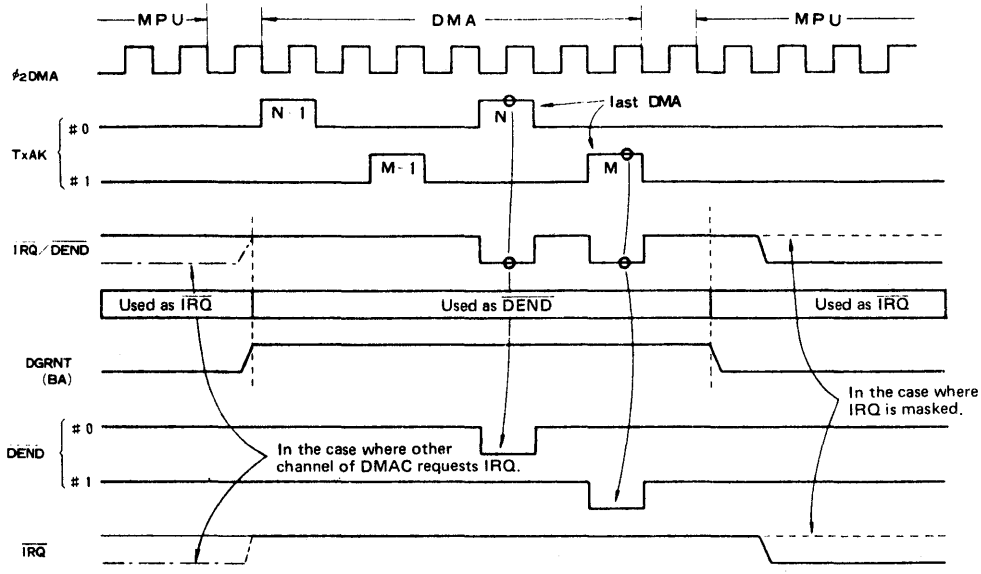


Figure 29 Timing of $\overline{IRQ}$/$\overline{DEND}$ Output

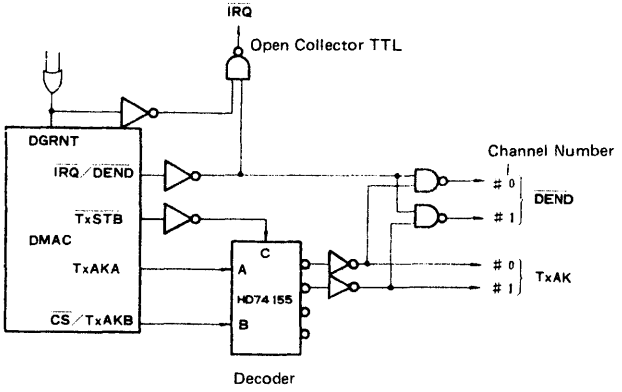


Figure 30 How to Use $\overline{IRQ}/\overline{DEND}$ Output Signal

Unusual DMA End

Following section describes how to terminate or change normal sequence of DMA transfer.

(1) When "0" is written into BCR
When "0" is written into BCR before it becomes "0", subsequent TxRQ are not accepted and this causes the termination of the DMA transfer since the internal ZERO Flag is reset to "0". In this case, note that $\overline{DEND}$ pulse is not provided.

(2) When "1" is written into BCR
When "1", instead of "0", is written into BCR, only the next TxRQ is accepted and 1-byte DMA transfer is performed. In this case, $\overline{DEND}$ pulse is provided, being synchronous with the last transfer.

(3) When another value is written into ADR & BCR during the transfer
When the data in ADR & BCR are changed during the transfer, the following transfer is performed according to the change of the data.

(4) When "0" is written into TxRQ Enable bit
When TxEN is reset to "0" during the transfer, this causes TxRQ comes not to be accepted and the transfer halts. But the state is different from that in the case (1), the number in BCR remains unchanged. Therefore, when TxEN is set to "1" again, the transfer is performed again.

(5) When $\overline{RES}$ pin is set to "Low"
When $\overline{RES}$ is provided during the transfer, the transfer stops.
Then all of the control registers and their internal flags are reset to "0". But the data in ADR & BCR are not reset.

(Supplement)

It is only in the cycle steal mode that DMAC registers such as BCR and ADR can be read or written during the transfer. In the burst mode, it is usually impossible (But special external circuits enable it).

- **Data Chain Function**

The data chain function of DMAC is to transfer the contents of ADR & BCR of Channel #3 to ADR & BCR of a specified channel automatically and renew the data of them after the channel has completed 1-block transfer.

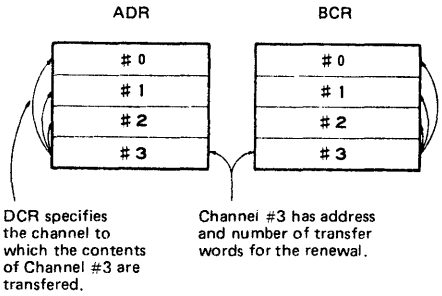


Figure 31 Data Chain Operation

Its detailed timing is shown in Fig. 32 and Fig. 33. As shown in these figures the contents of ADR & BCR of Channel #3 are transfered to the channel during the clock cycle next to the last one of 1-block transfer (which provides DEND pulse). Then $\overline{DRQH}$ or $\overline{DRQT}$ provides "Low" output for one more clock cycle than in the normal case. Therefore, MPU takes back the bus control again 1-clock later than in the normal case, that is, after the data renewal of the specified channel by the data chain from Channel #3.

In the TSC mode, the stretching period of clockϕ_1 is longer than in the normal case.

The contents of ADR & BCR of Channel #3 remain unchanged as long as new data are not written by MPU, even if the data chain is executed.

As for $\overline{DEND}$ output, DEND Flag and BUSY Flag in the case of data chain execution, they function in the same way as in the normal case. They provide $\overline{DEND}$ pulse everytime 1-block transfer has completed, and then DEND Flag is set to "1". Therefore, in the case where more than 3-block data chain is needed, DEND Flag is used for the execution. Its sequence is shown in Fig. 34. First, DEND Flag="1" that shows the end of the first-block data chain is read out. Next, the data of ADR & BCR for the third-block data chain need to be written into Channel #3, in parallel with the execution of the second-block data chain. (This data chain is feasible only in the cycle steal mode.)

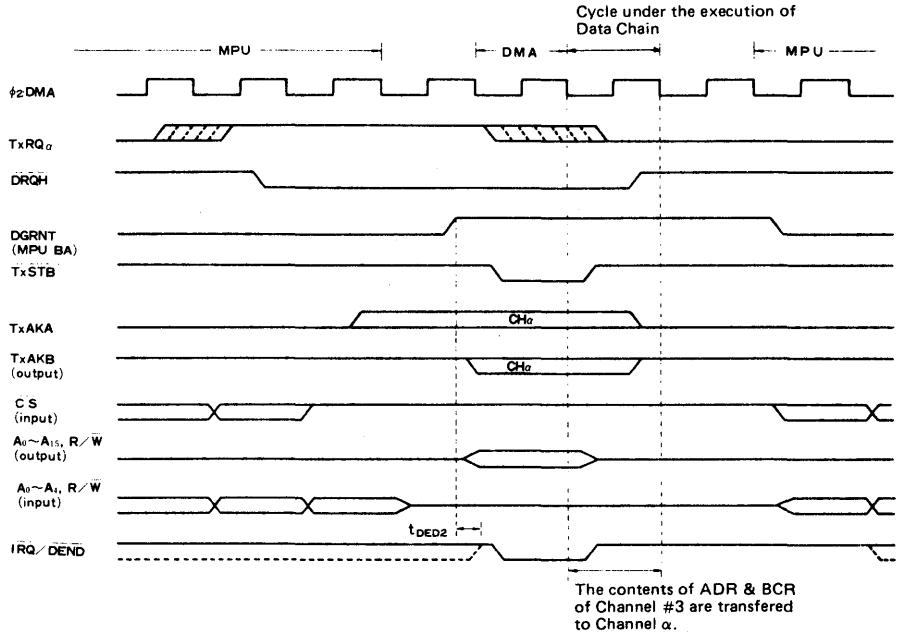


Figure 32 Data Chain Operation (HALT Mode)

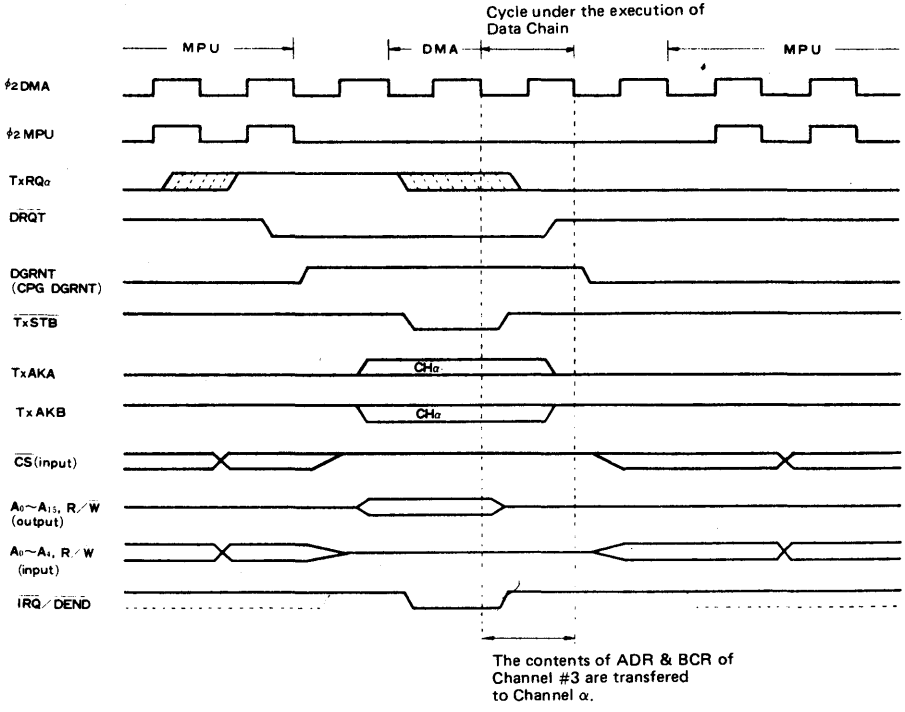


Figure 33 Data Chain Operation (TSC Mode)

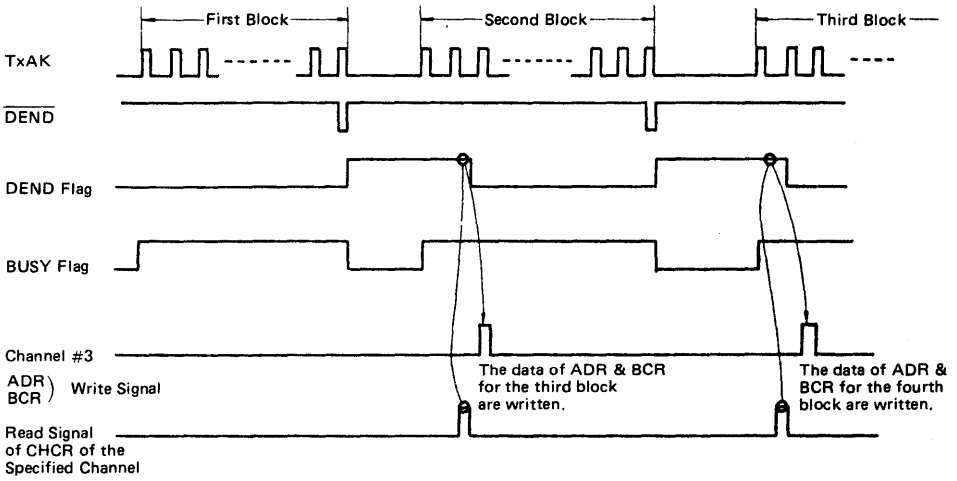


Figure 34 Sequence of More than 3-block Data Chain

■ DMAC PROGRAMMING

Preparation of a channel for a DMA transfer requires:

1) Load the starting address into the Address Register.
2) Load the number of bytes into the Byte Count Register.
3) Program the Channel Control Register for the transfer characteristics: direction (bit 0), mode (bits 1 and 2), and the address update (bit 3).

The channel is now configured. To enable the transfer request, set the appropriate enable bit (bits 0~3) of the Priority Control Register, as well as the Rotate Control bit.

If an interrupt on DMA End is desired, the enable bit (bits 0~3) of the Interrupt Control Register must be set.

If data chaining for the channel is necessary, it is programmed into the Data Chain Register and the appropriate data must be written into the Address and Byte Count Registers for channel #3.

Table 8 DMAC Programming Model

Register	Address (Hex)	Register Content							
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 2
Channel Control	1x*	DMA End Flag (DEND)	Busy/Ready Flag	Not Used	Not Used	Address Up/Down	TSC/ Halt	Burst/ Steal	Read/Write (R/W)
Priority Control	14	Rotate Control	Not Used	Not Used	Not Used	TxRQ Enable #3 (TxEN3)	TxRQ Enable #2 (TxEN2)	TxRQ Enable #1 (TxEN1)	TxRQ Enable #0 (TxEN0)
Interrupt Control	15	IRQ Flag	Not Used	Not Used	Not Used	IRQ Enable #3 (IE3)	IRQ Enable #2 (IE2)	IRQ Enable #1 (IE1)	IRQ Enable #0 (IE0)
Data Chain	16	Not Used	Not Used	Not Used	Not Used	Two/Four Channel Select (2/4)	Data Chain Channel Select B	Data Chain Channel Select A	Data Chain Enable

* The x represents the binary equivalent of the channel desired.

A comparison of the response times and maximum transfer rates is shown in Table 9. The data are shown for a system clock rate of 1 MHz.

The two 8-bit bytes that form the registers in Table 10 are placed in consecutive memory locations, making it very easy to use the MPU index register in programming them.

Fig. 38 shows an example of its minimum structure (1 channel, HALT mode, combination with FDC). Fig. 39 shows an example of its maximum structure. (but only one DMAC is used.)

Table 9 Maximum Transfer Speed & Response Time of the DMAC when $t_{cyc\phi}$ equals 1 μsec.

Mode		Maximum Transfer Speed (μsec/byte)	Response Time (μsec)	
			maximum	minimum
HALT Cycle Steal		(executing time of one instruction) + 3	(executing time of one instruction) +3.5 − t_{TQH1}	3.5 + t_{TQS1}
HALT Burst	first byte	1		
	since second byte		2 − t_{TQH2}	1 + t_{TQS2}
TSC Cycle Steal		4	3.5 − t_{TQH1}	2.5 + t_{TQH1}

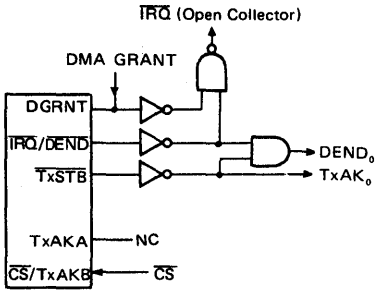


Figure 35 One Channel

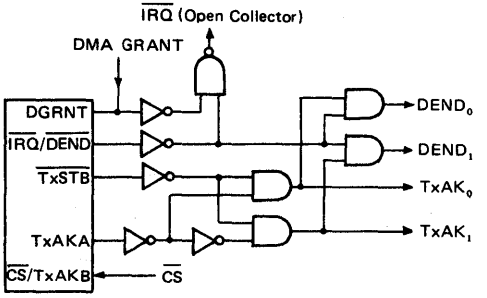


Figure 36 Two Channel

Table 10 Address and Byte Count Registers

Register	Channel	Address (Hex)
Address High	0	0
Address Low	0	1
Byte Count High	0	2
Byte Count Low	0	3
Address High	1	4
Address Low	1	5
Byte Count High	1	6
Byte Count Low	1	7
Address High	2	8
Address Low	2	9
Byte Count High	2	A
Byte Count Low	2	B
Address High	3	C
Address Low	3	D
Byte Count High	3	E
Byte Count Low	3	F

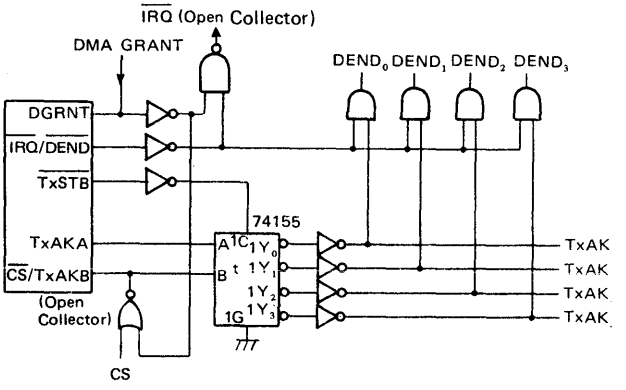


Figure 37 Four-Channel

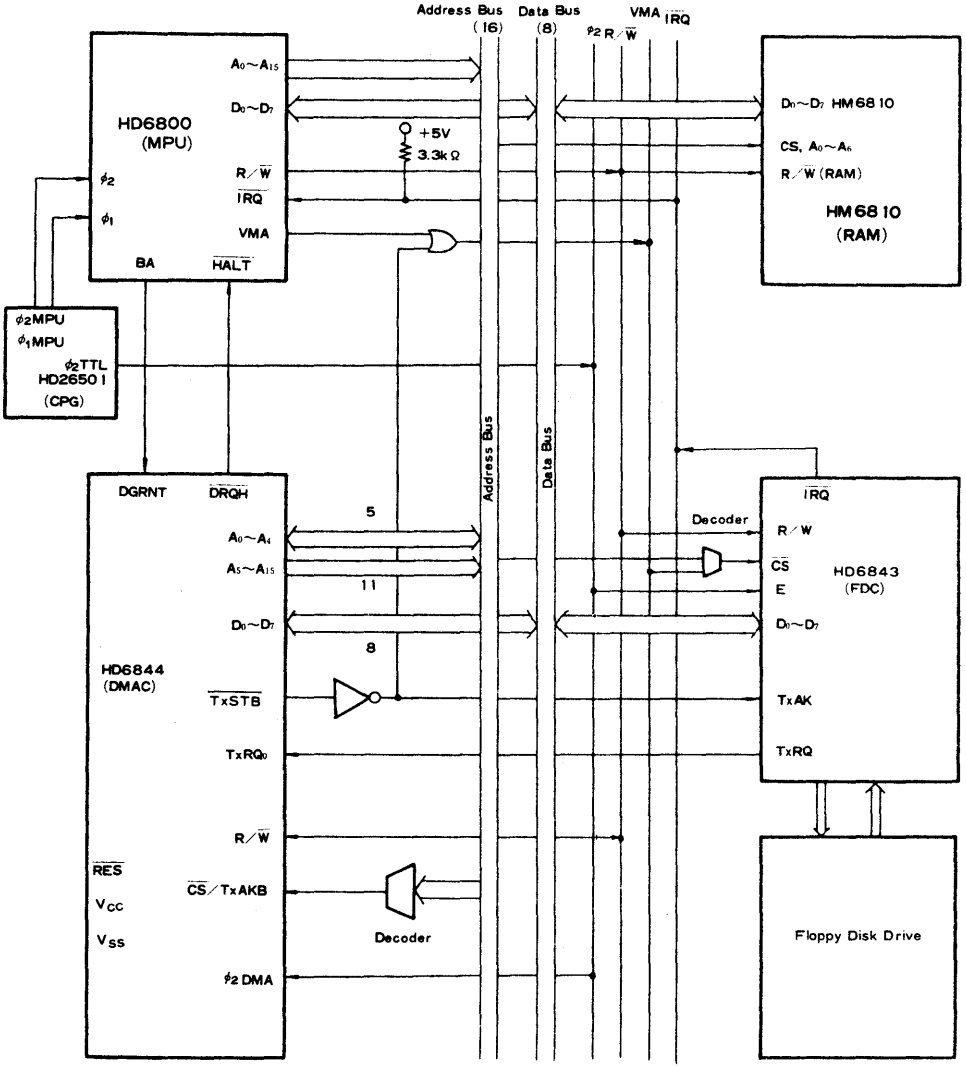


Figure 38 Example of DMA System Structure (1) (minimum)

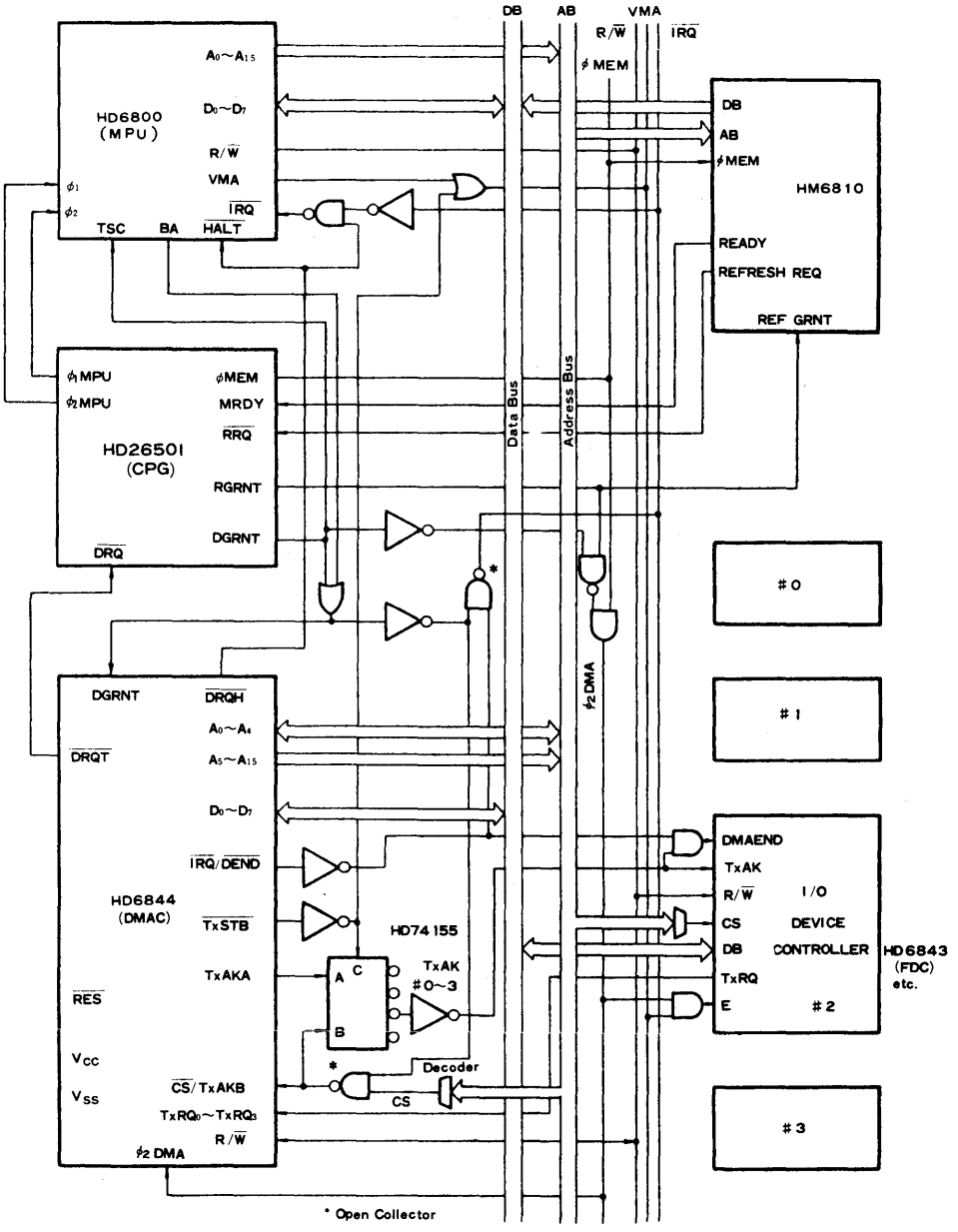
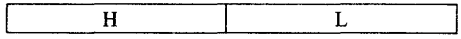


Figure 39 Example of DMA System Structure (2) (maximum)

■ APPENDIX

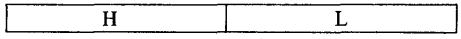
Contents of the DMAC Registers

(1) ADR0 ~ ADR3 (Address Register) (1 ADR on each channel)



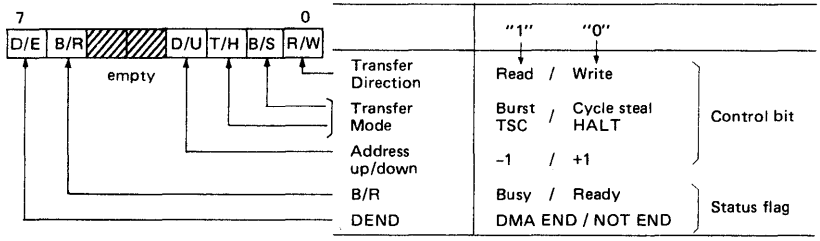
16 bit x 4

(2) BCR0 ~ BCR3 (Byte Count Register) (1 BCR on each channel)

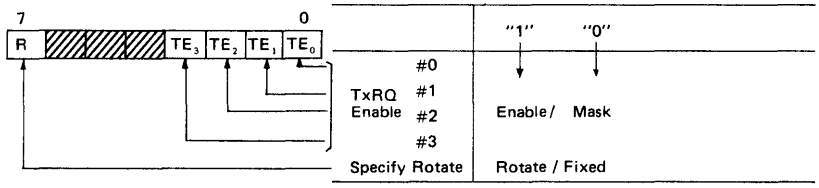


16 bit x 4

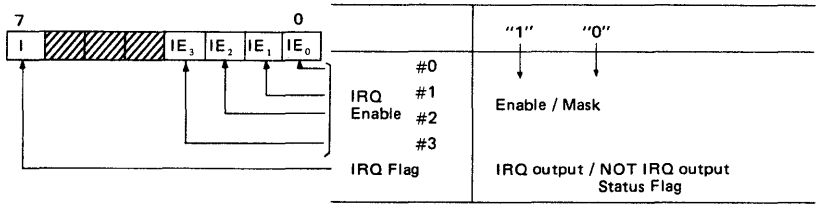
(3) CHCR0 ~ CHCR3 (Channel Control Register) (1 CHCR on each channel) (6 bit x 4)



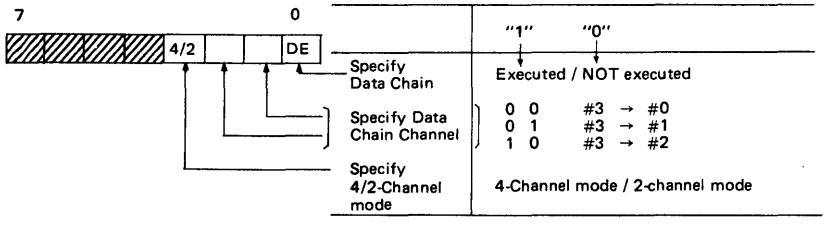
(4) PCR (Priority Control Register) (5 bit x 1)



(5) ICR (Interrupt Control Register) (5 bit x 1)



(6) DCR (Data Chain Control Register) (4 bit x1)

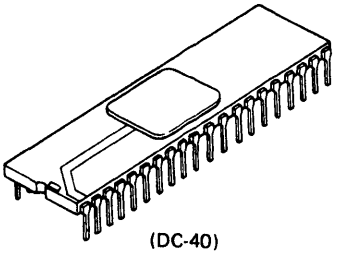


HD6845S, HD68A45S, HD68B45S

CRTC (CRT Controller)

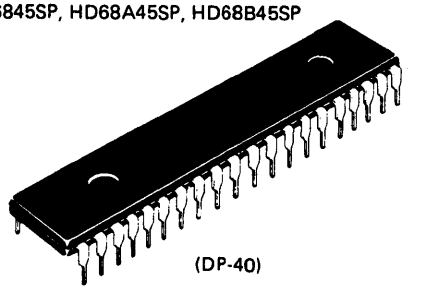
The CRTC is a LSI controller which is designed to provide an interface for microcomputers to raster scan type CRT displays. The CRTC belongs to the HMCS6800 LSI Family and has full compatibility with MPU in both data lines and control lines. Its primary function is to generate timing signal which is necessary for raster scan type CRT display according to the specification programmed by MPU. The CRTC is also designed as a programmable controller, so applicable to wide-range CRT display from small low-functioning character display up to raster type full graphic display as well as large high-functioning limited graphic display.

HD6845S, HD68A45S, HD68B45S



(DC-40)

HD6845SP, HD68A45SP, HD68B45SP

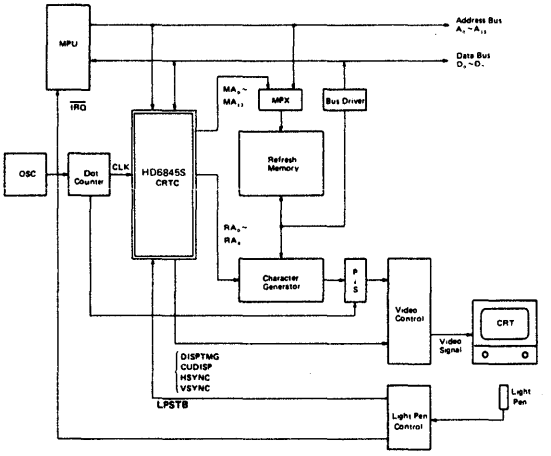


(DP-40)

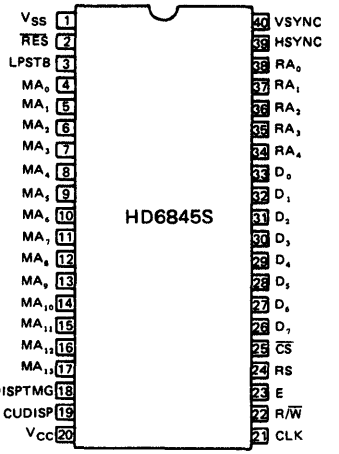
■ FEATURES

- Number of Displayed Characters on the Screen, Vertical Dot Format of One Character, Horizontal and Vertical Sync Signal, Display Timing Signal are Programmable
- 3.7 MHz High Speed Display Operation
- Line Buffer-less Refreshing
- 14-bit Refresh Memory Address Output (16k Words max. Access)
- Programmable Interlace/Non-interlace Scan Mode
- Built-in Cursor Control Function
- Programmable Cursor Height and its Blink
- Built-in Light Pen Detection Function
- Paging and Scrolling Capability
- TTL Compatible
- Single +5V Power Supply
- Compatible with MC6845☆1, MC68A45☆1, MC68B45☆1

■ SYSTEM BLOCK DIAGRAM



■ PIN ARRANGEMENT



(Top View)

■ ORDERING INFORMATION

CRTC	Bus Timing	CRT Display Timing
HD6845S	1.0 MHz	3.7 MHz max.
HD68A45S	1.5 MHz	
HD68B45S	2.0 MHz	

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC} *	−0.3 ~ +7.0	V
Input Voltage	V_{in} *	−0.3 ~ +7.0	V
Operating Temperature	T_{opr}	− 20 ~ + 75	°C
Storage Temperature	T_{stg}	− 55 ~ +150	°C

* With respect to V_{SS} (SYSTEM GND)

[NOTE] Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ RECOMMENDED OPERATING CONDITIONS

Item	Symbol	min	typ	max	Unit
Supply Voltage	V_{CC}*	4.75	5.0	5.25	V
Input Voltage	V_{IL}*	−0.3	–	0.8	V
	V_{IH}*	2.0	–	V_{CC}	V
Operating Temperature	T_{opr}	− 20	25	75	°C

* With respect to V_{SS} (SYSTEM GND)

■ ELECTRICAL CHARACTERISTICS

• DC CHARACTERISTICS (V_{CC} = 5V ± 5%, V_{SS} = 0V, Ta = −20~+75°C, unless otherwise noted.)

Item	Symbol	Test Condition		min	typ*	max	Unit
Input "High" Voltage	V_{IH}			2.0	–	V_{CC}	V
Input "Low" Voltage	V_{IL}			−0.3	–	0.8	V
Input Leakage Current	I_{in}	V_{in} = 0 ~ 5.25V (Except D_0~D_7)		−2.5	–	2.5	μA
Three-State Input Current (off-state)	I_{TSI}	V_{in} = 0.4 ~ 2.4V V_{CC} = 5.25V (D_0~ D_7)		− 10	–	10	μA
Output "High" Voltage	V_{OH}	I_{LOAD} = −205 μA (D_0~ D_7)		2.4	–	–	V
		I_{LOAD} = −100 μA (Other Outputs)					
Output "Low" Voltage	V_{OL}	I_{LOAD} = 1.6 mA		–	–	0.4	V
Input Capacitance	C_{in}	V_{in} = 0 Ta = 25°C f = 1.0 MHz	D_0 ~ D_7	–	–	12.5	pF
			Other Inputs	–	–	10.0	pF
Output Capacitance	C_{out}	V_{in} = 0V, Ta = 25°C, f = 1.0 MHz		–	–	10.0	pF
Power Dissipation	P_D			–	600	1000	mW

* Ta = 25°C, V_{CC} = 5.0V

- **AC CHARACTERISTICS (V_{CC} = 5V ±5%, V_{SS} = 0V, Ta = -20~+75°C, unless otherwise noted.)**

1. TIMING OF CRTC SIGNAL

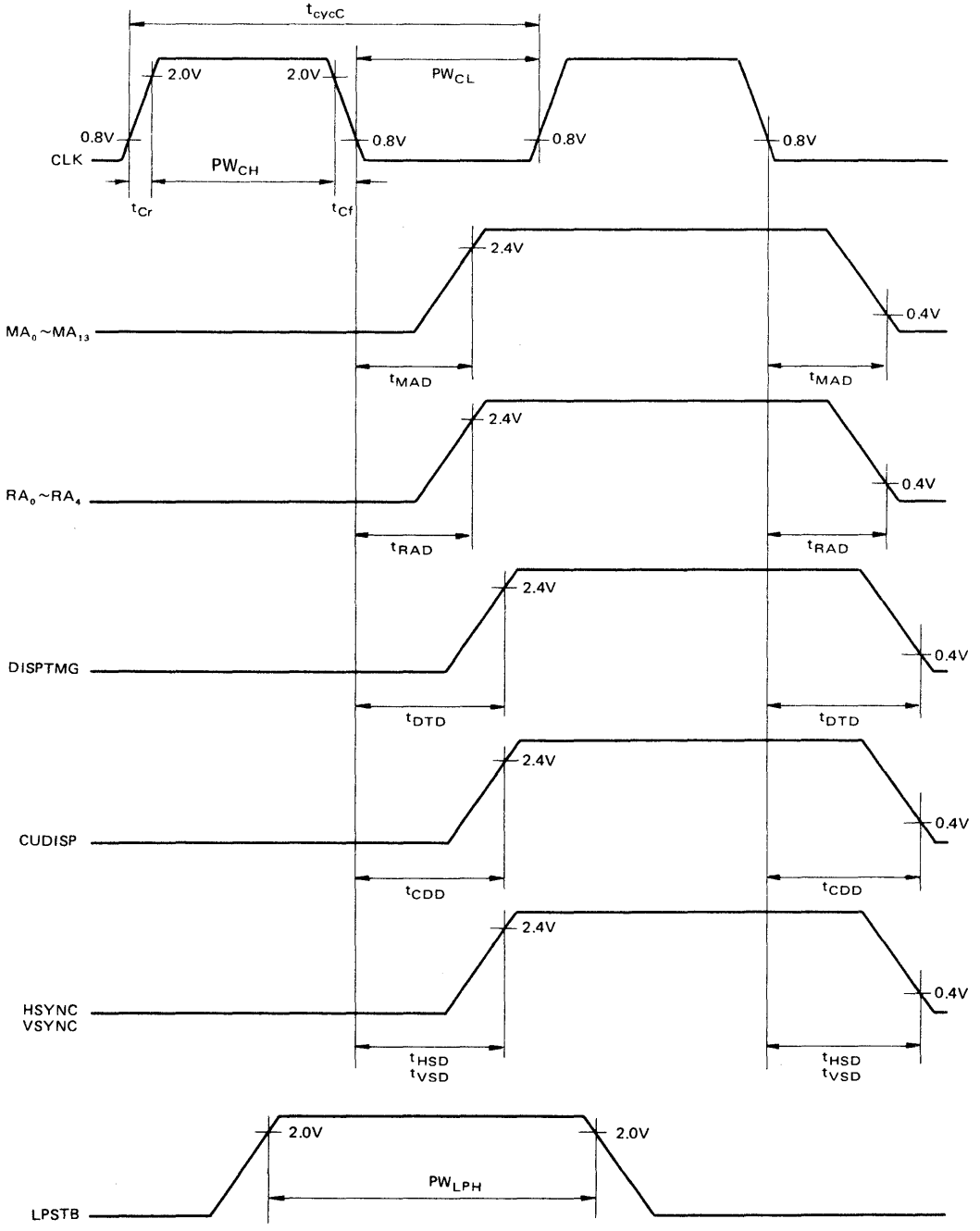
Item	Symbol	Test Condition	min	typ	max	Unit
Clock Cycle Time	t_{cycC}	Fig. 1	270	–	–	ns
Clock "High" Pulse Width	PW_{CH}		130	–	–	ns
Clock "Low" Pulse Width	PW_{CL}		130	–	–	ns
Rise andFall Time for Clock Input	t_{Cr}, t_{Cf}		–	–	20	ns
Memory Address Delay Time	t_{MAD}		–	–	160	ns
Raster Address Delay Time	t_{RAD}		–	–	160	ns
DISPTMG Delay Time	t_{DTD}		–	–	250	ns
CUDISP Delay Time	t_{CDD}		–	–	250	ns
Horizontal Sync Delay Time	t_{HSD}		–	–	200	ns
Vertical Sync Delay Time	t_{VSD}		–	–	250	ns
Light Pen Strobe Pulse Width	PW_{LPH}		60	–	–	ns
Light Pen Strobe	t_{LPD1}	Fig. 2	–	–	70	ns
Uncertain Time of Acceptance	t_{LPD2}		–	–	0	ns

2. MPU READ TIMING

Item	Symbol	Test Condition	HD6845S			HD68A45S			HD68B45S			Unit
			min	typ	max	min	typ	max	min	typ	max	
Enable Cycle Time	t_{cycE}	Fig. 3	1.0	–	–	0.666	–	–	0.5	–	–	μs
Enable "High" Pulse Width	PW_{EH}		0.45	–	–	0.280	–	–	0.22	–	–	μs
Enable "Low" Pulse Width	PW_{EL}		0.40	–	–	0.280	–	–	0.21	–	–	μs
Enable Rise and Fall Time	t_{Er}, t_{Ef}		–	–	25	–	–	25	–	–	25	ns
Address Set Up Time	t_{AS}		140	–	–	140	–	–	70	–	–	ns
Data Delay Time	t_{DDR}		–	–	320	–	–	220	–	–	180	ns
Data Hold Time	t_{H}		10	–	–	10	–	–	10	–	–	ns
Address Hold Time	t_{AH}		10	–	–	10	–	–	10	–	–	ns
Data Access Time	t_{ACC}		–	–	460	–	–	360	–	–	250	ns

3. MPU WRITE TIMING

Item	Symbol	Test Condition	HD6845S			HD68A45S			HD68B45S			Unit
			min	typ	max	min	typ	max	min	typ	max	
Enable Cycle Time	t_{cycE}	Fig. 4	1.0	–	–	0.666	–	–	0.5	–	–	μs
Enable "High" Pulse Width	PW_{EH}		0.45	–	–	0.280	–	–	0.22	–	–	μs
Enable "Low" Pulse Width	PW_{EL}		0.40	–	–	0.280	–	–	0.21	–	–	μs
Enable Rise and Fall Time	t_{Er}, t_{Ef}		–	–	25	–	–	25	–	–	25	ns
Address Set Up Time	t_{AS}		140	–	–	140	–	–	70	–	–	ns
Data Set Up Time	t_{DSW}		195	–	–	80	–	–	60	–	–	ns
Data Hold Time	t_{H}		10	–	–	10	–	–	10	–	–	ns
Address Hold Time	t_{AH}		10	–	–	10	–	–	10	–	–	ns



This Figure shows the relation in time between CLK signal and each output signals. Output sequence is shown in Figs. 10~15.

Figure 1 Time Chart of the CRTC

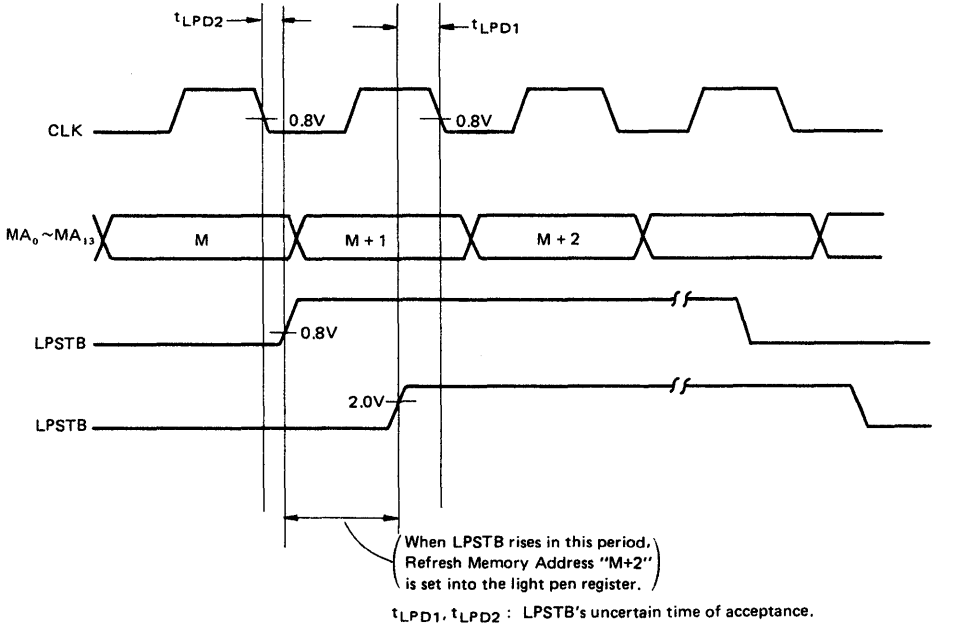


Figure 2 LPSTB Input Timing & Refresh Memory Address that is set into the light pen register.

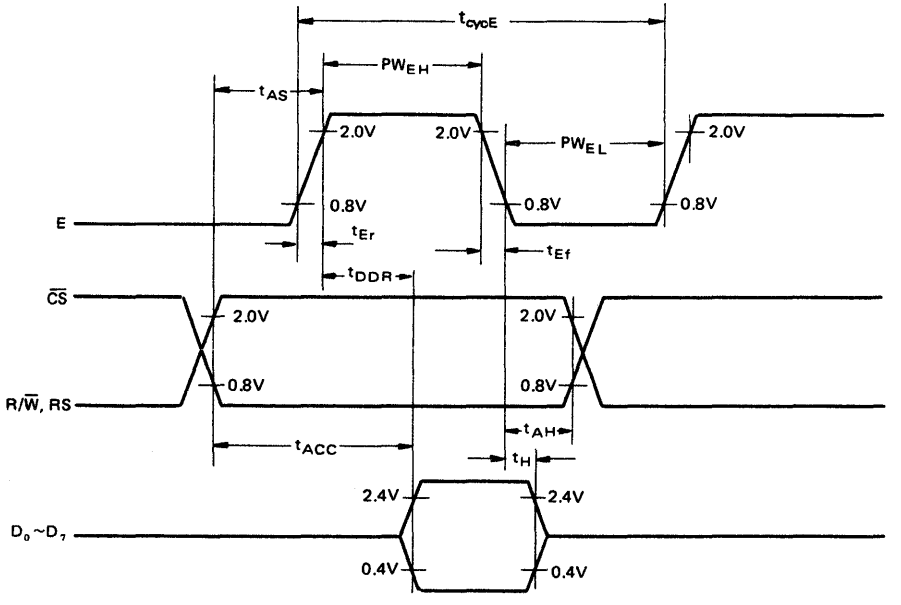


Figure 3 Read Sequence

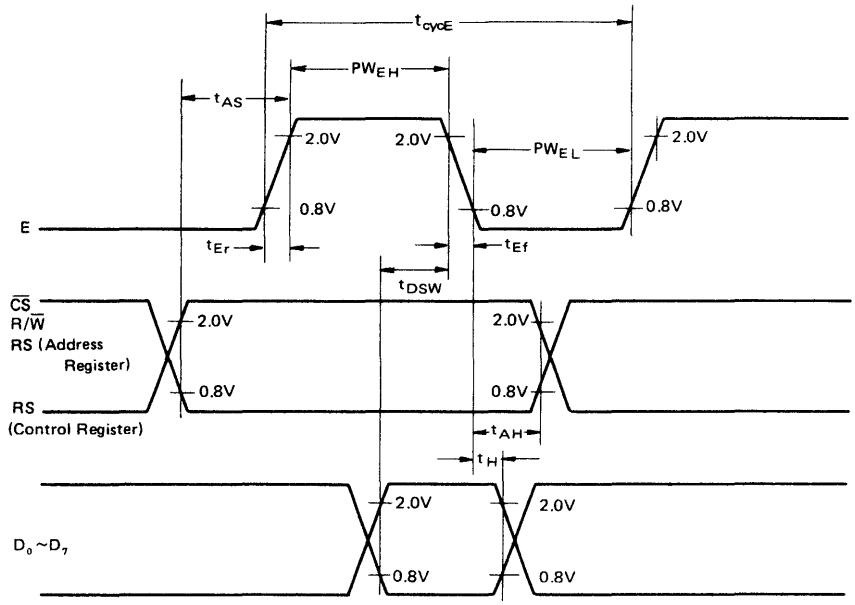
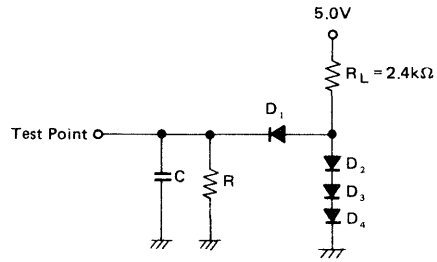


Figure 4 Write Sequence



C = 130pF (D_0 ~ D_7)
= 30pF (Output signals except D_0 ~ D_7)
R = 11kΩ (D_0 ~ D_7)
= 24kΩ (Output signals except D_0 ~ D_7)
D_1 ~ D_4 are 1S2074Ⓗ or equivalent.

Figure 5 Test Loads

■ SYSTEM DESCRIPTION

The CRTC is a LSI which is connected with MPU and CRT display device to control CRT display. The CRTC consists of internal register group, horizontal and vertical timing circuits, linear address generator, cursor control circuit, and light pen detection circuit. Horizontal and vertical timing circuit generate RA_0~RA_4, DISPTMG, HSYNC, and VSYNC. RA_0~RA_4 are raster address signals and used as input signals for Character Generator. DISPTMG, HSYNC, and VSYNC signals are received by video control circuit. This horizontal and vertical timing circuit consists of internal counter and comparator circuit.

Linear address generator generates refresh memory address MA_0 ~MA_{13} to be used for refreshing the screen. By these address signals, refresh memory is accessed periodically. As 14 refresh memory address signals are prepared, 16k words max are accessible. Moreover, the use of start address register enables paging and scrolling. Light pen detection circuit detects light pen position on the screen. When light pen strobe signal is received, light pen register memorizes linear address generated by linear address generator in order to memorize where light pen is on the screen. Cursor control circuit controls the position of cursor, its height, and its blink.

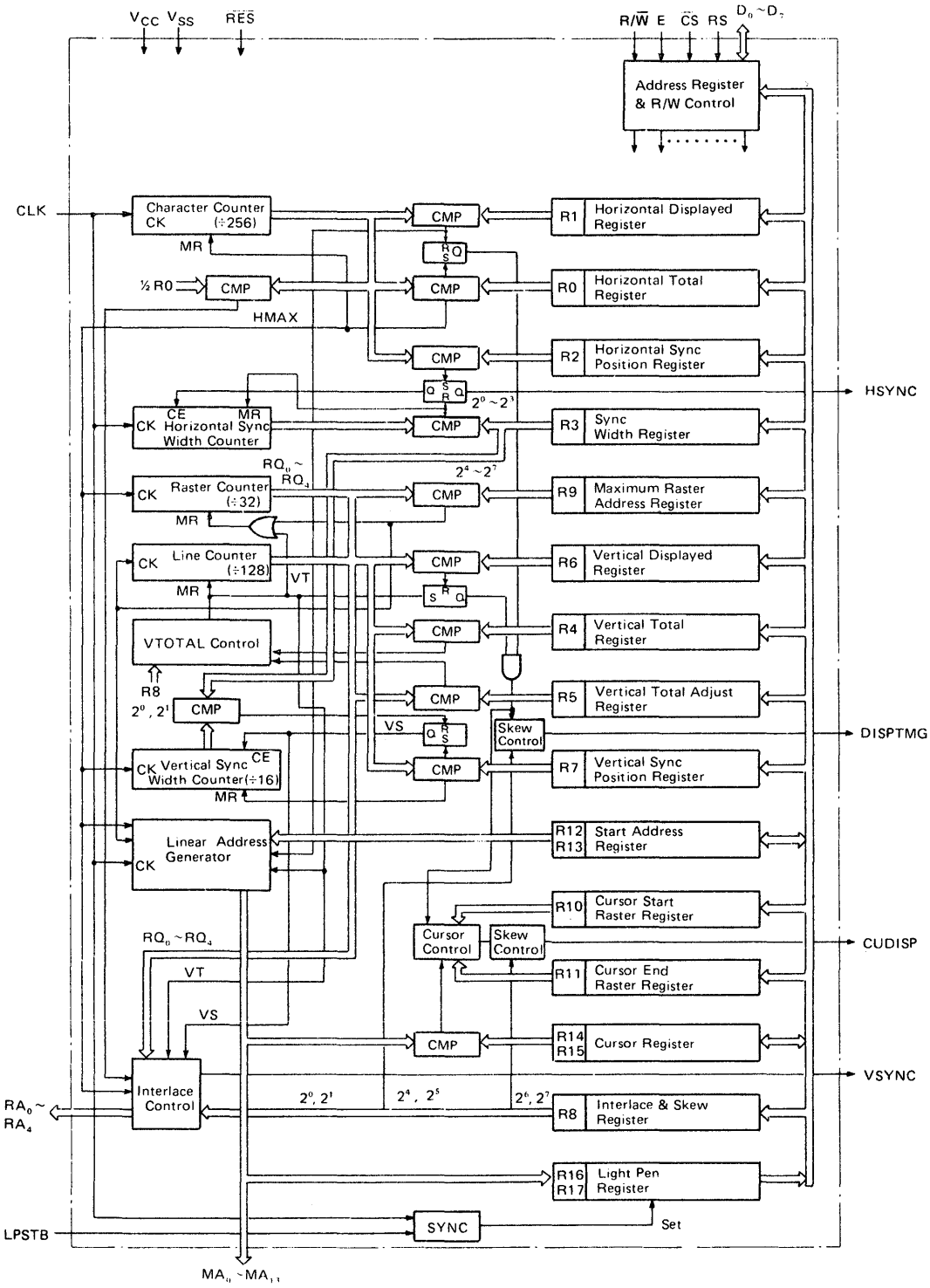


Figure 6 Internal Block Diagram of the CRTC

■ FUNCTION OF SIGNAL LINE

The CRTC provides 13 interface signals to MPU and 25 interface signals to CRT display.

• Interface Signals to MPU

Bi-directional Data Bus (D_0~D_7)

Bi-directional data bus(D_0~D_7) are used for data transfer between the CRTC and MPU. The data bus outputs are 3-state buffers and remain in the high-impedance state except when MPU performs a CRTC read operation.

Read/Write (R/$\overline{W}$)

R/$\overline{W}$ signal controls the direction of data transfer between the CRTC and MPU. When R/$\overline{W}$ is at "High" level, data of CRTC is transfered to MPU. When R/$\overline{W}$ is at "Low" level, data of MPU is transfered to CRTC.

Chip Select ($\overline{CS}$)

Chip Select signal ($\overline{CS}$) is used to address the CRTC. When $\overline{CS}$ is at "Low" level, it enables R/W operation to CRTC internal registers. Normally this signal is derived from decoded address signal of MPU under the condition that VMA signal of MPU is at "High" level.

Register Select (RS)

Register Select signal (RS) is used to select the address register and 18 control registers of the CRTC. When RS is at "Low" level, the address register is selected and when RS is at "High" level, control registers are selected. This signal is normally a derivative of the lowest bit (A0) of MPU address bus.

Enable(E)

Enable signal (E) is used as strobe signal in MPU R/W operation with the CRTC internal registers. This signal is normally a derivative of the HMCS6800 System ϕ_2 clock.

Reset ($\overline{RES}$)

Reset signal ($\overline{RES}$) is an input signal used to reset the CRTC. When $\overline{RES}$ is at "Low" level, it forces the CRTC into the following status.

1) All the counters in the CRTC are cleared and the device stops the display operation.
2) All the outputs go down to "Low" level.
3) Control registers in the CRTC are not affected and remain unchanged.

This signal is different from other HMCS6800 family LSIs in the following functions and has restrictions for usage.

1) $\overline{RES}$ signal has capability of reset function only when LPSTB is at "Low" level.
2) The CRTC starts the display operation immediately after $\overline{RES}$ signal goes "High".

• Interface Signals to CRT Display Device

Character Clock (CLK)

CLK is a standard clock input signal which defines character timing for the CRTC display operation. This signal is normally derived from the external high-speed dot timing logic.

Horizontal Sync (HSYNC)

HSYNC is an active "High" level signal which provides horizontal synchronization for display device.

Vertical Sync (VSYNC)

VSYNC is an active "High" level signal which provides vertical synchronization for display device.

Display Timing (DISPTMG)

DISPTMG is an active "High" level signal which defines the display period in horizontal and vertical raster scanning. It is necessary to enable video signal only when DISPTMG is at "High" level.

Refresh Memory Address (MA_0~MA_{13})

MA_0~MA_{13} are refresh memory address signals which are used to access to refresh memory in order to refresh the CRT screen periodically. These outputs enables 16k words max. refresh memory access. So, for instance, these are applicable up to 2000 characters/screen and 8-page system.

Raster Address (RA_0~RA_4)

RA_0~RA_4 are raster address signals which are used to select the raster of the character generator or graphic pattern generator etc.

Cursor Display (CUDISP)

CUDISP is an active "High" level video signal which is used to display the cursor on the CRT screen. This output is inhibited while DISPTMG is at "Low" level. Normally this output is mixed with video signal and provided to the CRT display device.

Light Pen Strobe (LPSTB)

LPSTB is an active "High" level input signal which accepts strobe pulse detected by the light pen and control circuit. When this signal is activated, the refresh memory address (MA_0~MA_{13}) which are shown in Fig. 2 are stored in the 14-bit light pen register. The stored refresh memory address need to be corrected in software, taking the delay time of the display device, light pen, and light pen control circuits into account.

■ REGISTER DESCRIPTION

Table 1 Internal Registers Assignment

$\overline{CS}$	RS	Address Register 4	3	2	1	0	Register #	Register Name	Program Unit	READ	WRITE	Data Bit 7	6	5	4	3	2	1	0
1	×	×	×	×	×	×			–	–	–								
0	0	×	×	×	×	×	AR	Address Register	–	×	○								
0	1	0	0	0	0	0	R0	Horizontal Total *	Character	×	○								
0	1	0	0	0	0	1	R1	Horizontal Displayed	Character	×	○								
0	1	0	0	0	1	0	R2	Horizontal Sync* Position	Character	×	○								
0	1	0	0	0	1	1	R3	Sync Width	Vertical-Raster, Horizontal-Character	×	○	wv3	wv2	wv1	wv0	wh3	wh2	wh1	wh0
0	1	0	0	1	0	0	R4	Vertical Total *	Line	×	○								
0	1	0	0	1	0	1	R5	Vertical Total Adjust	Raster	×	○								
0	1	0	0	1	1	0	R6	Vertical Displayed	Line	×	○								
0	1	0	0	1	1	1	R7	Vertical Sync * Position	Line	×	○								
0	1	0	1	0	0	0	R8	Interlace & Skew	–	×	○	C1	C0	D1	D0			V	S
0	1	0	1	0	0	1	R9	Maximum Raster Address	Raster	×	○								
0	1	0	1	0	1	0	R10	Cursor Start Raster	Raster	×	○		B	P					
0	1	0	1	0	1	1	R11	Cursor End Raster	Raster	×	○								
0	1	0	1	1	0	0	R12	Start Address(H)	–	○	○								
0	1	0	1	1	0	1	R13	Start Address(L)	–	○	○								
0	1	0	1	1	1	0	R14	Cursor(H)	–	○	○								
0	1	0	1	1	1	1	R15	Cursor (L)	–	○	○								
0	1	1	0	0	0	0	R16	Light Pen(H)	–	○	×								
0	1	1	0	0	0	1	R17	Light Pen(L)	–	○	×								

[NOTE] 1. The Registers marked *: (Written Value) = (Specified Value) – 1
2. Written Value of R9 is mentioned below.
1) Non-interlace Mode / Interlace Sync Mode } (Written Value Nr) = (Specified Value) – 1
2) Interlace Sync & Video Mode
(Written Value Nr) = (Specified Value) – 2
3. C0 and C1 specify skew of CUDISP output signal.
D0 and D1 specify skew of DISPTMG output signal.
When S is "1", V specifies video mode. S specifies the Interlace Sync Mode.
4. B specifies the cursor blink. P specifies the cursor blink period.
5. wv0~wv3 specify the pulse width of Vertical Sync Signal.
wh0~wh3 specify the pulse width of Horizontal Sync Signal.
6. R0 is ordinally programmed to be odd number in interlace mode.
7. ○; Yes, ×; No

- **Address Register (AR)**

This is a 5-bit register used to select 18 internal control registers (R0~R17). Its contents are the address of one of 18 internal control registers. Programming the data from 18 to 31 produces no results. Access to R0~R17 requires, first of all, to write the address of corresponding control register into this register. When RS and $\overline{CS}$ are at "Low" level, this register is selected.

- **Horizontal Total Register (R0)**

This is a register used to program total number of horizontal characters per line including the retrace period. The data is 8-bit and its value should be programmed according to the specification of the CRT. When M is total number of characters, (M-1) shall be programmed to this register. When programming for interlace mode, M must be even.

- **Horizontal Displayed Register (R1)**

This is a register used to program the number of horizontal displayed characters per line. Data is 8-bit and any number that is smaller than that of horizontal total characters can be programmed.

- **Horizontal Sync Position Register (R2)**

This is a register used to program horizontal sync position as multiples of the character clock period. Data is 8-bit and any number that is lower than the horizontal total number can be programmed. When H is character number of horizontal Sync Position, (H-1) shall be programmed to this register. When programmed value of this register is increased, the display position on the CRT screen is shifted to the left. When programmed value is decreased, the position is shifted to the right. Therefore, the optimum horizontal position can be determined by this value.

- **Sync Width Register (R3)**

This is a register used to program the horizontal sync pulse width and the vertical sync pulse width. The horizontal sync pulse width is programmed in the lower 4-bit as multiples of the character clock period. "0" cannot be programmed. The vertical sync pulse width is programmed in higher 4-bit as multiples of the raster period. When "0" is programmed in higher 4-bit, 16 raster period (16H) is specified.

- **Vertical Total Register (R4)**

This is a register used to program total number of lines per frame including vertical retrace period. The data is within 7-bit and its value should be programmed according to the specification of the CRTC. When N is total number of lines, (N-1) shall be programmed to this register.

- **Vertical Total Adjust Register (R5)**

This is a register used to program the optimum number to adjust total number of rasters per field. This register enables to decide the number of vertical deflection frequency more strictly.

- **Vertical Displayed Register (R6)**

This is a register used to program the number of displayed character rows on the CRT screen. Data is 7-bit and any number that is smaller than that of vertical total characters can be programmed.

Table 2 Pulse Width of Vertical Sync Signal

VSW				Pulse Width
2^7	2^6	2^5	2^4	
0	0	0	0	16H
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	10
1	0	1	1	11
1	1	0	0	12
1	1	0	1	13
1	1	1	0	14
1	1	1	1	15

H; Raster period

Table 3 Pulse Width of Horizontal Sync Signal

HSW				Pulse Width
2^3	2^2	2^1	2^0	
0	0	0	0	– (Note)
0	0	0	1	1 CH
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	10
1	0	1	1	11
1	1	0	0	12
1	1	0	1	13
1	1	1	0	14
1	1	1	1	15

CH; Character clock period
(Note) HSW = "0" cannot be used.

- **Vertical Sync Position Register (R7)**

This is a register used to program the vertical sync position on the screen as multiples of the horizontal character line period. Data is 7-bit and any number that is equal to or less than vertical total characters can be programmed. When V is character number of vertical sync position, (V-1) shall be programmed to this register. When programmed value of this register is increased, the display position is shifted up. When programmed value is decreased, the position is shifted down. Therefore, the optimum vertical position may be determined by this value.

- **Interlace and Skew Register (R8)**

This is a register used to program raster scan mode and skew (delay) of CUDISP signal and DISPTMG signal.

Interlace Mode Program Bit (V, S)

Raster scan mode is programmed in the V, S bit.

Table 4 Interlace Mode (2^1, 2^0)

V	S	Raster Scan Mode
0	0	Non-interlace Mode
1	0	
0	1	Interlace Sync Mode
1	1	Interlace Sync & Video Mode

In the non-interlace mode, the rasters of even number field and odd number field are scanned duplicatedly. In the interlace sync mode, the rasters of odd number field are scanned in the middle of even number field. Then it is controlled to display the same character pattern in two fields. In the interlace sync and video mode, the raster scan method is the same as the interlace sync mode, but it is controlled to display different character pattern in two field.

Skew Program Bit (C1, C0, D1, D0)

These are used to program the skew (delay) of CUDISP signal and DISPTMG signal.

Skew of these two kinds of signals are programmed separately.

Table 5 DISPTMG Skew Bit (2^5, 2^4)

D1	D0	DISPTMG Signal
0	0	Non-skew
0	1	One-character skew
1	0	Two-character skew
1	1	Non-output

Table 6 Cursor Skew Bit (2^7, 2^6)

C1	C0	Non-skew
0	0	Non-skew
0	1	One-character skew
1	0	Two-character skew
1	1	Non-output

Skew function is used to delay the output timing of CUDISP and DISPTMG signals in LSI for the time to access refresh memory, character generator or pattern generator, and to make the same phase with serial video signal.

- **Maximum Raster Address Register (R9)**

This is a register used to program maximum raster address within 5-bit. This register defines total number of rasters per character including space. This register is programmed as follows.

Non-interlace Mode, Interlace Sync Mode

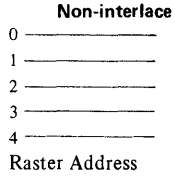
When total number of rasters is RN, (RN-1) shall be programmed.

Interlace Sync & Video Mode

When total number of rasters is RN, (RN-2) shall be programmed.

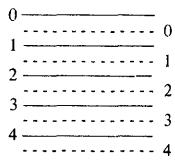
This manual defines total number of rasters in non-interlace mode, interlace sync mode and interlace sync & video mode as follows:

Non-interlace Mode



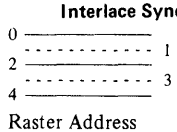
Total Number of Rasters 5
Programmed Value Nr = 4
(The same as displayed total number of rasters)

Interlace Sync Mode



Total Number of Rasters 5
Programmed Value Nr = 4
(In the interlace sync mode, total number of rasters in both the even and odd fields is ten. On programming, the half of it is defined as total number of rasters.)

Interlace Sync & Video Mode



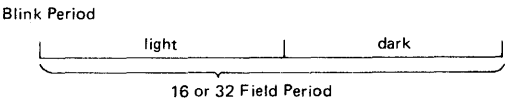
Total Number of Rasters 5
Programmed Value Nr = 3
(Total number of rasters displayed in the even field and the odd field.)

- **Cursor Start Raster Register (R10)**

This is a register used to program the cursor start raster address by lower 5-bit (2^0~2^4) and the cursor display mode by higher 2-bit (2^5, 2^6).

Table 7 Cursor Display Mode (2^6, 2^5)

B	P	Cursor Display Mode
0	0	Non-blink
0	1	Cursor Non-display
1	0	Blink, 16 Field Period
1	1	Blink, 32 Field Period



- **Cursor End Raster Register (R11)**
  This is register used to program the cursor end raster address.

- **Start Address Register (R12, R13)**
  These are used to program the first address of refresh memory to read out.
  Paging and scrolling is easily performed using this register. This register can be read but the higher 2-bit (2^6, 2^7) of R12 are always "0".

- **Cursor Register (R14, R15)**
  These two read/write registers stores the cursor location. The higher 2-bit (2^6, 2^7) of R14 are always "0".

- **Light Pen Register (R16, R17)**
  These read only registers are used to catch the detection address of the light pen. The higher 2-bit (2^6, 2^7) of R16 are always "0". Its value needs to be corrected by software because there is time delay from address output of the CRTC to signal input LPSTB pin of the CRTC in the process that raster is lit after address output and light pen detects it. Moreover, delay time shown in Fig. 2 needs to be taken into account.

Restriction on Programming Internal Register

1) $0 < Nhd < Nht + 1 \leqq 256$
2) $0 < Nvd < Nvt + 1 \leqq 128$
3) $0 \leqq Nhsp \leqq Nht$
4) $0 \leqq Nvsp \leqq Nvt$*
5) $0 \leqq N_{CSTART} \leqq N_{CEND} \leqq Nr$ (Non-interlace, Interlace sync mode)
   $0 \leqq N_{CSTART} \leqq N_{CEND} \leqq Nr + 1$ (Interlace sync & video mode)
6) $2 \leqq Nr \leqq 30$
7) $3 \leqq Nht$ (Except non-interlace mode)
   $5 \leqq Nht$ (Non-interlace mode only)

* In the interlace mode, pulse width is changed ±½ raster time when vertical sync signal extends over two fields.

Notes for Use

The method of directly using the value programmed in the internal register of LSI for controlling the CRT is adopted. Consequently, the display may flicker on the screen when the contents of the registers are changed from bus side asyncronously with the display operation.

Cursor Register

Writing into this register at frequent intervals for moving the cursor should be performed during horizontal and vertical retrace period.

Start Address Register

Writing into the start address register at frequent intervals for scrolling and paging should be performed during horizontal and vertical display period.

It is desirable to avoid programming other registers during display operation.

■ OPERATION OF THE CRTC

- **Time Chart of CRT Interface Signals**
  The following example shows the display operation in which values of Table 8 are programmed to the CRTC internal registers. Fig. 7 shows the CRT screen format. Fig. 10 shows the time chart of signals output from the CRTC.

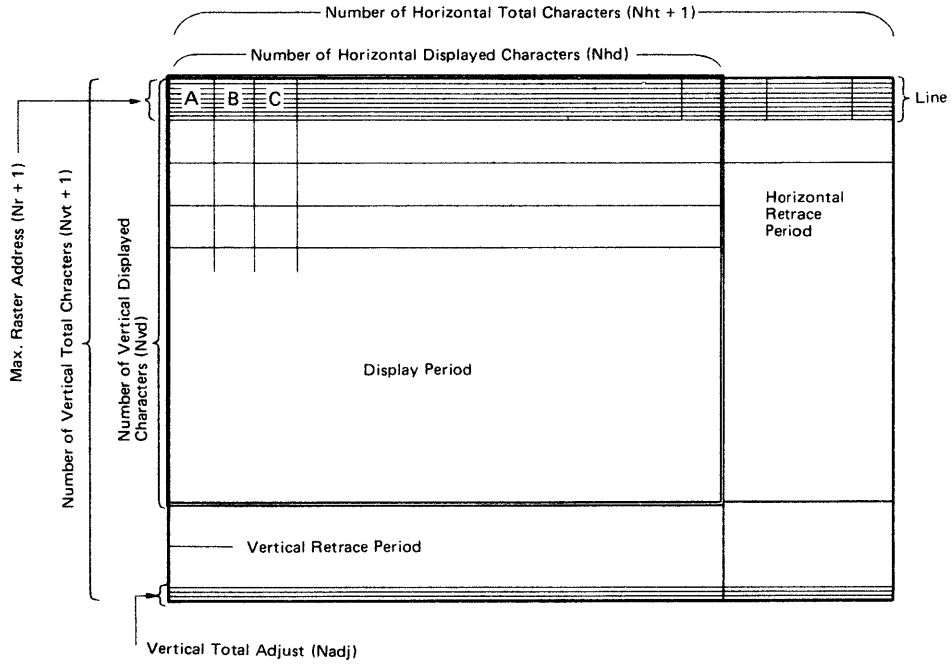


Figure 7 CRT screen Format

Table 8 Programmed Values into the Registers

Register	Register Name	Value	Register	Register Name	Value
R0	Horizontal Total	Nht	R9	Max. Raster Address	Nr
R1	Horizontal Displayed	Nhd	R10	Cursor Start Raster	
R2	Horizontal Sync Position	Nhsp	R11	Cursor End Raster	
R3	Sync Width	Nvsw, Nhsw	R12	Start Address (H)	0
R4	Vertical Total	Nvt	R13	Start Address (L)	0
R5	Vertical Total Adjust	Nadj	R14	Cursor (H)	
R6	Vertical Displayed	Nvd	R15	Cursor (L)	
R7	Vertical Sync Position	Nvsp	R16	Light Pen (H)	
R8	Interlace & Skew		R17	Light Pen (L)	

[NOTE] Nhd<Nht, Nvd<Nvt

The relation between values of Refresh Memory Address ($MA_0 \sim MA_{13}$) and Raster Address ($RA_0 \sim RA_4$) and the display position on the screen is shown in Fig. 16. Fig. 16 shows the case where the value of Start Address is 0.

- **Interlace Control**

Fig. 8 shows an example where the same character is displayed in the non-interlace mode, interlace sync mode, and video mode.

Non-interlace Mode Display

In non-interlace mode, each field is scanned duplicatedly. The values of raster addresses ($RA_0 \sim RA_4$) are counted up one from 0.

Interlace Sync Mode Display

In the interlace sync mode, raster addressed in the even field and the odd field are the same as addressed in the noninterlace mode. One character pattern is displayed mutually and its displayed position in the odd field is set at 1/2 raster space down from that in the even field.

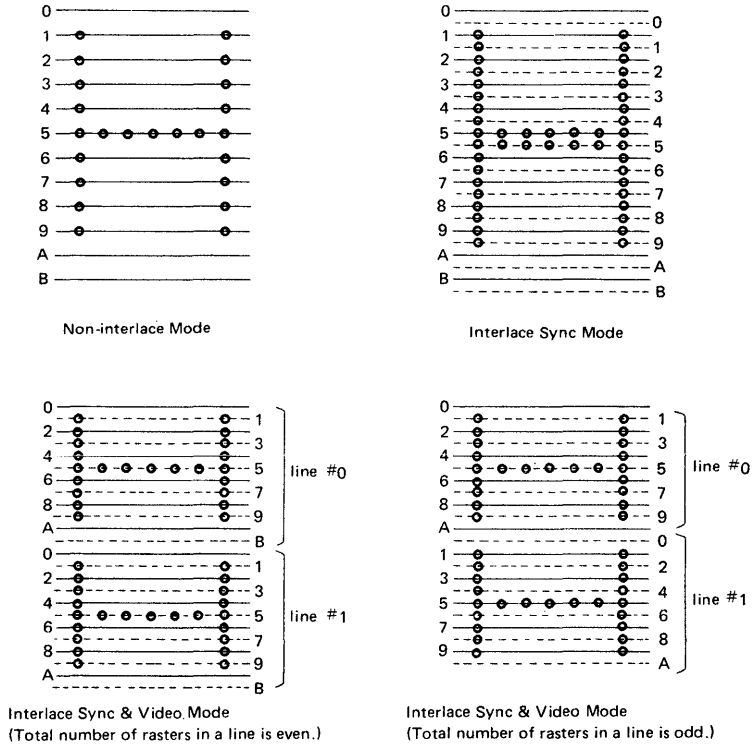


Figure 8 Example of Raster Scan Display

Interlace Sync & Video Mode Display

In interlace sync & video mode, the output raster address when the number of rasters is even is different from that when the number of rasters is odd.

Table 9 The Output of Raster Address in Interlace Sync & Video Mode

Total Number of Rasters in a Line \ Field		Even Field	Odd Field
Even		Even Address	Odd Address
Odd	Even Line*	Even Address	Odd Address
	Odd Line*	Odd Address	Even Address

* Internal line address begins from 0.

1) Total number of rasters in a line is even;

When number of rasters is programmed to be even, even raster address is output in the even field and odd raster address is output in the odd field.

2) Total number of rasters in a line is odd;

When total number of rasters is programmed to be odd, odd and even addresses are reversed according to the odd and even lines in each field. In this case, the difference in numbers of dots displayed between even field and odd field is usually smaller the case of 1). Then interlace can be displayed more stably.

[NOTE] The wide disparity of dots between number of dots between even field and odd field influences beam current of CRT. CRT, which has a stable high-voltage part, can make interlace display normal. On the contrary, CRT, which has unstable high-voltage part, moves deflection angle of beam current and also dots displayed in the even and odd fields may be shifted. Characters appears distroting on a border of the screen. So 2) programming has an effect to decrease such evil influences as mentioned above. Fig. 13 shows fine chart in each mode when interlace is performed.

- **Cursor Control**

Fig. 9 shows the display patterns where each value is programmed to the cursor start raster register and the cursor end raster register. Programmed values to the cursor start raster register and the cursor end raster register need to be under the following condition.

Cursor Start Raster Register ≦ Cursor End Raster Register ≦ Maximum Raster Address Register.

Time chart of CUDISP output signal is shown in Fig. 14 and Fig. 15.

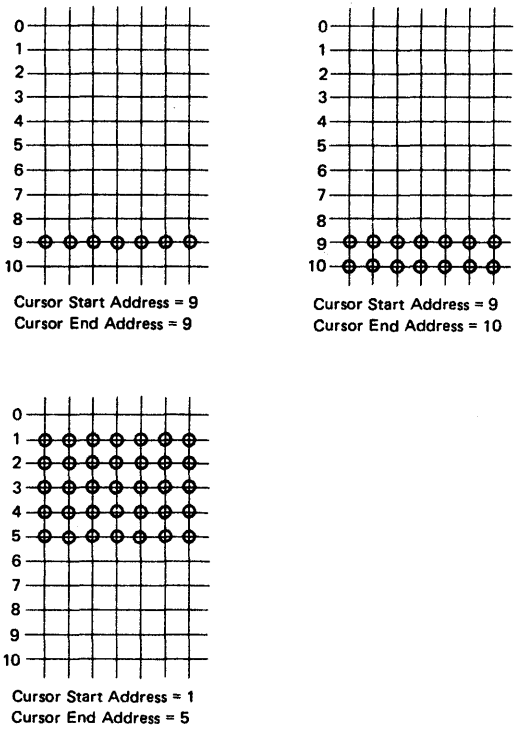


Figure 9 Cursor Control

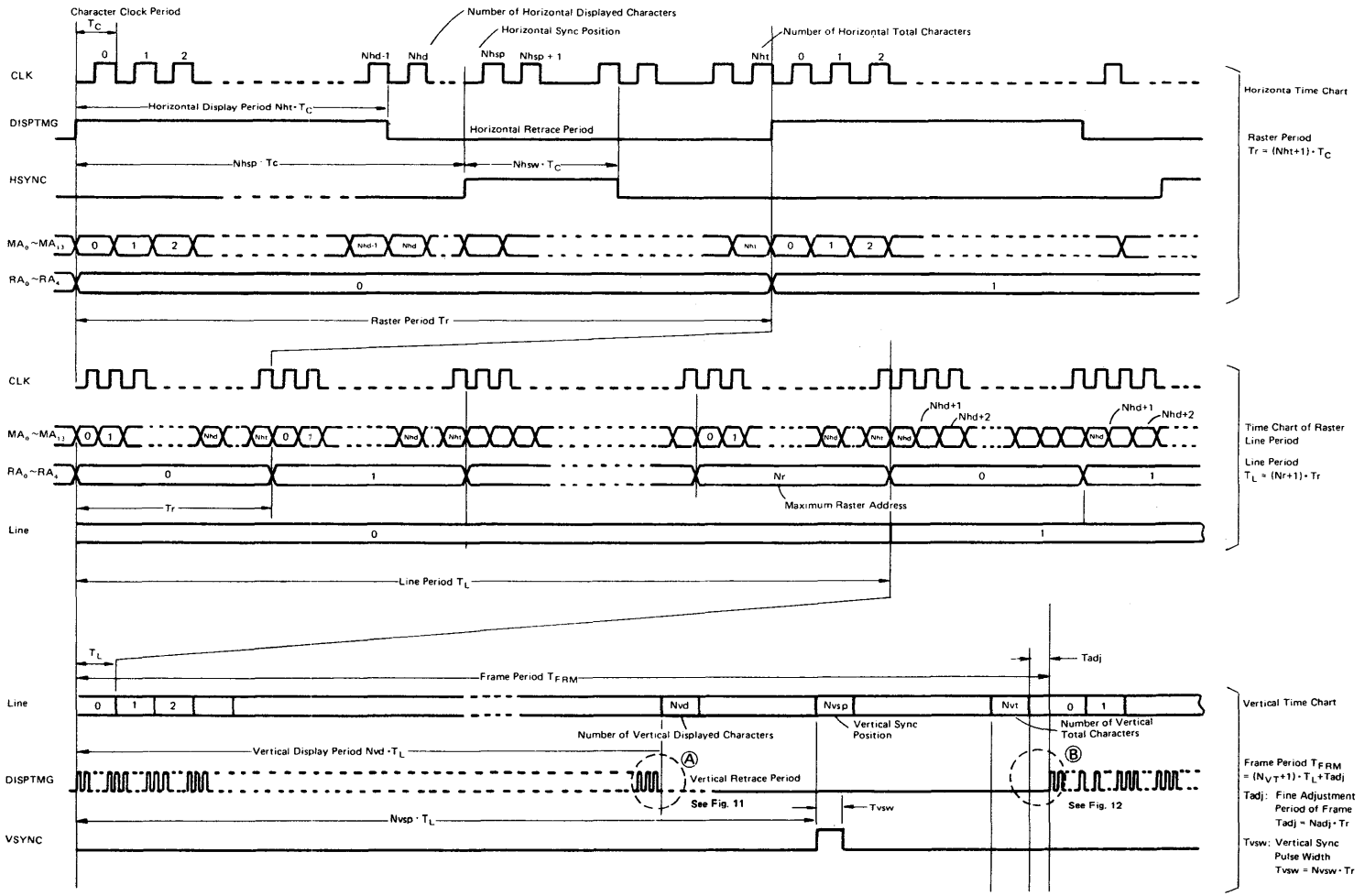


Figure 10 CRTC Time Chart

(Output waveform of horizontal & vertical display in the case where values shown in Table 8 are Programmed to each register.)

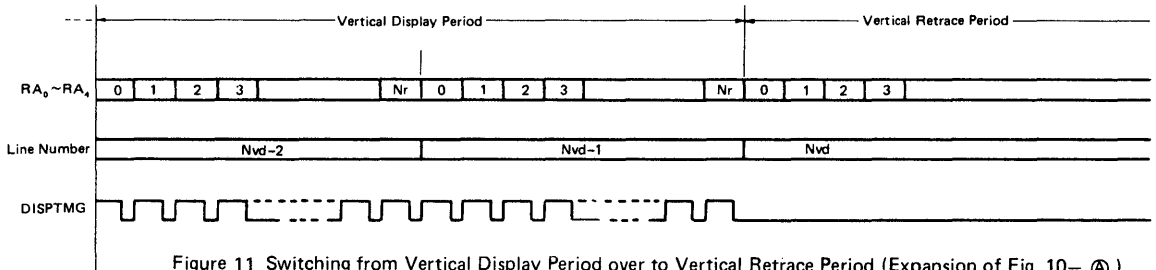


Figure 11 Switching from Vertical Display Period over to Vertical Retrace Period (Expansion of Fig. 10– Ⓐ)

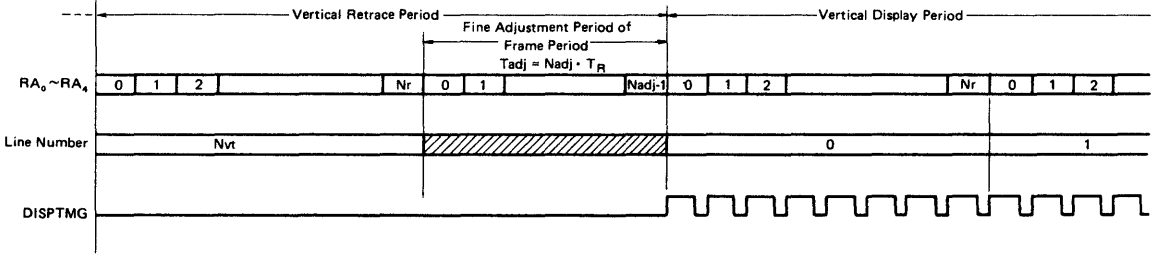


Figure 12 Fine Adjustment Period of Frame in Vertical Display
(Expansion of Fig. 10– Ⓑ)

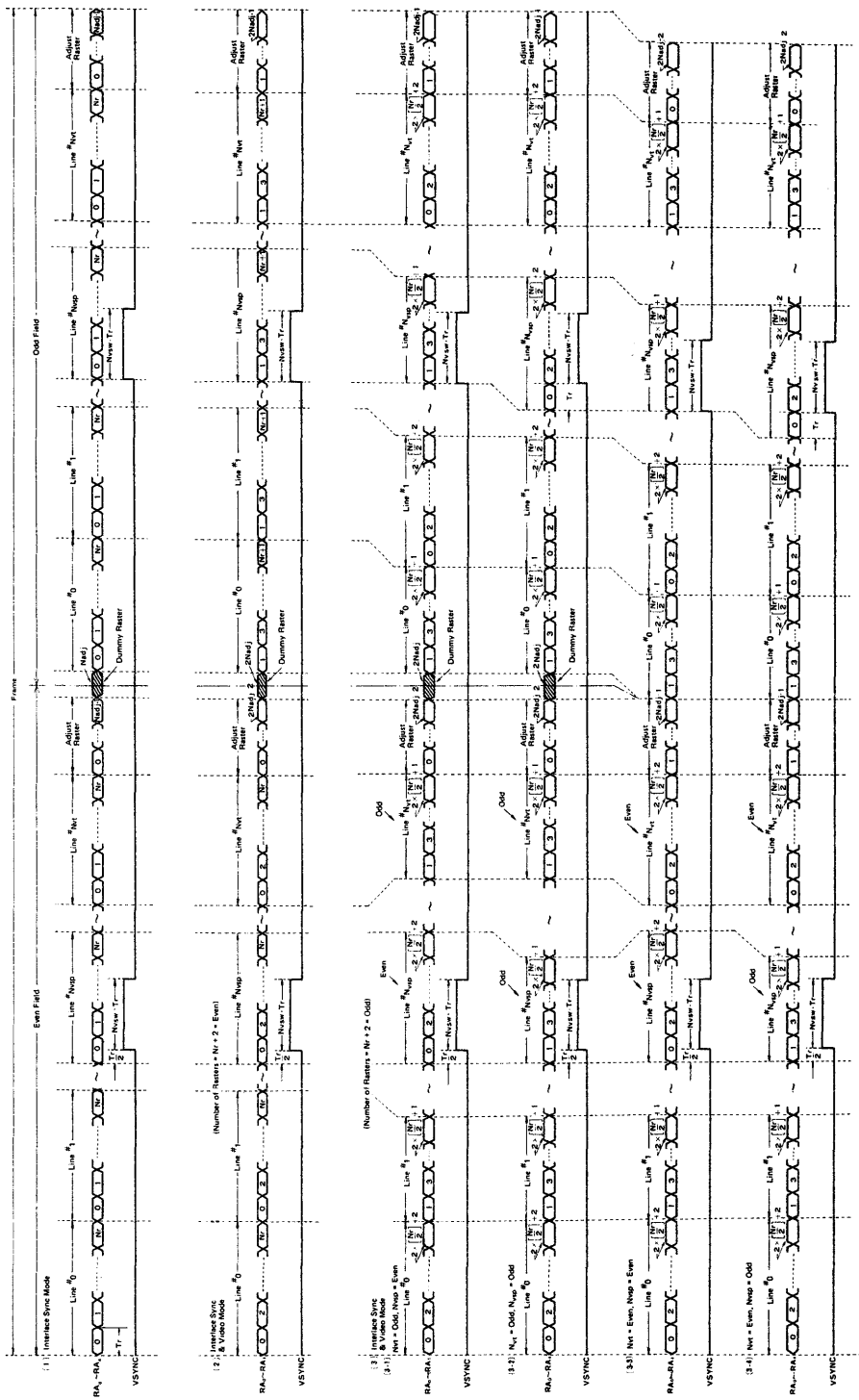


Figure 13 Interlace Control

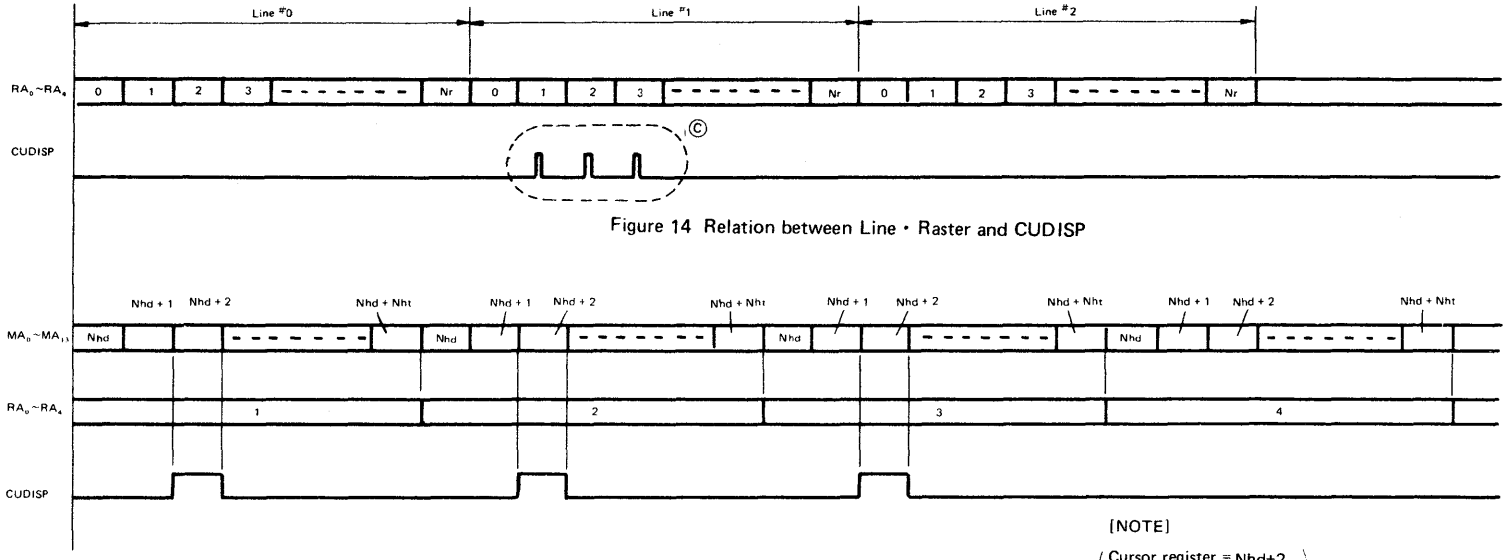


Figure 14 Relation between Line · Raster and CUDISP

[NOTE]

(Cursor register = Nhd+2
Cursor Start
Raster Register = 1
Cursor End
Raster Register = 3)

are Programmed in cursor display mode.

In blink mode, it is changed into display or non-display mode when field period is 16 or 32-time period.

Figure 15 CUDISP Output Timing (Exapnsion of Fig. 14- ©)

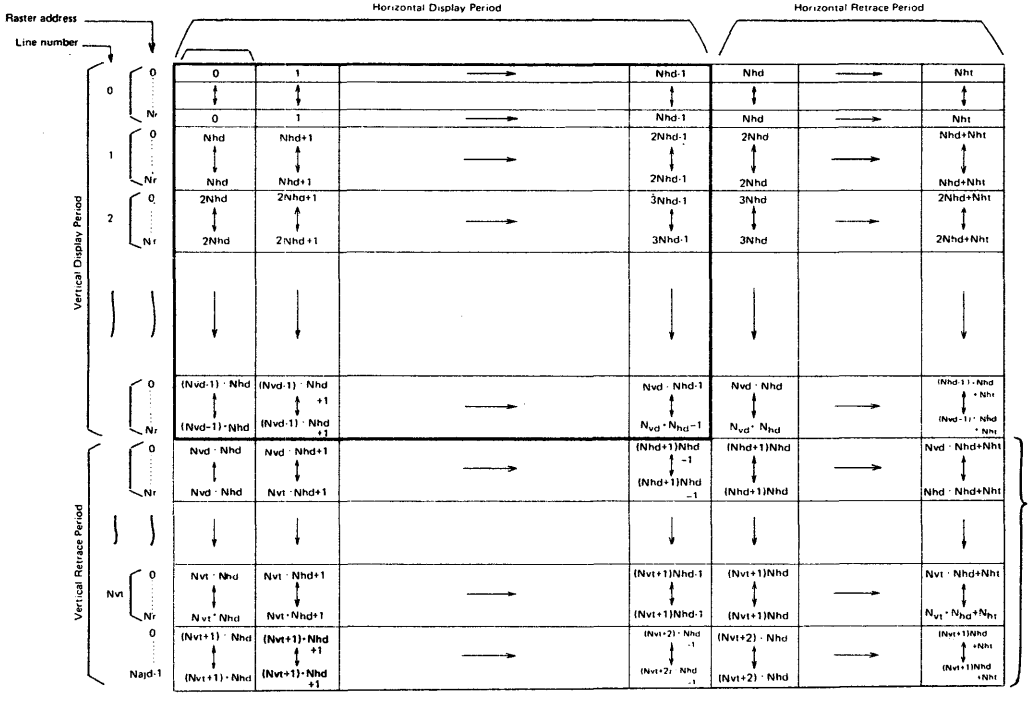


Figure 16 Refresh Memory Address (MA_0~MA_{13})

■ How to Use the CRTC

• Interface to MPU

As shown in Fig. 17, the CRTC is connected with the standard bus of MPU to control the data transfer between them. The CRTC address is determined by $\overline{CS}$ and RS, and the R/W operation is controlled by R/$\overline{W}$ and Enable signals. When $\overline{CS}$ is "Low" and RS is also "Low", the CRTC address register is selected. When $\overline{CS}$ is "Low" and RS is "High", one of 18 internal registers is selected.

$\overline{RES}$ is the system reset signal. When $\overline{RES}$ becomes "Low", the CRTC internal control logic is reset. But internal registers shown in Table 1 (R0~R17) are not affected by $\overline{RES}$ signal and remain unchanged.

The CRTC is designed so as to provide an interface to microcomputers, but adding some external circuits enables an interface to other data sources.

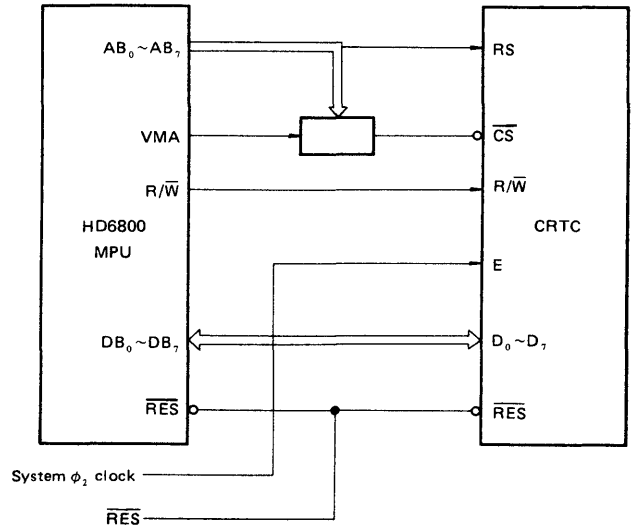


Figure 17 Interlace to MPU

• Dot Timing Generating Circuit

CRTC's CLK input (21 pin) is provided with CLK signal which defines horizontal character time period from the outside. This CLK signal is generated by dot counter shown in Fig. 18. Fig. 18 shows a example of circuit where horizontal dot number of the character is "9". Fig. 19 shows the operation time chart of dot counter shown in Fig. 18. As this example shows explicitly, CLK signal is at "Low" level in the former half of horizontal character time and at "High" level in the latter half. It is necessary to be careful so as not to mistake this polarity.

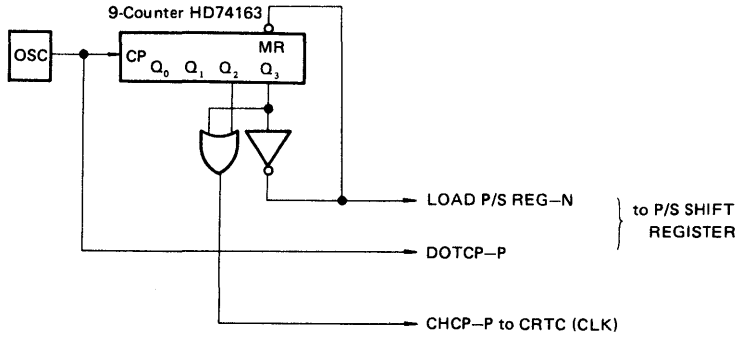


Figure 18 Dot Counter

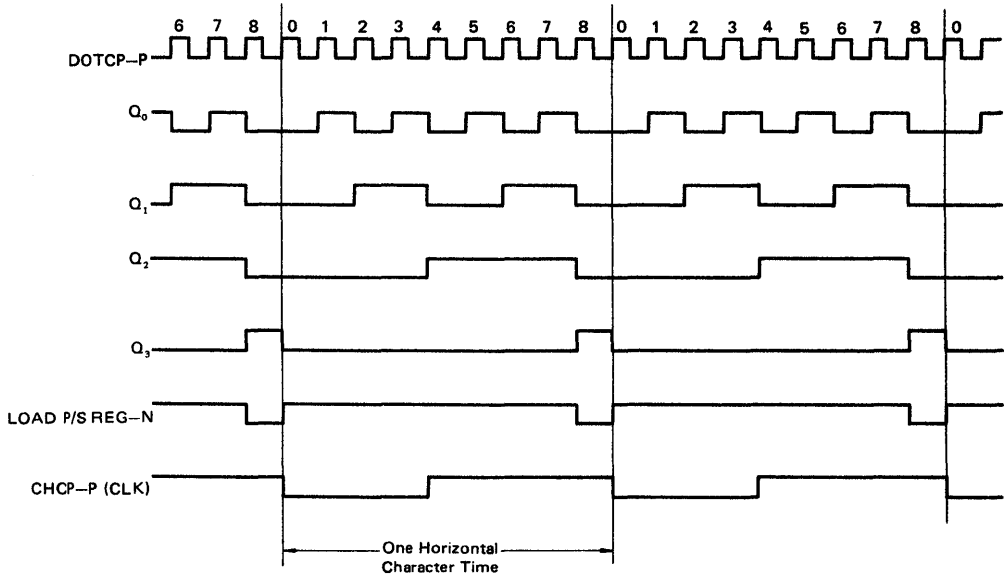


Figure 19 Time Chart of Dot Counter

■ INTERFACE TO DISPLAY CONTROL UNIT

Fig. 20 shows the interface between the CRTC and display control unit. Display control unit is mainly composed of Refresh Memory, Character Generator, and Video Control circuit. For refresh memory, 14 Memory Address line (0~16383) max are provided and for character generator, 5 Raster Address line (0~31) max are provided. For video control circuit, DISPTMG, CUDISP, HSYNC, and VSYNC signals are sent out. DISPTMG signal is used to control the blank period of video signal. CUDISP signal is used as video signal to display the cursor on the CRT screen. Moreover, HSYNC and VSYNC signals are used as drive signals respectively for CRT horizontal and vertical deflection circuits.

Outputs from video control circuit, (video signals and sync signals) are provided to CRT display unit to control the deflection and brightness of CRT, thus characters are displayed on the screen.

Fig. 21 shows detailed block diagram of display control unit. This shows how to use CUDISP and DISPTMG signals. CUDISP and DISPTMG signals should be used being latched at least one time at external flip-flop F1 and F2. Flip-flop F1 and F2 function to make one-character delay time so as to synchronize them with video signal from parallel-serial converter. High-speed D type flip-flop as TTL is used for this purpose. After being delayed at F1 and F2 DISPTMG signal is AND-ed with character video signal, and CUDISP signal is OR-ed with output from AND gate. By using this circuitry, blanking of horizontal and vertical retrace time is controlled. And cursor video is mixed with character video signal.

Fig. 21 shows the example in the case that both refresh memory and CG can be accessed for horizontal one character time. Time chart for this case is shown in Fig. 24. This method is used when a few character needed to be displayed in horizontal direction on the screen.

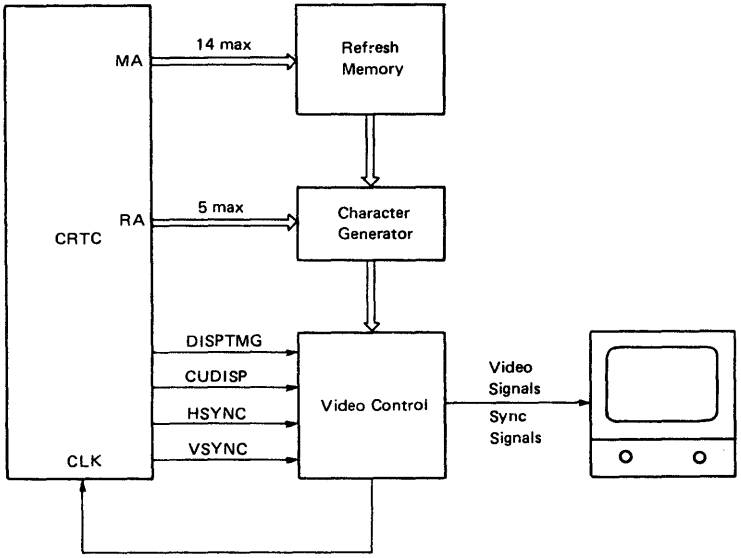


Figure 20 Interface to Display Control Unit

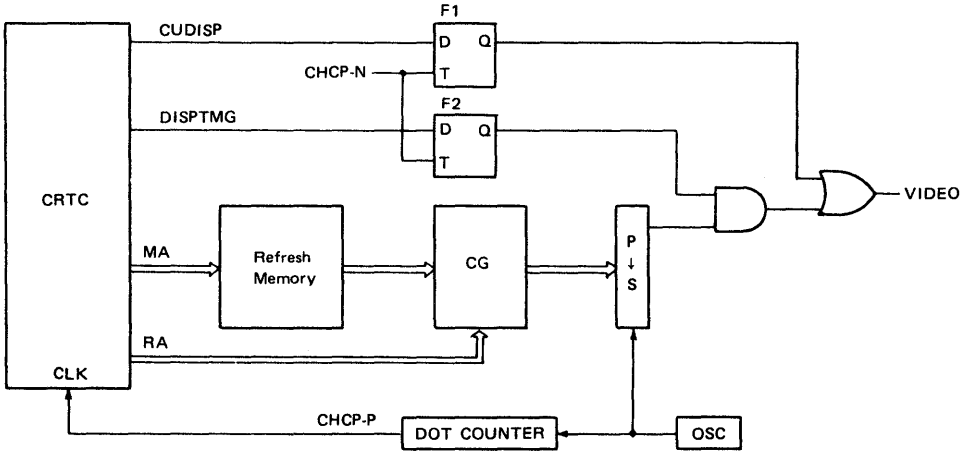


Figure 21 Display Control Unit (1)

When many characters are displayed in horizontal direction on the screen, and horizontal one-character time is so short that both refresh memory and CG cannot be accessed, the circuitry shown in Fig. 22 should be used. In this case refresh memory output shall be latched and CG shall be accessed at the next cycle. The time chart in this case is shown in Fig. 25. CUDISP and DISPTMG signals should be provided after being delayed by one-character time by using skew bit of interlace & skew register (R8). Moreover, when there are some troubles about delay time of MA during horizontal one-character time on high-speed display operation, system shown in Fig. 23 is adopted. The time chart in this case is shown in Fig. 26. Character video signal is delayed for two-character time because each MA outputs and refresh memory outputs are latched, and they are made to be in phase with CUDISP and DISPTMG signals by delaying for two-character time. Table 10 shows the circuitry selection standard of display units.

Table 10 Circuitry Standard of Display Control Unit

Case	Relation among t_{CH}, RM and CG	Block Diagram	Interlace & Skew Register Bit Programming			
			C1	C0	D1	D0
1	$t_{CH} >$ RM Access + CG Access + t_{MAD}	Fig. 21	0	0	0	0
2	RM Access + CG Access + $t_{MAD} \geqq t_{CH} >$ RM Access + t_{MAD}	Fig. 22	0	1	0	1
3	RM Access + $t_{MAD} \geqq t_{CH} >$ RM Access	Fig. 23	1	0	1	0

t_{CH} : CHCP Period; t_{MAD} : MA Delay

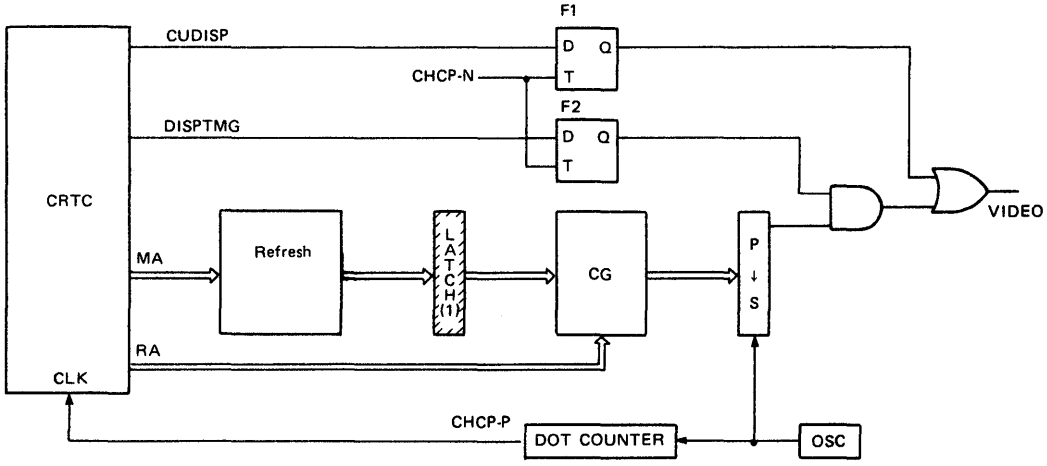


Figure 22 Display Control Unit (2)

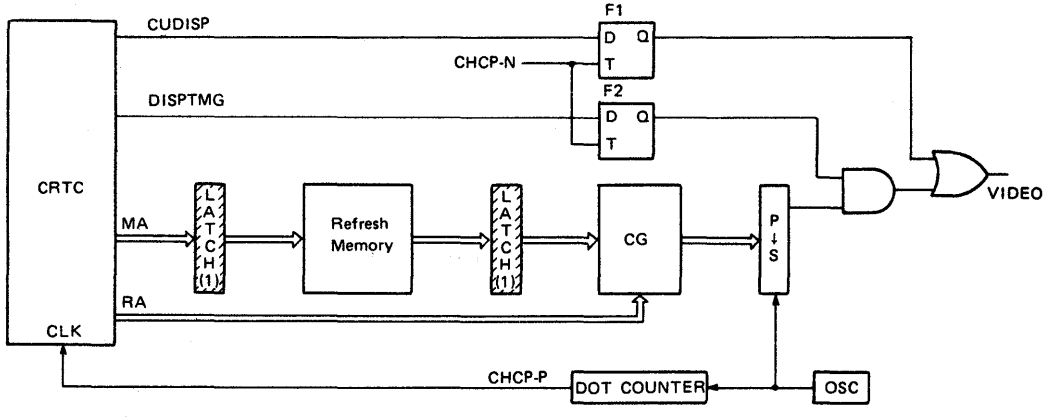


Figure 23 Display Control Unit (For high-speed display operation) (3)

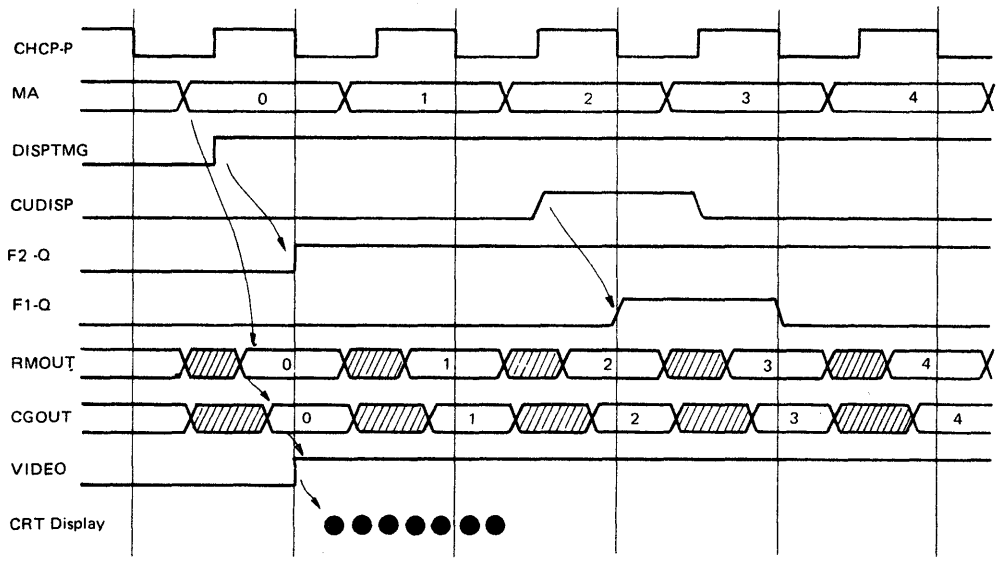


Figure 24 Time Chart of Display Control Unit (1)

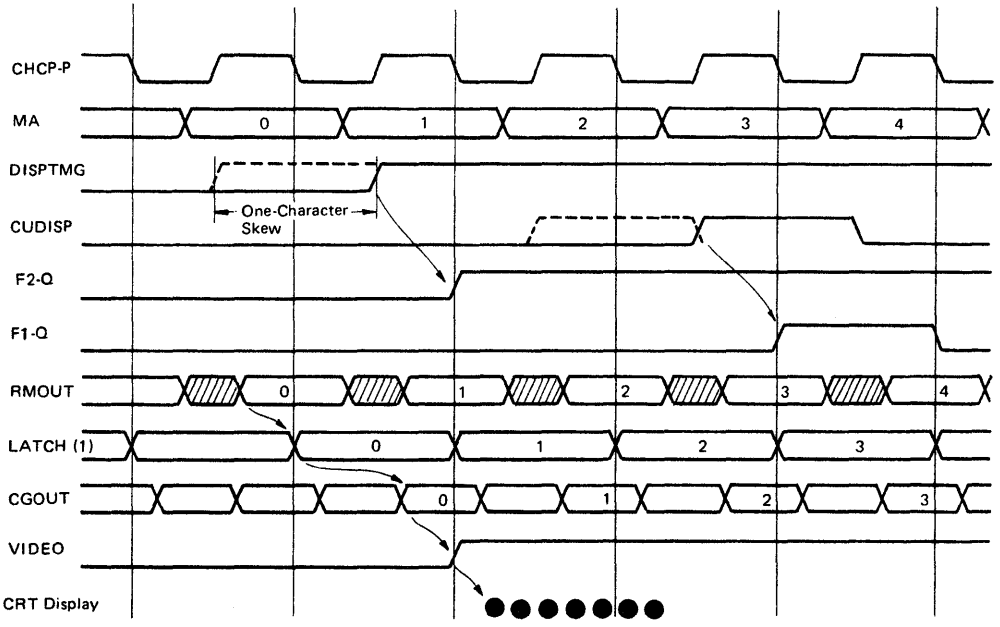


Figure 25 Time Chart of Display Control Unit (2)

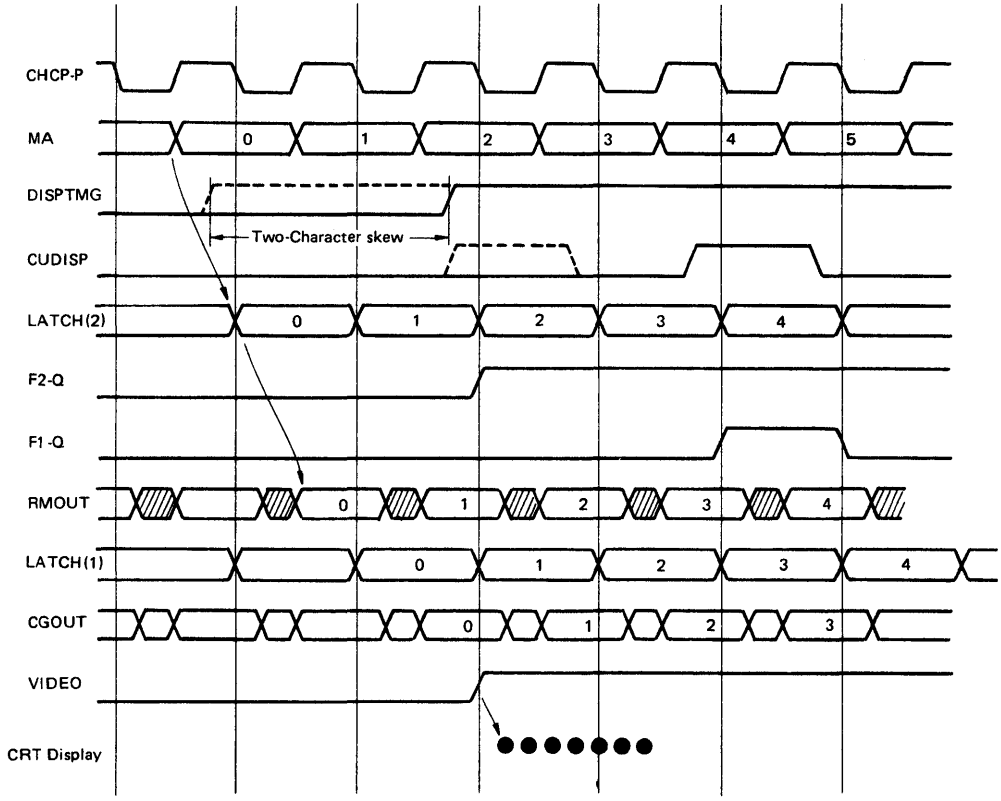


Figure 26 Time Chart of Display Unit (3)

■ HOW TO DECIDE PARAMETERS SET ON THE CRTC

• How to Decide Parameters Based on Specification of CRT Display Unit (Monitor)

Number of Horizontal Total Characters

Horizontal deflection frequency fh is given by specification of CRT display unit. Number of horizontal total characters is determined by the following equation.

$$fh = \frac{1}{tc\ (Nht + 1)}$$

where,

t_C : Cycle Time of CLK (Character Clock)
Nht : Programmed Value of Horizontal Total Register (R0)

Number of Vertical Total Characters

Vertical deflection frequency is given by specification of CRT display unit. Number of vertical Total characters is determined by the following equation.

1) Non-interlace Mode
$Rt = (Nvt + 1)(Nr + 1) + Nadj$
2) Interlace Sync Mode
$Rt = (Nvt + 1)(Nr + 1) + Nadj + 0.5$
3) Interlace Sync & Video Mode

$$Rt = \frac{(Nvt + 1)(Nr + 2) + 2Nadj}{2} \quad \cdots\cdots\cdots\cdots \text{(a)}$$

$$Rt = \frac{(Nvt + 1)(Nr + 2) + 2Nadj + 1}{2} \quad \cdots\cdots\cdots\cdots \text{(b)}$$

(a) is applied when both total numbers of vertical characters (Nvt + 1) and that of rasters in a line (Nr + 2) are odd.
(b) is applied when total number of rasters (Nr + 2) is even, or when (Nr + 2) is odd and total number of vertical characters (Nvt + 1) is even.

where,

Rt : Number of Total Rasters per frame (Including retrace period)
Nvt : Programmed Value of Vertical Total Register (R4)
Nr : Programmed Value of Maximum Raster Address Register (R9)
Nadj : Programmed Value of Vertical Total Adjust Register (R5)

Horizontal Sync Pulse Width

Horizontal sync pulse width is programmed to low order 4-bit of horizontal sync width register (R3) in unit of horizontal character time. Programmed value can be selected within from 1 to 15.

Horizontal Sync Position

As shown in Fig. 27, horizontal sync position is normally selected to be in the middle of horizontal blank period. But there are some cases where its optimum sync position is not located in the middle of horizontal blank period according to specification of CRT. Therefore, horizontal sync position should be determined by specification of CRT. Horizontal sync pulse position is programmed in unit of horizontal character time.

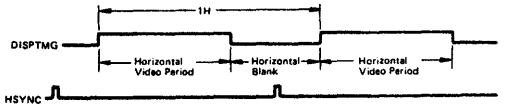


Figure 27 Time Chart of HSYNC

Vertical Sync Pulse Width

Vertical Sync Pulse Width is programmed to high order 4-bit of vertical sync pulse width register (R3) in unit of raster period. Programmed value can be selected within from 1 to 16.

Vertical Sync Position

As shown in Fig. 28, vertical sync position is normally selected to be in the middle of vertical blank period. But there are some cases where its optimum sync position is not located in the middle of vertical blank period according to specification of CRT. Therefore, vertical sync position should be determined by specification of CRT. Vertical sync pulse position is programmed to vertical sync position register (R7) in unit of line period.

- **How to Decide Parameters Based on Screen Format**

Dot Number of Characters (Horizontal)

Dot number of characters (horizontal) is determined by character font and character space. An example is shown in Fig. 29. More strictly, dot number of characters (horizontal) N is determined by external N-counter. Character space is set by means shown in Fig. 30.

Dot Number of Characters (Vertical)

Dot number of characters (vertical) is determined by characters font and line space. An example is shown in Fig. 29. Dot number of characters (vertical) is programmed to maximum raster address (register R9) of CRTC. When Nr is programmed

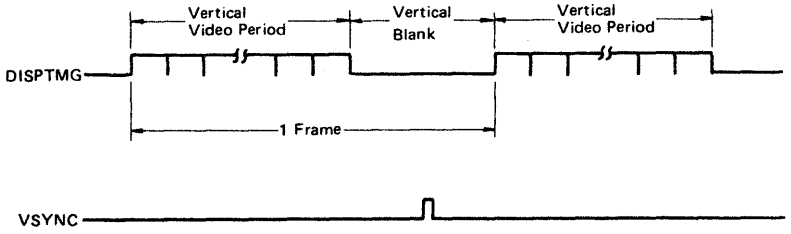


Figure 28 Time Chart of VSYNC

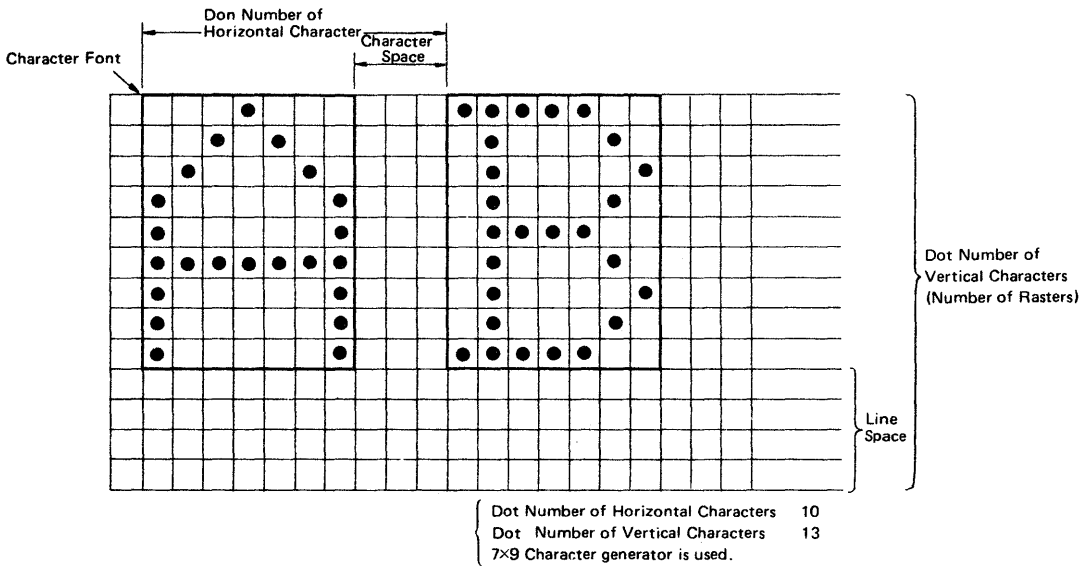


Figure 29 Dot Number of Horizontal and Vertical Characters

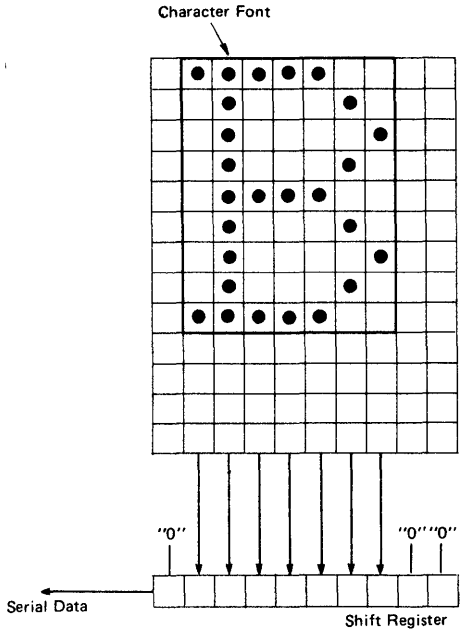


Figure 30 How to Make Character Space

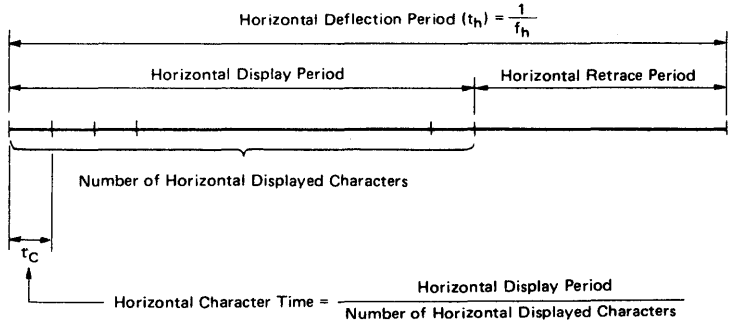


Figure 31 Number of Horizontal Displayed Characters

value of R9, dot number of characters (vertical) is (Nr+1).

Number of Horizontal Displayed Characters

Number of horizontal displayed characters is programmed to horizontal displayed register (R1) of the CRTC. Programmed value is based on screen format. Horizontal display period, which is given by specification of horizontal deflection frequency and horizontal retrace period of CRT display unit, determines horizontal character time, being divided by number of horizontal displayed characters. Moreover, its cycle time and access time which are necessary for CRT display system are determined by horizontal character time.

Number of Vertical Displayed Characters

Number of vertical displayed characters is programmed to vertical displayed register (R6). Programmed value is based on screen format. As specification of vertical deflection frequency of CRT determines number of total rasters (Rt) including vertical retrace period and the relation between number of vertical displayed character and total number of rasters on a screen is as mentioned above, CRT which is suitable for desired screen format should be selected.

For optimum screen format, it is necessary to adjust number of rasters per line, number of vertical displayed characters, and total adjust raster (Nadj) within specification of vertical deflection frequency.

Scan Mode

The CRTC can program three-scan modes shown in Table 11 to interlace mode register (R8). An example of character display in each scan mode is shown in Fig. 8.

Table 11 Program of Scan Mode

2^1	2^0	Scan Mode	Main Usage
0	0	Non-interlace	Normal Display of Characters & Figures
1	0		
0	1	Interlace Sync	Fine Display of Characters & Figures
1	1	Interlace Sync & Video	Display of Many Characters & Figures Without Using High-resolution CRT

[NOTE] In the interlace mode, the number of times per sec. in raster scanning on one spot on the screen is half as many as that in non-interlace mode. Therefore, when persistence of luminescence is short, flickering may happen. It is necessary to select optimum scan mode for the system, taking characteristics of CRT, raster scan speed, and number of displayed characters and figures into account.

Cursor Display Method

Cursor start raster register and cursor end raster register (R10, R11) enable programming the display modes shown in Table 7 and display patterns shown in Fig. 9. Therefore, it is possible to change the method of cursor display dynamically according to the system conditions as well as to realize the cursor display that meets the system requirements.

Start Address

Start address resisters (R12, R13) give an offset to the address of refresh memory to read out. This enables paging and scrolling easily.

Cursor Register

Cursor registers (R14, R15) enable programming the cursor display position on the screen. As for cursor address, it is not X, Y address but linear address that is programmed.

■ Applications of the CRTC

● Monochrome Character Display

Fig. 32 shows a system of monochrome character display. Character clock signal (CLK) is provided to the CRTC through OSC and dot counter. It is used as basic clock which drives internal control circuits. MPU is connected with the CRTC by standard bus and controls the CRTC initialization and READ/WRITE of internal registers.

Refresh memory is composed of RAM which has capacity of one frame at least and the data to be displayed is coded and stored. The data to refresh memory is changed through MPU

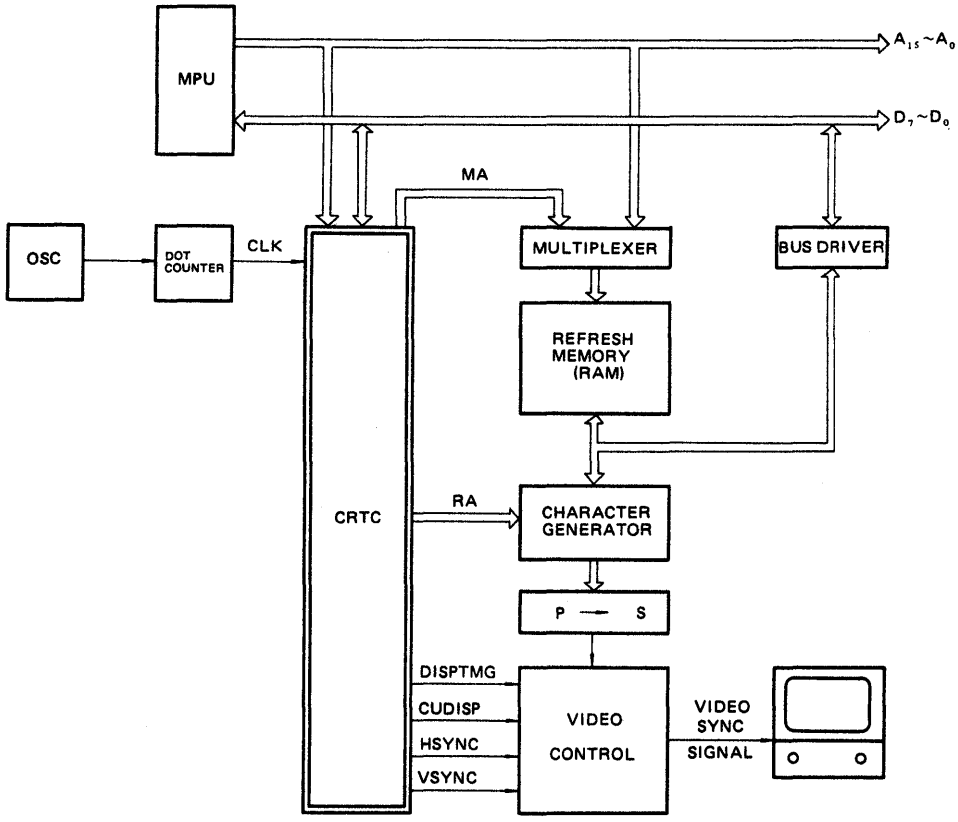


Figure 32 Monochrome Character Display

bus, while refresh memory is read out successively by the CRTC to display a static pattern on the screen. Refresh memory is accessed by both MPU and the CRTC, so it needs to change its address selectively by multiplexer. The CRTC has 14 MA (Memory Address output), but in fact some of them that are needed are used according to capacity of refresh memory.

Code output of refresh memory is provided to character generator. Character generator generates a dot pattern of a specified raster of a specified character in parallel according to code output from refresh memory and RA (Raster Address output) from the CRTC. Parallel-serial converter is normally composed of shift register to convert output of character generator into a serial dot pattern. Moreover, DISPTMG, CUDISP, HSYNC, and VSYNC signals are provided to video control circuit. It controls blanking for output of parallel-serial converter, mixes these signals with cursor video signal, and generates sync signals for an interface to monitor.

- **Color Character Display**

Fig. 33 shows a system of color character display. In this example, a 3-bit color control bit (R, G, B) is added to refresh memory in parallel with character code and provided to video control circuit. Video control circuit controls coloring as well as blanking and provides three primary color video signals (R, G, B signals) to CRT display device to display characters in seven kinds of color on the screen.

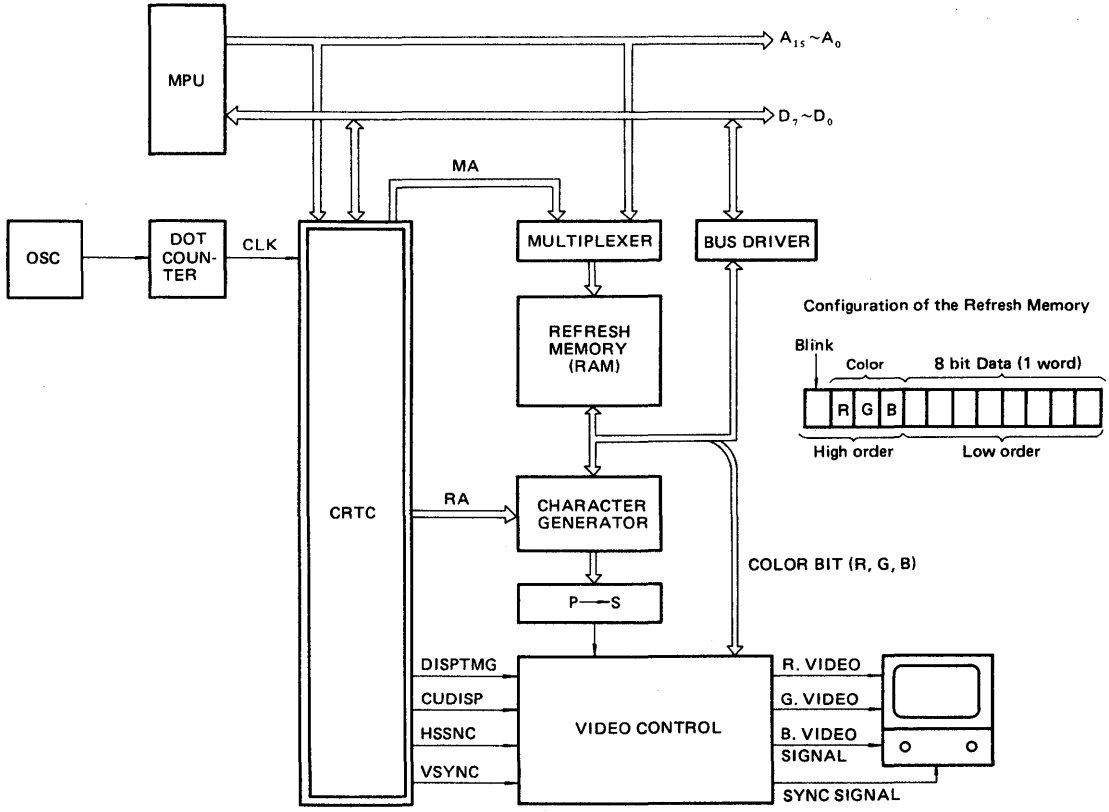


Figure 33 Color Character Display

- **Color limited Graphic Display**

Limited graphic display is to display simple figures as well as character display by combination of picture element which are defined in unit of one character.

As shown in Fig. 34, graphic pattern generator is set up in parallel with character generator and output of these generators are wire-ORed. Which generator is accessed depends on coded output of refresh memory.

In this example, graphic pattern generator adopts ROM, so only the combination of picture elements which are programmed to it is used for this graphic display system. Adopting RAM instead of ROM enables dynamically writable symbols in any combination on one display by changing the contents of them.

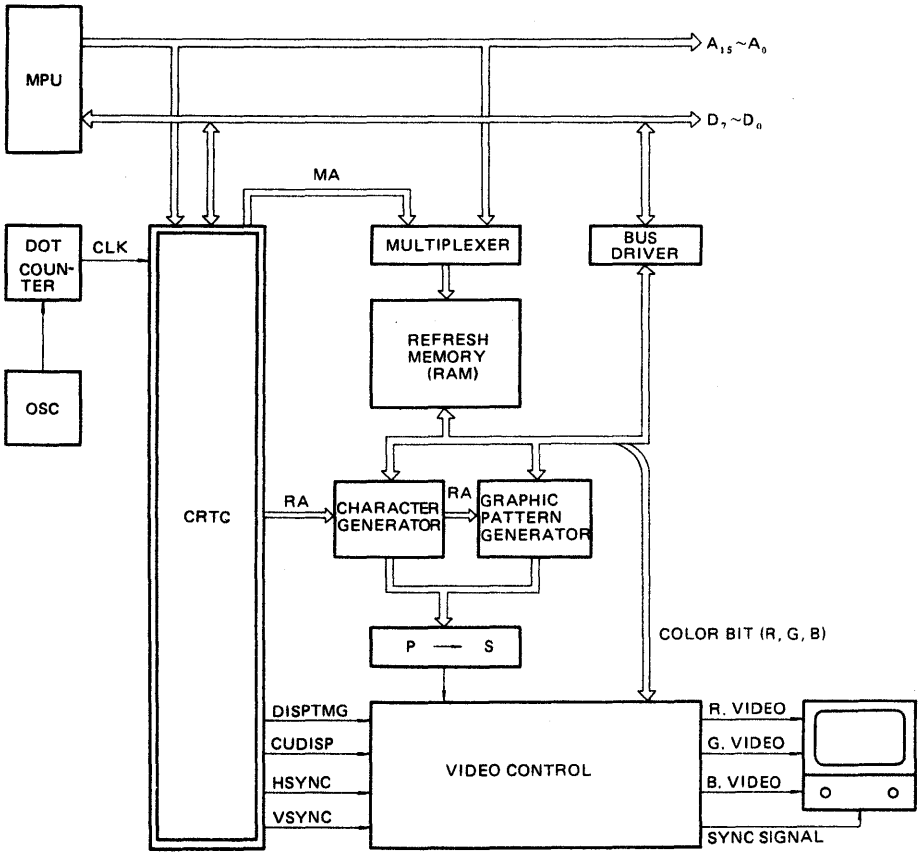


Figure 34 Color Limited Graphic Display

- **Monochrome Full Graphic Display**

Fig. 35 shows a system of monochrome full graphic display. While simple graphic display is figure display by combination of picture elements in unit of 1 picture elements, full graphic display is display of any figures in unit of 1 dot. In this case, refresh memory is dot memory that stores all the dot patterns, so its output is directly provided to parallel-serial converter to be displayed. Dot memory address to refresh the screen is set up by combination of MA output and RA output of CRTC.

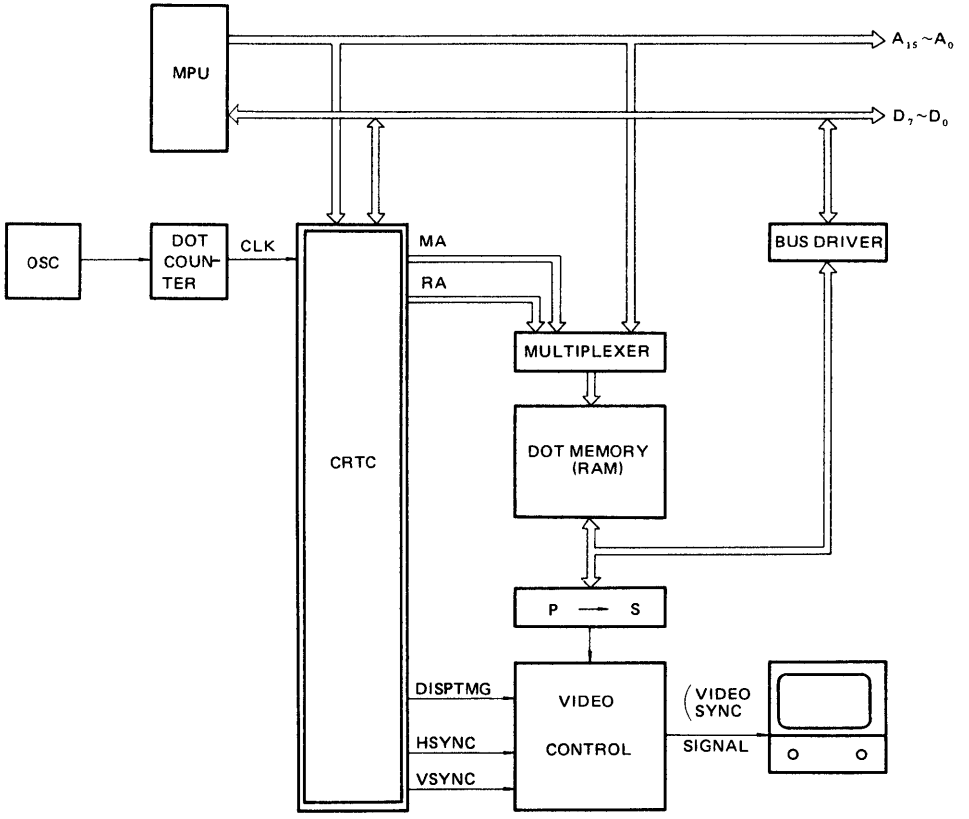


Figure 35 Monochrome Full Graphic Display

Fig. 36 shows an example of access to refresh memory by combination of MA output and RA output. Fig. 36 shows a refresh memory address method for full graphic display. Correspondence between dot on the CRT screen and refresh memory address is shown in Fig. 37.

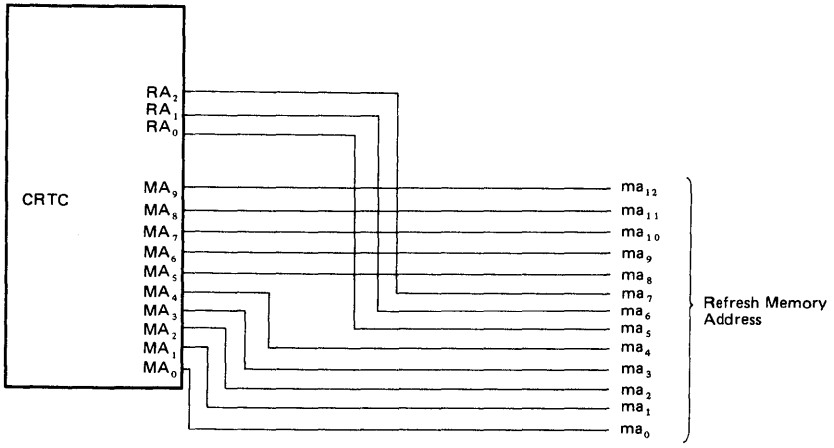


Figure 36 Refresh Memory Address Method for Full Graphic Display

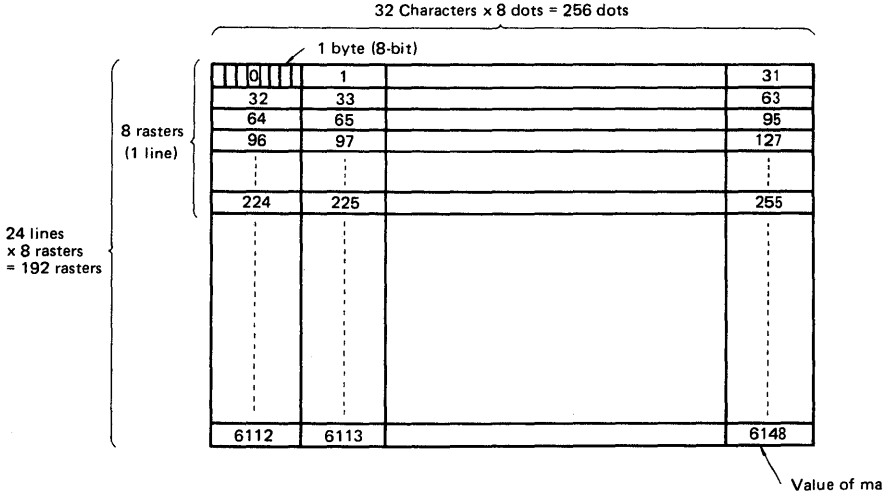


Figure 37 Memory Address and Dot Display Position on the Screen for Full Graphic Display

- **Color Full Graphic Display**

Fig. 38 shows a system of color full graphic display by 7-color display. Refresh memory is composed of three dot memories which are respectively used for red, green, and blue. These dot memories are read out in parallel at one time and their output is provided to three parallel-serial converters. Then video control circuit adds the blanking control to output of these converters and provides it to CRT display device as red, green, and blue video signals with sync signals.

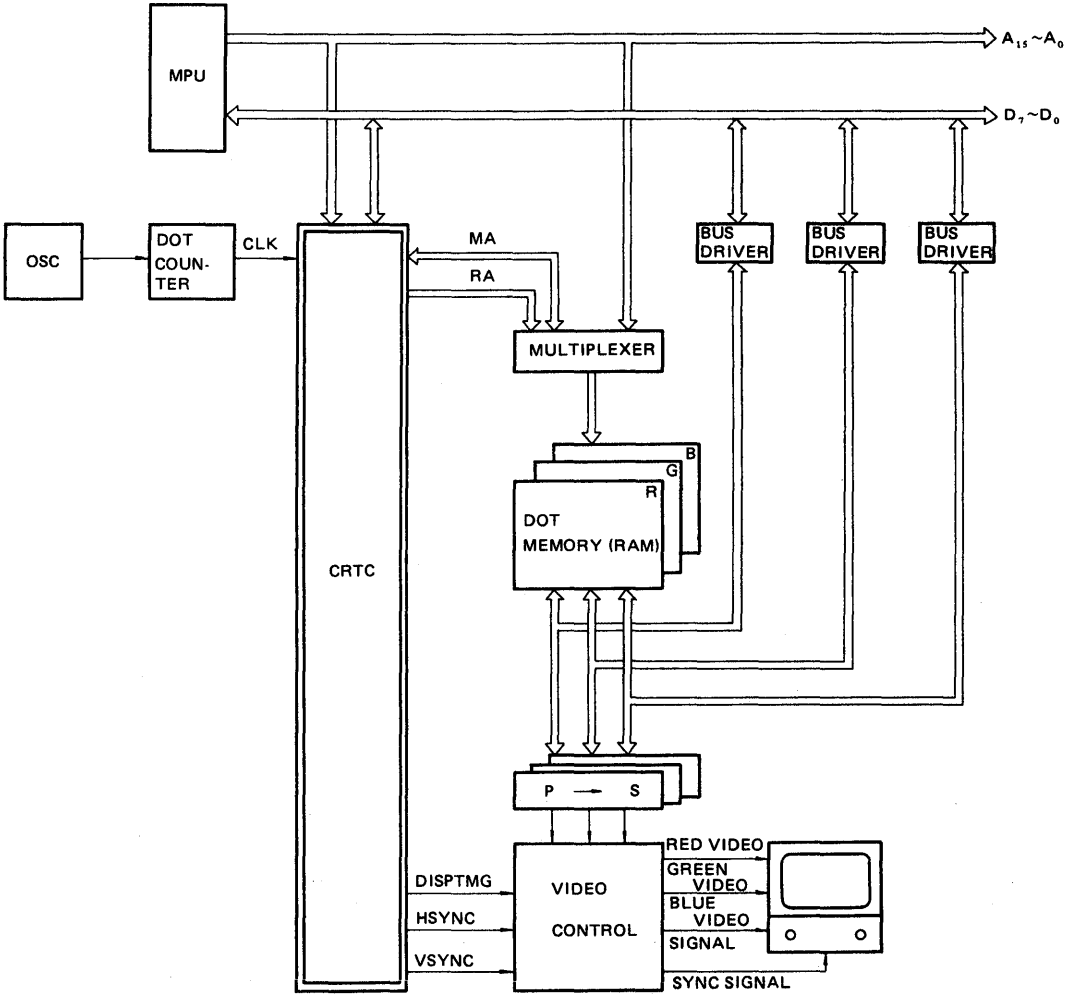


Figure 38 Color Full Graphic Display

- **Cluster Control of CRT Display**

The CRTC enables cluster control that is to control CRT display of plural devices by one CRTC. Fig. 39 shows a system of cluster control. Each display control unit has refresh memory, character generator, parallel-serial converter, and video control circuit separately, but these are controlled together by the CRTC.

In this system, it is possible for plural CRT display devices to have their own display separately.

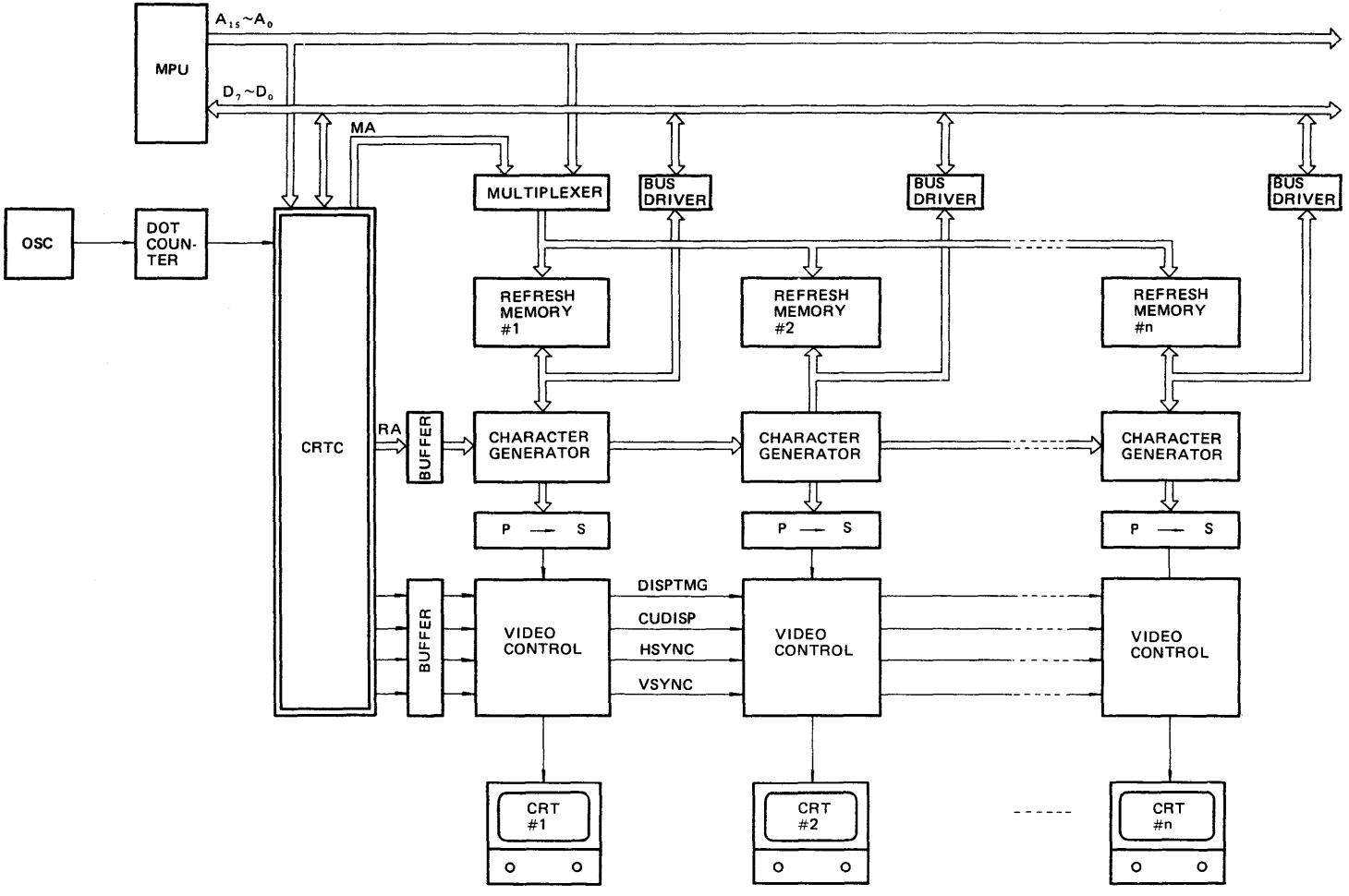


Figure 39 Cluster Control by the CRTC

■ EXAMPLES OF APPLIED CIRCUIT OF THE CRTC

Fig. 41 shows an example of application of the CRTC to monochrome character display. Its specification is shown in Table 12. Moreover, specification of CRT display unit is shown in Table 13 and initializing values for the CRTC are shown in Table 14.

Table 12 Specification of Applied Circuit

Item	Specification
Character Format	5 × 7 Dot
Character Space	Horizontal : 3 Dot Vertical : 5 Dot
One Character Time	1 μs
Number of Displayed Characters	40 characters × 16 lines = 640 characters
Access Method to Refresh Memory	Snychronous Method (DISPTMG Read)
Refresh Memory	1 kB
Address Map	2^{15} 2^{14} 2^{13} 2^{12} 2^{11} 2^{10} 2^{9} 2^{8} 2^{7} 2^{6} 2^{5} 2^{4} 2^{3} 2^{2} 2^{1} 2^{0} Refresh Memory: 0 0 0 0 0 0 * * * * * * * * * * CRTC Address Register: 0 0 0 1 0 0 × × × × × × × × × 0 CRTC Control Register: 0 0 0 1 0 0 × × × × × × × × × 1 × · · · don't care, * · · · 0 or 1
Synchronization Method	HVSYNC Method

Table 13 Specification of Character Display

Item	Specification
Scan Mode	Non-interlace
Horizontal Deflection Frequency	15.625 kHz
Vertical Deflection Frequency	60.1 Hz
Dot Frequency	8 MHz
Character Dot (Horizontal × Vertical)	8 × 12 (Character Font 5 × 9)
Number of Displayed Characters (Row × Line)	40 × 16
HSYNC Width	4 μs
VSYNC Width	3 H
Cursor Display	Raster 9 ~ 10, Blink 16 Field Period
Paging, Scrolling	Not used

Table 14 Initializing Values for Character Display

Register	Name	Symbol	Initializing Value Hex (Decimal)
R0	Horizontal Total	Nht	3F (63)
R1	Horizontal Displayed	Nhd	28 (40)
R2	Horizontal Sync Position	Nhsp	34 (52)
R3	Sync Width	Nvsw, Nhsw	34
R4	Vertical Total	Nvt	14 (20)
R5	Vertical Total Adjust	Nadj	08 (8)
R6	Vertical Displayed	Nvd	10 (16)
R7	Vertical Sync Position	Nvsp	13 (19)
R8	Interlace & Skew		00
R9	Maximum Raster Address	Nr	0B (11)
R10	Cursor Start Raster	B, P, NCSTART	49
R11	Cursor End Raster	NCEND	0A (10)
R12	Start Address (H)		00 (0)
R13	Start Address (L)		00 (0)
R14	Cursor (H)		00 (0)
R15	Cursor (L)		00 (0)

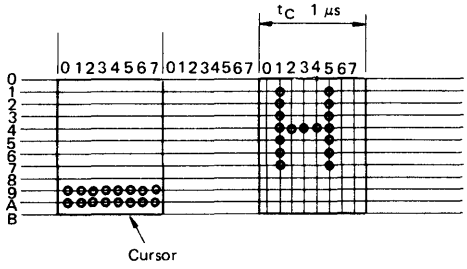


Figure 40 Non-interlace Display (Example)

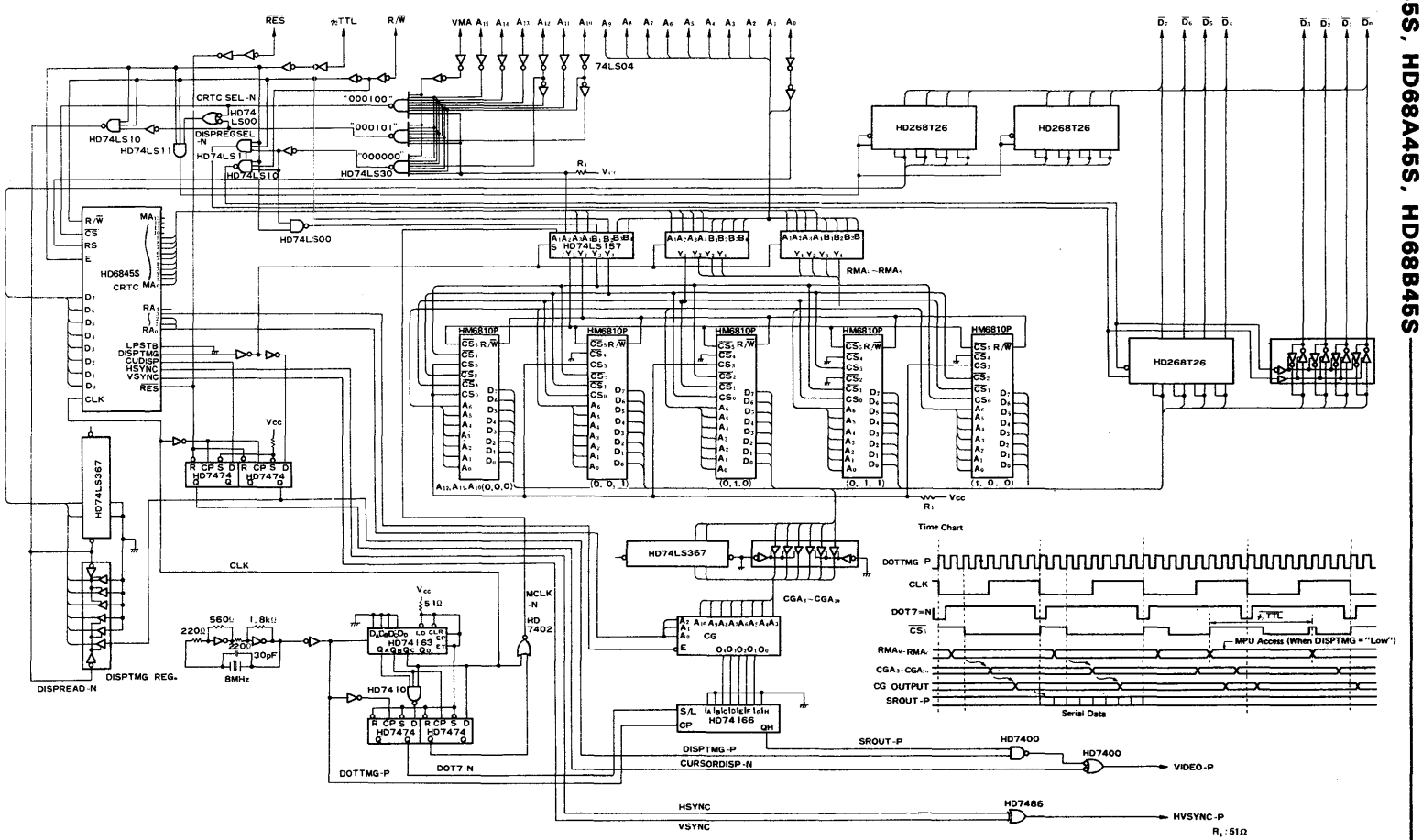
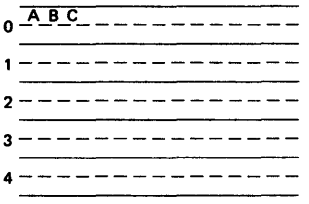
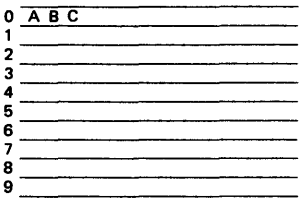
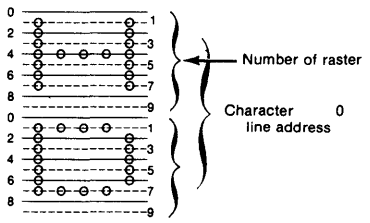
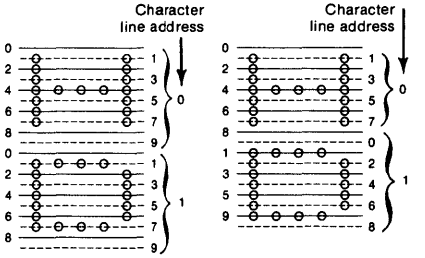
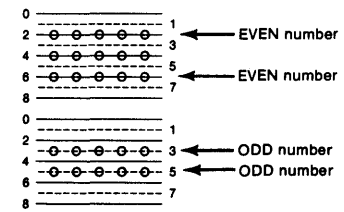
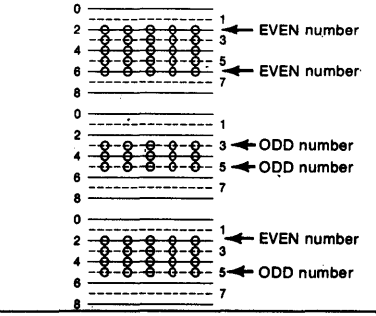
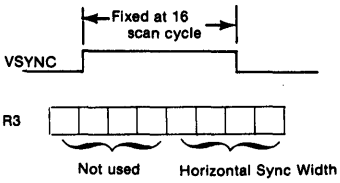
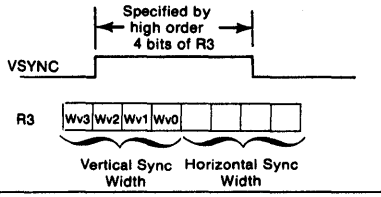
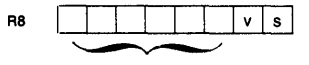
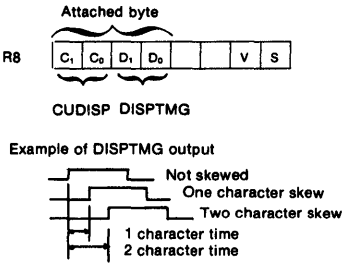


Figure 41 Example of Applied Circuit of the CRTC (Monochrome Character Display)

Differences between the HD6845R (Motorola MC6845 Compatible) and the HD6845S (Enhanced)

No.	Functional Difference		HD6845R	HD6845S
1	Interlace Sync & Video Mode Display	Programming Method of number of vertical characters	Character line address  In HD6845R, number of characters is vertically programmed in units of two lines, as illustrated above. (Number of vertical total characters, Number of vertical displayed characters, Vertical Sync Position) Example of above figure . . . Programmed number into Vertical Displayed Register = 5	Character line address  In HD6845S, number of characters is vertically programmed in unit of one line, as illustrated above. (Number of vertical total characters, Number of vertical displayed characters, Vertical Sync Position) Example of above figure . . . Programmed number into Vertical Displayed Register = 10
		Number of raster per character line	Only even number can be specified.  Number of raster = 10 scanline (specified) However, number which is programmed into register is calculated as follows. Programmed number (Nr) = (Number specified) − 1	Both even number and odd number can be specified.  When number of raster per character line is EVEN Number of raster = 10 scan line (specified) When number of raster per character line is ODD. Number of raster = 9 scan line (specified) However, number which is programmed into register is calculated as follows. Programmed number (NR) = (Number specified) − 2
		Cursor Display	Cursor is displayed in either EVEN field or ODD field. 	Cursor is displayed in both EVEN field and ODD field. 

No.	Functional Difference	HD6845R	HD6845S
2	Vertical Sync Pulse Width (VSYNC output)	Fixed at 16 raster scan cycle (16H) 	Programmable (1 - 16 raster scan cycle) 
3	SKEW Function	Not included 	SKEW capability is included in DISPTMG, CUDISP signals. 
4	Start Address Register	Write Only	Read or Write
5	RESET Signal (RES)	MA_0 ~ M_{13} Output, RA_0 ~ RA_4 Output } ------Synchronous reset Other Outputs --------Asynchronous reset Output signals of MA_0 ~ MA_{13}, RA_0 ~ RA_4, synchronized with CLK "LOW" level, go to "LOW" level, after RES has gone to "LOW." Other outputs go to "LOW" immediately after RES has gone to "LOW" level.	MA^0 ~ MA_{13} Output, RA_0 ~ RA_4 Output, Other Outputs } ---- Asynchronous reset Output signals of MA_0 ~ MA_{13}, RA_0 ~ RA_4 and others go to "LOW" level immediately after RES has gone to "LOW" level.

AC Characteristic Differences between HD6845R (Motorola MC6845 Compatible) and HD6845S (Enhanced)

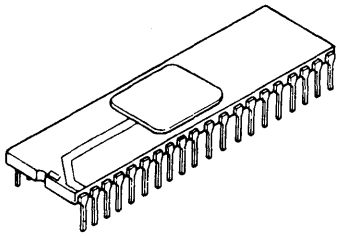
No.	Characteristic Difference	Symbol	HD6845R			HD6845S			Unit
			min.	typ.	max.	min.	typ.	max.	
1	Clock Cycle Time	*tcycc*	330	—	—	270	—	—	ns
2	Clock Pulse Width "High"	PW_{CH}	150	—	—	130	—	—	ns
3	Clock Pulse Width "Low"	PW_{CL}	150	—	—	130	—	—	ns
4	Rise and Fall Time for Clock Input	T_{CR}, T_{CF}	—	—	15	—	—	20	ns
5	Horizontal Sync Delay Time	T_{HSD}	—	—	250	—	—	200	ns
6	Light Pan Strobe Pulse Width	PW_{LPH}	80	—	—	60	—	—	ns
7	Light Pan Strobe, Uncertain Time of Acceptance	T_{LPD1}	—	—	80	—	—	70	ns
		T_{LPD2}	—	—	10	—	—	0	ns

HD6846
COMBO (Combination ROM I/O Timer)

The HD6846 combination chip provides the means, in conjunction with the HD6802, to develop a basic 2-chip microcomputer system. The HD6846 consists of 2048 bytes of mask-programmable ROM, an 8-bit bidirectional data port with control lines, and a 16-bit programmable timer-counter.

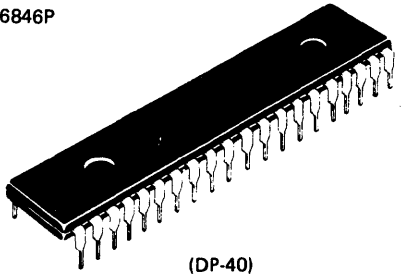
This device is capable of interfacing with the HD6802 (basic HD6800, clock and 128 bytes of RAM) as well as the HD6800 if desired. No external logic is required to interface with most peripheral devices.

HD6846



(DC-40)

HD6846P

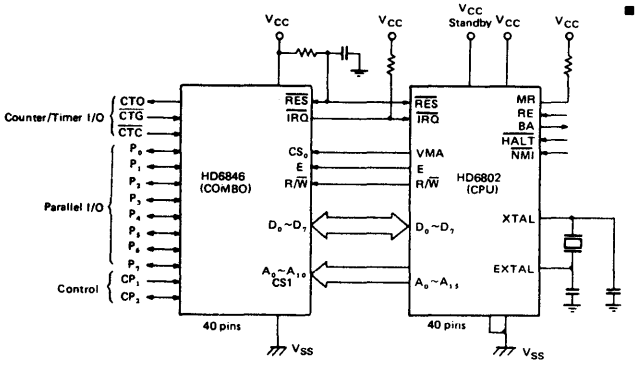


(DP-40)

■ FEATURES

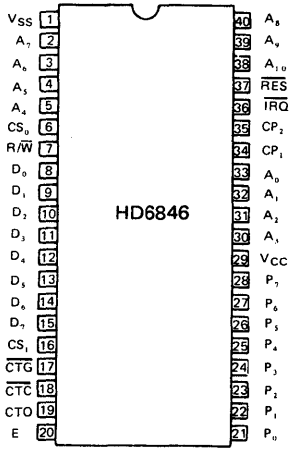
- 2048 8-Bit Bytes of Mask-Programmable ROM
- 8-Bit Bidirectional Data Port for Parallel Interface plus Two Control Lines
- Programmable Interval Timer-Counter Functions
- Programmable I/O Peripheral Data, Control and Direction Registers
- Compatible with the Complete HMCS6800 Microcomputer Product Family
- TTL-Compatible Data and Peripheral Lines
- Single 5-Volt Power Supply
- Compatible with MC6846

■ TYPICAL MICROCOMPUTER



This is a block diagram of a typical cost effective microcomputer. The MPU is the center of the microcomputer system and is shown in a minimum system interfacing with a ROM combination chip. It is not intended that this system be limited to this function but that it be expandable with other parts in the HMCS6800 Microcomputer family.

■ PIN ARRANGEMENT



(Top View)

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC}*	-0.3 ~ +7.0	V
Input Voltage	V_{in}*	-0.3 ~ +7.0	V
Operating Temperature	T_{opr}	-20 ~ +75	°C
Storage Temperature	T_{stg}	-55 ~ +150	°C

* With respect to V_{SS} (SYSTEM GND)

(NOTE) Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ RECOMMENDED OPERATING CONDITIONS

Item	Symbol	min	typ	max	Unit
Supply Voltage	V_{CC}*	4.75	5.0	5.25	V
Input Voltage	V_{IL}*	-0.3	–	0.8	V
	V_{IH}*	2.0	–	V_{CC}	V
Operating Temperature	T_{opr}	-20	25	75	°C

* With respect to V_{SS} (SYSTEM GND)

■ ELECTRICAL CHARACTERISTICS

● DC CHARACTERISTICS (V_{CC}=5.0V±5%, V_{SS}=0V, Ta=-20~+75°C, unless otherwise noted.)

Item		Symbol	Test Condition	min.	typ.	max.	Unit
Input "High" Voltage	All Inputs	V_{IH}		2.0	–	V_{CC}	V
Input "Low" Voltage	All Inputs	V_{IL}		-0.3	–	0.8	V
Clock Overshoot/Undershoot	Input "High" Level	V_{OS}		V_{CC} −0.5	–	V_{CC} +0.5	V
	Input "Low" Level			V_{SS} −0.5	–	V_{SS} +0.5	
Input Leakage Current	R/$\overline{W}$, $\overline{RES}$, $\overline{CS_0}$, $\overline{CS_1}$, CP_1, $\overline{CTG}$, $\overline{CTC}$, E, A_0~A_{10}	I_{in}	V_{in} = 0 ~ 5.25V	-2.5	–	2.5	μA
Three-State (Off State) Input Current	D_0~D_7, P_0~P_7, CP_2	I_{TSI}	V_{in} = 0.4 ~ 2.4V	-10	–	10	μA
Output "High" Voltage	D_0~D_7	V_{OH}	I_{OH} = -205μA	2.4	–	–	V
	CP_2, P_0~P_7		I_{OH} = -200μA	2.4	–	–	V
	CTO		I_{OH} = -200μA	2.4	–	–	V
Output "Low" Voltage	D_0~D_7	V_{OL}	I_{OL} = 1.6mA	–	–	0.4	V
	Other Outputs		I_{OL} = 3.2mA	–	–	0.4	V
Output "High" Current (Sourcing)	D_0~D_7	I_{OH}	V_{OH} = 2.4V	-205	–	–	μA
	CTO, CP_2, P_0~P_7		V_{OH} = 2.4V	-200	–	–	μA
Output "High" Current (Sourcing) (the current for driving other than TTL, e.g., Darlington Base)	CP_2, P_0~P_7	I_{OH}	V_{OH} = 1.5V	-1.0	–	−10	mA
Output "Low" Current (Sinking)	D_0~D_7	I_{OL}	V_{OL} = 0.4V	1.6	–	–	mA
	Other Outputs			3.2	–	–	
Output Leakage Current (Off State)	$\overline{IRQ}$	I_{LOH}	V_{OH} = 2.4V	–	–	10	μA
Power Dissipation		P_D		–	–	800	mW
Capacitance	E	C_{in}	V_{CC} = 0V V_{in} = 0V T_a = 25°C f = 1MHz	–	–	20	pF
	D_0~D_7	C_{in}		–	–	12.5	pF
	P_0~P_7, CP_2, CTO	C_{out}		–	–	10	pF
	A_0~A_{10}, R/$\overline{W}$	C_{in}		–	–	7.5	pF
	$\overline{RES}$, $\overline{CS_0}$, CS_1, CP_1, $\overline{CTG}$	C_{in}		–	–	10	pF
	$\overline{IRQ}$	C_{out}		–	–	7.5	pF

- AC CHARACTERISTICS (V_{CC}=5.0V±5%, V_{SS}=0V, Ta=-20~+75°C, unless otherwise noted.)

1. BUS TIMING

Item	Symbol	Test Condition	min	typ	max	Unit
Enable Cycle Time	t_{cycE}	Fig. 1	1.0	–	10	μs
Enable Pulse Width, "Low"	PW_{EL}		430	–	4500	ns
Enable Pulse Width, "High"	PW_{EH}		430	–	4500	ns
Address Set Up Time	t_{AS}		140	–	–	ns
Data Delay Time	t_{DDR}		–	–	320	ns
Data Hold Time	t_H		10	–	–	ns
Address Hold Time	t_{AH}		10	–	–	ns
Enable Rise and Fall Time	t_{Ef},t_{Er}		–	–	25	ns
Data Set Up Time	t_{DSW}		195	–	–	ns
Reset "Low" Time	t_{RL}		2	–	–	μs
Interrupt Release Time	t_{IR}	Fig. 2	–	–	1.6	μs

2. PALLAREL PERIPHERAL I/O LINE TIMING

Item	Symbol	Test Condition	min	typ	max	Unit
Peripheral Data Setup Time	t_{PDSU}	Fig. 3	200	–	–	ns
Rise and Fall Times CP_1, CP_2	t_{Pr}, t_{pf}	Fig. 5	–	–	1.0	μs
Delay Time E to CP_2 Fall	t_{CP2}	Fig. 4	–	–	1.0	μs
Delay Time I/O Data CP_2 Fall	t_{DC}		20	–	–	ns
Delay Time E to CP_2 Rise	t_{RS1}		–	–	1.0	μs
Delay Time CP_1 to CP_2 Rise	t_{RS2}	Fig. 5	–	–	2.0	μs
Peripheral Data Delay	t_{PDW}	Fig. 4	–	–	1.0	μs
Peripheral Data Setup Time for Latch	t_{PSU}	Fig. 9	100	–	–	ns
Peripheral Data Hold Time for Latch	t_{PDH}		15	–	–	ns

3. TIMER/COUNTER LINE TIMING

Item	Symbol	Test Condition	min	typ	max	Unit
$\overline{CTC}$, $\overline{CTG}$ Rise and Fall Time	t_{Cr}, t_{Cf}	Fig. 6	–	–	100	ns
$\overline{CTC}$, $\overline{CTG}$ Pulse Width, "High" (Asynchronous Mode)	t_{PWH}		t_{cycE} +250	–	–	ns
$\overline{CTC}$, $\overline{CTG}$ Pulse Width, "Low" (Asychronous Mode)	t_{PWL}		t_{cycE} +250	–	–	ns
$\overline{CTC}$, $\overline{CTG}$ Setup Time (Synchronous Mode)	t_{su}	Fig. 7	200	–	–	ns
$\overline{CTC}$, $\overline{CTG}$ Hold Time (Synchronous Mode)	t_{hd}		50	–	–	ns
CTO Delay Time	t_{CTO}	Fig. 8	–	–	1.0	μs

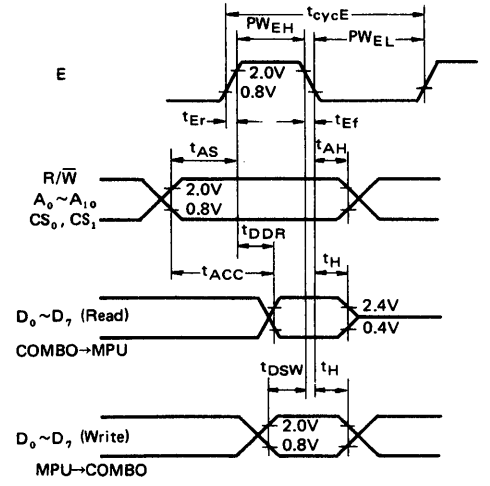


Figure 1 Bus Read/Write Timing

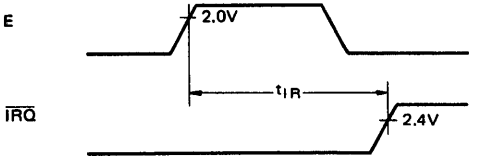


Figure 2 $\overline{IRQ}$ Release Time

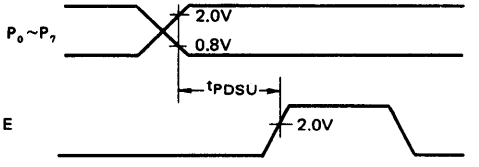


Figure 3 Peripheral Data Set Up Time

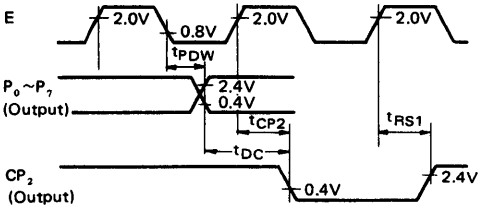


Figure 4 Peripheral Data and CP_2 (Output) Delay Time

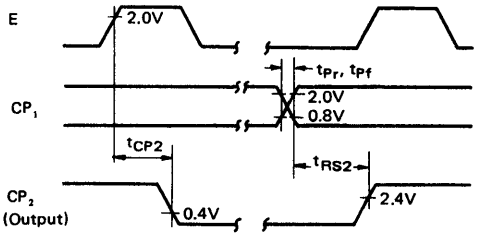


Figure 5 CP_2 (Output) Delay Time

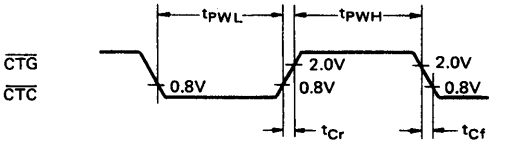


Figure 6 $\overline{CTG}$, $\overline{CTC}$ Pulse Width

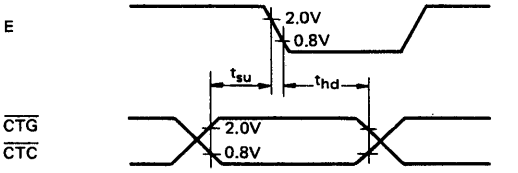


Figure 7 $\overline{CTG}$, $\overline{CTC}$ Setup Time and Hold Time

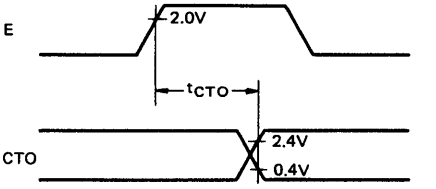


Figure 8 CTO Delay Time

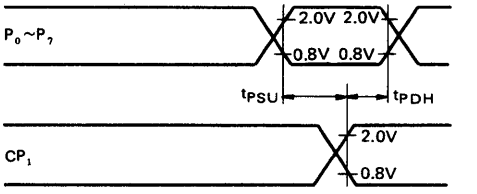


Figure 9 Peripheral Port Latch Setup and Hold Time

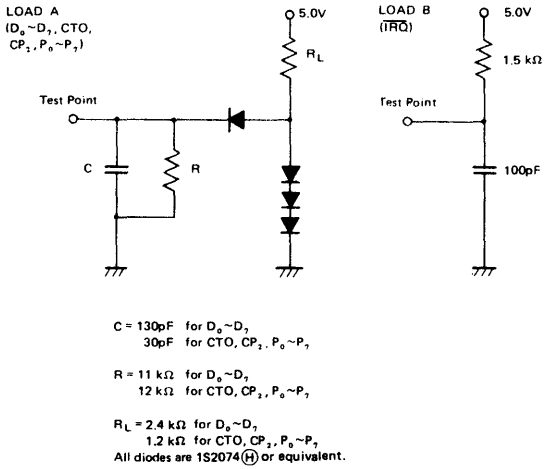


Figure 10 Bus Timing Test Loads

■ GENERAL DESCRIPTION

The HD6846 combination chip may be partitioned into three functional operating sections: programmed storage, timer-counter functions, and a parallel I/O port.

• Programmed Storage

The mask-programmable ROM section is similar to other ROM products of the HMCS6800 family. The ROM is organized in a 2048 by 8-bit array to provide read only storage for a minimum microcomputer system. Two mask-programmable chip selects are available for user definition.

Address inputs $A_0 \sim A_{10}$ allow any of the 2048 bytes of ROM to be uniquely addressed. Bidirectional data lines ($D_0 \sim D_7$) allow the transfer of data between the MPU and the HD 6846.

• Timer-Counter Functions

Under software control this 16-bit binary counter may be programmed to count events, measure frequencies, time intervals, or similar tasks. Internal registers associated with the I/O functions may be selected with A_0, A_1 and A_2. It may also be used for square wave generation, single pulses of controlled duration, and gated signals. Interrupts may be generated from a number of conditions selectable by software programming.

The timer/counter control register allows control of the interrupt enable, output enable, selection of an internal or external clock source, a ÷ 8 prescaler, and operating mode. Input pin $\overline{CTC}$ (counter-timer clock) will accept an asynchronous clock pulse to decrement the internal register for the counter-timer. If the divide-by-8 prescaler is used, the maximum clock rate can be four times the master clock frequency with an absolute maximum of 4 MHz. Gate input ($\overline{CTG}$) accepts an asynchronous TTL-compatible signal which may be used as a trigger or gating function to the counter-timer. A counter-timer output (CTO) is also available and is under software control being dependent on the timer control register, the gate input, and the clock source.

• Parallel I/O Port

The parallel bidirectional I/O port has functional operational characteristics similar to the B port on the HD6821 PIA. This includes 8 bidirectional data lines and two handshake control signals. The control and operation of these lines are completely software programmable.

The interrupt input (CP_1) will set the interrupt flag CSR1 of the composite status register. The peripheral control (CP_2) may be programmed to act as an interrupt input (set CSR2) or as a peripheral control output.

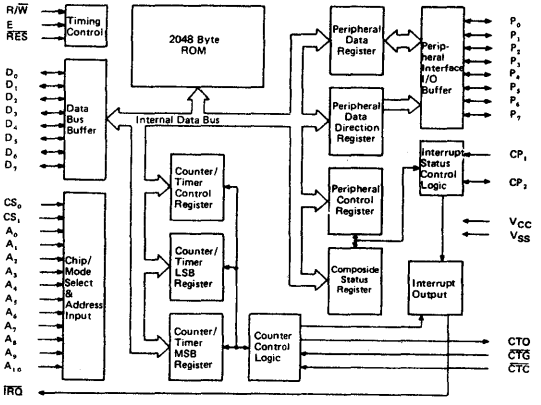


Figure 11 Combination ROM I/O Timer (COMBO)

Basic Block Diagram

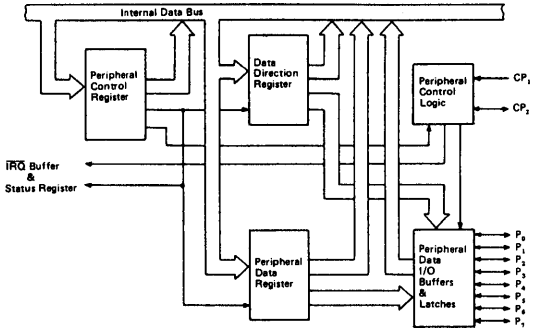


Figure 12 Parallel I/O Port Block Diagram

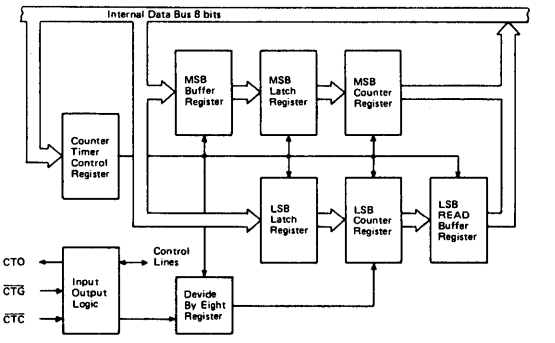


Figure 13 Timer/Counter Block Diagram

■ SIGNAL DESCRIPTION

• Bus Interface

The HD6846 interfaces to the HMCS6800 Bus via an eight bit bidirectional data bus, two Chip Select lines, a Read/Write line, and eleven address lines. These signals, in conjunction with the HMCS6800 VMA output, permit the MPU to control the HD6846.

• Bidirectional Data Bus ($D_0 \sim D_7$)

The bidirectional data lines ($D_0 \sim D_7$) allow the transfer of data between the MPU and the HD6846. The data bus output drivers are three-state devices which remain in the high-impedance (Off) state except when the MPU performs an HD6846 register or ROM read ($R/\overline{W}$ = 1 and I/O Registers or ROM selected).

• Chip Select (CS_0, CS_1)

The CS_0 and CS_1 inputs are used to select the ROM or I/O timer of the HD6846. They are mask programmed to be active "High" or active "Low" as chosen by the user.

• Address Inputs ($A_0 \sim A_{10}$)

The Address Inputs allow any of the 2048 bytes of ROM to be uniquely selected when the circuit is operating in the ROM mode. In the I/O-Timer mode, address inputs A_0, A_1, and A_2 select the proper I/O Register, while A_3 through A_{10} (together with CS_0 and CS_1) can be used as additional qualifiers in the I/O Select circuitry. (See the section on I/O-Timer Select for additional details.)

• Reset ($\overline{RES}$)

The active "Low" state of the $\overline{RES}$ input is used to initialize all register bits in the I/O section of the device to their proper values. (See the section on Initialization for Reset conditions for timer and peripheral registers.)

• Enable (E)

This signal synchronizes data transfer between the MPU and the HD6846. It also performs an equivalent synchronization function on the external clock, reset, and gate inputs of the HD6846 Timer section.

• Read/Write ($R/\overline{W}$)

This signal is generated by the MPU and is used to control the direction of data transfer on the bidirectional data pins. A "Low" level on the $R/\overline{W}$ input enables the HD6846 input buffers and data is transferred to the circuit during the E pulse when the part has been selected. A "High" level on the $R/\overline{W}$ input enables the output buffers and data is transferred to the MPU during E when the part is selected.

• Interrupt Request ($\overline{IRQ}$)

The active "Low" $\overline{IRQ}$ output acts to interrupt the MPU through logic included on the HD6846. This output utilizes an open drain configuration and permits other interrupt request outputs from other circuits to be connected in a wire-OR configuration.

• Peripheral Data ($P_0 \sim P_7$)

The peripheral data lines can be individually programmed as either inputs or outputs via the Data Direction Register. When programmed as outputs, these lines will drive two standard TTL

loads (3.2 mA). They are also capable of sourcing up to 1.0 mA at 1.5 Volts (Logic "1" output.)

When programmed as inputs, the output drivers associated with these lines enter a three-state (high impedance) mode. Since there is no internal pull-up for these lines, they represent a maximum 10μA load to the circuitry driving them -- regardless of logic state.

A logic zero at the $\overline{RES}$ input forces the peripheral data lines to the input configuration by clearing the Data Direction Register. This allows the system designer to preclude the possibility of having a peripheral data output connected to an external driver output during power-up sequence.

- **Interrupt Input (CP_1)**

Peripheral input line CP_1 is an input-only that sets the Interrupt Flags of the Composite Status register. The active transition for this signal is programmed by the peripheral control register for the parallel port. CP_1 may also act as a strobe for the peripheral data register when it is used as an input latch. Details for programming CP_1 are in the section on the parallel peripheral port.

(Note)

Unexpected noise may occur on the peripheral data line when the peripheral data register is loaded with "1". This erroneous noise may occur only when peripheral data line is specified as output and the peripheral data register has already been loaded with "1". Note that peripheral data line doesn't keep "High" level continuously in the case write peripheral data register operation is executed.

- **Peripheral Control (CP_2)**

Peripheral Control line CP_2 may be programmed to act as an Interrupt input or Peripheral Control output. As an input, this line has high impedance and is compatible with standard TTL voltage levels. As an output, it is also TTL compatible and may be used as a source of 1 mA at 1.5 V to directly drive the base of a Darlington transistor switch. This line is programmed by the Peripheral Control Register.

- **Counter Timer Output (CTO)**

The Counter Timer Output is software programmable by selected bits in the timer/counter control register. The mode of operation is dependent on the Timer control register, the gate input, and the clock source. The output is TTL compatible.

- **External Clock Input ($\overline{CTC}$)**

Input pin $\overline{CTC}$ will accept asynchronous TTL voltage level signals to be used as a clock to decrement the Timer. The "High" and "Low" levels of the external clock must be stable for at least one system clock period plus the sum of the setup and hold times for the inputs. The asynchronous clock rate can vary from dc to the limit imposed by System E, setup, and hold times.

The external clock input is clocked in by Enable pulses. Three Enable periods are used to synchronize and process the external clock. The fourth Enable pulse decrements the internal counter. This does not affect the input frequency; it merely creates a delay between a clock input transition and internal recognition of that transition by the HD6846. All references to $\overline{CTC}$ inputs in this document relate to internal recognition of the input transition. Note that a clock transition which does not meet setup and hold time specifications may require an additional Enable pulse for recognition.

When observing recurring events, a lack of synchronization will result in either "System jitter" or "Input jitter" being observed on the output of the HD6846 when using an asynchronous clock and gate input signal. "System jitter" is the result of the input signals being out of synchronization with Enable, permitting signals with marginal set-up and hold time to be recognized by either the bit time nearest the input transition or subsequent bit time. "Input jitter" can be as great at the time between the negative going transitions of the input signal plus the system jitter if the first transition is recognized during one system cycle, and not recognized the next cycle or vice-versa.

- **Gate Inputs ($\overline{CTG}$)**

The input pin $\overline{CTG}$ accepts an asynchronous TTL-compatible signal which is used as a trigger or a clock gating function to the Timer. The gating input is clocked into the HD6846 by the Enable signal in the same manner as the previously discussed clock inputs. That is, a $\overline{CTG}$ transition is recognized on the fourth Enable pulse (provided setup and hold time requirements are met), and the "High" or "Low" levels of the $\overline{CTG}$ input must be stable for at least one system clock period plus the sum of setup and hold times. All references to $\overline{CTG}$ transition in this document relate to internal recognition of the input transition.

The $\overline{CTG}$ input of the timer directly affects the internal 16-bit counter. The operation of $\overline{CTG}$ is therefore independent of the ÷ 8 prescaler selection.

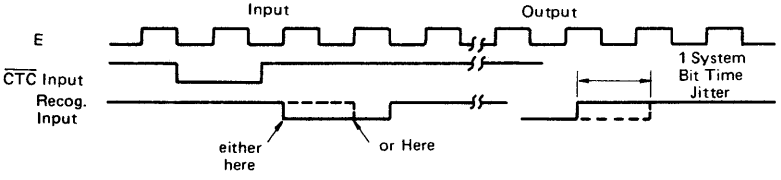
■ FUNCTIONAL SELECT CIRCUITRY

- **I/O-Timer Select Circuitry**

CS_0 and CS_1 are user programmable. Any of the four binary combinations of CS_0 and CS_1 can be used to select the ROM. Likewise, any other combination can be used to select the I/O-Timer. In addition, several address lines are used as qualifiers for the I/O-Timer. Specifically, $A_3 = A_4 = A_5$ = logical "0". A_6 can be programmed to a "1", "0", or don't care. $A_7 = A_8 = A_9 = A_{10}$ = don't care or one line only may be programmed to a logical "1". Figure 14 outlines in diagrammatic form the available chip select options.

- **Internal Addressing**

Seven I/O Register locations within the HD6846 are accessible to the MPU data bus. Selection of these registers is controlled by A_0, A_1, and A_2 (as shown in Table 1) provided the I/O timer is selected. The combination status register is Read-only; all other Registers are Read and Write.



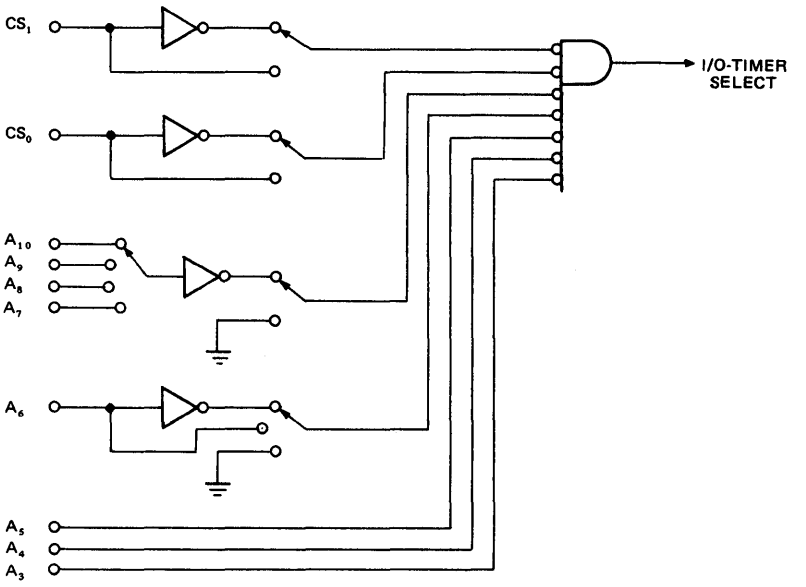


Figure 14 I/O-Timer Select Circuitry

Table 1 Internal Register Addresses

REGISTER SELECTED	A_2	A_1	A_0
Combination Status Register	0	0	0
Peripheral Control Register	0	0	1
Data Direction Register	0	1	0
Peripheral Data Register	0	1	1
Combination Status Register	1	0	0
Timer Control Register	1	0	1
Timer MSB Register	1	1	0
Timer LSB Register	1	1	1
ROM Address	×	×	×

Initialization

When the $\overline{RES}$ input has accepted a "Low" signal, all registers are initialized to the reset state. The data direction and peripheral data registers are cleared. The Peripheral Control Register is cleared except for bit 7 (the $\overline{RES}$ bit). This forces the parallel port to the input mode with Interrupts disabled. To remove the $\overline{RES}$ condition from the parallel port, a "0" must be written into the Peripheral Control Register bit 7 (PCR7).

The counter latches are preset to their maximal count, the Timer control register bits are reset to zero except for Bit 0 (TCR0 is set), the counter output is cleared, and the counter clock disabled. This state forces the timer counter to remain in an inactive state. The combination status register is cleared of all interrupt flags. During timer initialization, the reset bit (CCR0) must be cleared.

ROM

The Mask Programmable ROM section is similar in operation to other ROM products of the HMCS6800 Microprocessor family. The ROM is organized as 2048 words of 8-bits to provide read-only storage for a minimum microcomputer system. The ROM is active when selected by the unique combination of the chip select inputs.

ROM Select

The active levels of CS_0 and CS_1 for ROM and I/O select are a user programmable option. Either CS_0 or CS_1 may be programmed active "High" or active "Low", but different codes must be used for ROM or I/O select. CS_0 and CS_1 are mask programmed simultaneously with the ROM pattern. The ROM Select Circuitry is shown in Figure 15.

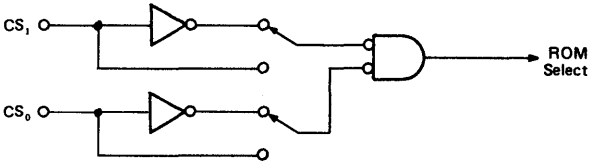


Figure 15 ROM Select Circuitry

■ TIMER OPERATION

The Timer may be programmed to operate in modes which fit a wide variety of applications. The device is fully bus compatible with the HMCS6800 system, and is accessed by Load and Store operations from the MPU.

In a typical application, the timer will be loaded by storing two bytes of data into the counter latch. This data is then transferred into the counter during a Counter Initialization cycle. The counter decrements on each subsequent clock cycle (which may be Enable or an external clock) until one of several predetermined conditions causes it to halt or recycle. Thus the timer is programmable, cyclic in nature, controllable by external inputs or MPU program, and accessible to the MPU at any time.

- **Counter Latch Initialization**

The Timer consists of a 16-bit addressable counter and two 8-bit addressable latches. The function of the latches is to store a binary equivalent of the desired count value minus one. Counter initialization results in the transfer of the latch contents of the counter. It should be noted that data transfer to the counters is always accomplished via the latches. Thus, the counter latches may be accurately described as a 16-bit "counter initialization data" storage register.

In some modes of operation, the initialization of the latches will cause simultaneous counter initialization (i.e. immediate transfer of the new latch data into the counters). It is, therefore, necessary to insure that all-16-bit of the latches are updated simultaneously. Since the HD6846 data bus is 8-bit wide, a temporary register (MSB Buffer Register) is provided for in the Most Significant Byte of the desired latch data. This is a "write-only" register selected via address lines A_0, A_1, and A_2. Data is transferred directly from the data bus to the MSB Buffer when the chip is selected, R/$\overline{W}$ is "Low", and the timer MSB register is selected (A_0 = "0", A_1 = A_2 = "1").

The lower 8-bit of the counter latch can also be referred to as a "write-only" register. Data Bus information will be transferred directly to the LSB of a counter latch when the chip is selected, R/$\overline{W}$ is "Low" and the Timer LSB Register is selected (A_0 = A_1 = A_2 = "1"). Data from the MSB Buffer will automatically be transferred into the Most Significant Byte of the counter latches simultaneously with the transfer of the Data Bus information to the Least Significant Byte of the Counter Latch. For brevity, the conditions for this operation will be referred to henceforth as a "Write Timer Latches Command."

The HD6846 has been designed to allow transfer of two bytes of data into the counter latches from any source, provided the MSB is transferred first. In many applications, the source of data will be an HMCS6800 MPU. It should therefore be noted that the 16-bit store operations of the HMCS6800 family microprocessors (STS and STX) transfer data in the order required by the HD6846. A Store Index Register instruction, for example, results in the MSB of the index register being transferred to the selected address, then the LSB of the index register being written into the next higher location. Thus, either the index register or stack pointer may be transferred directly into a selected counter latch with a single instruction.

A logic zero at the $\overline{RES}$ input also initializes the counter latches. All latches will assume maximum count (65,535) values. It is important to note that an internal reset (Bit zero of the Timer/Control Register Set) has no effect on the counter latches.

- **Counter Initialization**

Counter Initialization is defined as the transfer of data from the latches to the counter with attendant clearing of the Individual Interrupt Flag associated with the counter. Counter Initialization always occurs when a reset condition (external $\overline{RES}$ = "0" or TCR0 = "1") is recognized. It can also occur (dependent on The Timer Mode) with a Write Timer Latches command or recognition of a negative transition of the $\overline{CTG}$ input.

Counter recycling or reinitialization occurs when a clock input is recognized after the counter has reached an all-zero state. In this case, data is transferred from the Latches to the Counter, but the Interrupt Flag is unaffected.

- **Timer Control Register**

The Timer Control register (see Table 2) in the HD6846 is used to modify timer operation to suit a variety of applications. The Timer Control Register has a unique address space (A_0 = "1, A_1 = "0", A_2 = "1") and therefore may be written into at any time. The least significant bit of the Control Register is used as an Internal Reset bit. When this bit is a logic zero, all timers are allowed to operate in the modes prescribed by the remaining bits of the timer control register.

Writing "1" into Timer Control Register Bit 0 (TCR0) causes the counter to be preset with the conents of the counter latches, all counter clocks are disabled, and the timer output and interrupt flag (Status Register) are reset. The Counter Latch and Timer/Control Register are undisturbed by an Internal Reset and may be written into regardless of the state of TCR0.

Timer Control Register Bit 1 (TCR1) is used to select the clock source. When TCR1 = "0", the external clock input $\overline{CTC}$ is selected, and when TCR1 = "1", the timer uses Enable.

Table 2 Format for Timer/Counter Control Register

CONTROL REGISTER BIT	STATE	BIT DEFINITION	STATE DEFINITION
TCR0	0	Internal Reset	Timer Enabled
	1		Timer in Preset State
TCR1	0	Clock Source	Timer uses External Clock ($\overline{CTC}$)
	1		Timer uses ϕ2 System Clock
TCR2	0	÷8 Prescaler Enabler	Clock is not Prescaled
	1		Clock is prescaled by ÷8 Counter
TCR3 TCR4 TCR5	× × ×	Operating Mode Selection	See Table 3
TCR6	0	Timer Interrupt Enable	$\overline{IRQ}$ Masked from Timer
	1		$\overline{IRQ}$ Enabled from Timer
TCR7	0	Timer Output Enable	Counter Output (CTO) Set "LOW"
	1		Counter Output Enabled

Table 3 Counter/Timer Operation Modes

Mode	TCR3	TCR4	TCR5	Counter Initialization	Counter Enable	Counter Clock "CC"	Interrupt Flag	
							Set	Clear
Continuous Mode	0	0	0	$\overline{G}\downarrow$ +W+R	($\overline{G}$=Low) · $\overline{R}$	CE · C	TO	RS–RT or CI
	0	1	0	$\overline{G}\downarrow$ +R	($\overline{G}$=Low) · $\overline{R}$	CE · C	TO	RS–RT or CI
Cascaded Single Shot Mode	0	0	1	$\overline{G}\downarrow$ +W+R	$\overline{R}$	CE · C	TO	RS–RT or CI
Normal Single Shot Mode	0	1	1	$\overline{G}\downarrow$ +R	$\overline{R}$	CE · C	TO	RS–RT or CI
Frequency Comparison Mode	1	0	0	$(\overline{CE}+TOF\cdot CE)\cdot\overline{G}\downarrow\cdot\overline{I}$ +R	CE set=$\overline{G}\downarrow\cdot\overline{W}\cdot\overline{R}\cdot\overline{T}$ CE reset=W+R+I	CE · C	$\overline{G}\downarrow$ before TO	RS–RT or CI or W
	1	0	1	$\overline{G}\downarrow\cdot\overline{T}$+R	CE set=$\overline{G}\downarrow\cdot\overline{W}\cdot\overline{R}\cdot\overline{T}$ CE reset=W+R+I	CE · C	$\overline{G}\downarrow$ before TO	RS–RT or CI or W
Pulse Width Comparison Mode	1	1	0	$\overline{G}\downarrow\cdot\overline{T}$+R	CE set=$\overline{G}\downarrow\cdot\overline{W}\cdot\overline{R}\cdot\overline{T}\cdot G$ CE reset=W+R+I+($\overline{G}$=High)	CE · C	$\overline{G}\uparrow$ before TO	RS–RT or CI or W
	1	1	1	$\overline{G}\downarrow\cdot\overline{T}$+R	CE set=$\overline{G}\downarrow\cdot\overline{W}\cdot\overline{R}\cdot\overline{T}\cdot G$ CE reset=W+R+I+($\overline{G}$=High)	CE · C	$\overline{G}\uparrow$ before TO	RS–RT or CI or W

R = External $\overline{RES}$ or Internal Reset TCR0
W = Write Timer Latch
I = Interrupt Flag
$\overline{G}$ = $\overline{CTG}$
C = Clock selected in the internal register
$\overline{G}\downarrow$ = Negative transition of $\overline{CTG}$ signal
$\overline{G}\uparrow$ = Positive transistion of $\overline{CTG}$ signal
RS–RT = Read Operation of Timer Counter after the read of Status Register (Normal operation to clear the interrupt)
CI = Counter Initialization (Internal Signal)
TOF = Time Out Flag (Set by $\overline{CI}$·TO, Reset by CI)
TO = Counter Time Out

Timer Control Register Bit 2 (TCR2) enables the ÷ 8 prescaler (TCR2 = "1"). In this mode, the clock frequency is divided by eight before being applied to the counter. When TCR2 = "0" Enable is applied directly to the counter.

TCR3, 4, 5 select the Timer Operating Mode, and are discussed in the next section.

Timer Control Register Bit 6 (TCR6) is used to mask or enable the Timer Interrupt Request. When TCR6 = "0", the Interrupt Flag is masked from the timer. When TCR6 = "1", the Interrupt Flag is enabled into Bit 7 of the Composite Status Register (Composite IRQ Bit), which appears on the $\overline{IRQ}$ output pin.

Timer Control Register Bit Seven (TCR7) has a special function when the timer is in the Cascaded Single Shot mode. (This function is explained in detail in the section describing the mode.) In all other modes, TCR7 merely acts as an output enable bit. If TCR7 = "0", the Counter Timer Output (CTO) is forced "Low". Writing a logic one into TCR7 enables CTO.

- **Timer Operating Modes**

The HD6846 has been designed to operate effectively in a wide variety of applications. This is accomplished by using three bits of the control register (TCR3, TCR4, and TCR5) to define different operating modes of the Timer, outlined in Table 3.

- **Continuous Operating Mode (TCR3 = 0, TCR5 = 0)**

The timer may be programmed to operate in a continuous counting mode by writing zeros into bits 3 and 5 of the timer control register. Assuming that the timer output is enabled (TCR7 = "1"), a square wave will be generated at the Timer Output CTO (See Table 4).

Table 4 Continuous Operating Modes

CONTINUOUS MODE
(TCR3=0, TCR7=1, TCR5=0)

CONTROL REGISTER		INITIALIZATION/OUTPUT WAVEFORMS
TCR4	Counter	CTO
0	Initialization $\overline{G}\downarrow$ +W+R	←(N+1) (T)→ ←(N+1) (T)→ ←(N+1) (T)→
1	$\overline{G}\downarrow$ +R	V_{OH} V_{OL}
		to TO* TO* TO*

$\overline{G}\downarrow$ = Negative Transition $\overline{CTG}$ Input.
W = Write Timer Latches Command.
R = Timer Reset (TCR0=1 or External $\overline{RES}$=0)
N = 16 Bit Number in Counter Latch.
T = Period of Clock Input to Counter.
to = Counter Initialization Cycle.
TO = Counter Time Out (All Zero Condition.)
* Point at which an interrupt may occur.

Either a Timer Reset (TCR0 = "1" or External $\overline{RES}$ = "0") condition or internal recognition of a negative transition of the $\overline{CTG}$ input results in Counter Initialization. A Write Timer Latches command can be selected as a Counter Initialization signal by clearing TCR4.

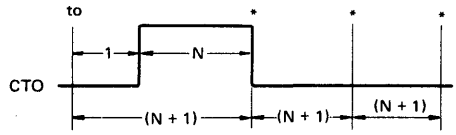
The discussion of the Continuous Mode has assumed the application requires an output signal. It should be noted the Timer operates in the same manner with the output disabled (TCR7 = "0"). A Read Timer Counter command is valid regardless of the state of TCR7.

- **Normal Single-Shot Timer Mode (TCR3 = 0, TCR4 = 1, TCR5 = 1)**

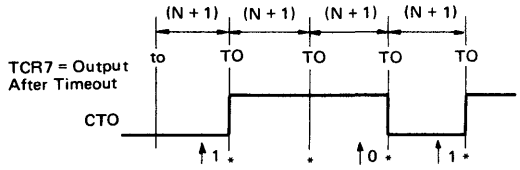
This mode is identical to the Continuous Mode with two exceptions. The first of these is obvious from the name – the output returns to a "Low" level after the initial Time Out and remains "Low" until another Counter Initialization cycle occurs. The output waveform (CTO) is shown in Figure 16.

As indicated in Figure 16, the internal counting mechanism remains cyclical in the Single-Shot Mode. Each Time Out of the counter results in the setting of an Individual Interrupt Flag and re-initialization of the counter.

The second major difference between the Single-Shot and Continuous modes is that the internal counter enable is not dependent on the $\overline{CTG}$ input level remaining in the "Low" state for the Single-Shot mode. Aside from these differences, the two modes are identical.



(A) NORMAL SINGLE-SHOT MODE OUTPUT WAVEFORM



(B) CASCADED SINGLE-SHOT MODE OUTPUT WAVEFORM

1 = Write a "1" into TCR7
0 = Write a "0" into TCR7

*Point at which an interrupt may occur.

(NOTE) All time intervals shown above assume the Gate ($\overline{CTG}$) and Clock ($\overline{CTC}$) signals are synchronized to Enable with the specified setup and hold time requirements.

Figure 16 Single-Shot Modes

- **Cascaded Single-shot Mode (TCR3=0, TCR4=0, TCR5=1)**

This mode is identical to the single-shot mode with two exceptions. First, the output waveform does not return to a "Low" level and remain "Low" after timeout. Instead, the output level remains at its initialized level until it is re-programmed and changed by timeout. The output level may be changed at any timeout or may have any number of timeouts between changes.

The second difference is the method used to change the output level. Timer Control Register Bit 7 (TCR7) has a special function in this mode. The timer output (CTO) is equal to TCR7 clocked by timeout. At every timeout, the content of TCR7 is clocked to and held at the CTO output. Thus, output pulses of length greater than one timer cycle can be generated by cascading timer cycles and counting timeouts with a software program (See Figure 16).

An interrupt is generated at each timeout. To cascade timer cycles, the MPU would need an interrupt routine to: 1) count each timeout and determine when to change TCR7: 2) write into TCR7 the state corresponding to the next desired state of the output waveform (only necessary during the last timer cycle before the output is to change state): and 3) clear the interrupt flag by reading the combination status register followed by Read Timer MSB. It is also possible, if desired, to change the length of the timer cycle by reinitializing the timer latches. This allows more flexibility for obtaining desired times.

- **Time Interval Modes (TCR3 = 1)**

The Time Interval Modes are provided for applications requiring more flexibility of interrupt generation and Counter Initialization. The Interrupt Flag is set in these modes as a function of both Counter Time Out and transitions of the $\overline{CTG}$ input. Counter Initialization is also affected by Interrupt Flag status. The output signal is not defined in any of these modes. Other features of the Time Interval Modes are Outlined in Table 5.

- **Frequency Comparison Mode (TCR3 = 1, TCR4 = 0)**

The timer within the HD6846 may be programmed to compare the period of a pulse (giving the frequency after calculations) at the $\overline{CTG}$ input with the time period required for Counter Time Out. A negative transition of the $\overline{CTG}$ input enables the counter and starts a Counter Initialization cycle – provided that other conditions as noted in Table 3 are satisfied. The counter decrements on each clock signal recognized during or after Counter Initialization until an Interrupt is generated, a

Table 5 Time Interval Modes

TCR3 = 1			
TCR4	TCR5	APPLICATION	CONDITION FOR SETTING INDIVIDUAL INTERRUPT FLAG
0	0	Frequency Comparison	Interrupt Generated if $\overline{CTG}$ Input Period (1/F) is Less Than Counter Time Out (TO).
0	1	Frequency Comparison	Interrupt Generated if $\overline{CTG}$ Input Period (1/F) is Greater Than Counter Time Out (TO).
1	0	Pulse Width Comparison	Interrupt Generated if $\overline{CTG}$ Input "Down Time" is Less Than Counter Time Out (TO).
1	1	Pulse Width Comparison	Interrupt Generated if $\overline{CTG}$ Input "Down Time" is Greater Than Counter Time Out (TO).

Write Timer Latches command is issued, or a Timer Reset condition occurs. It can be seen from Table 3 that an interrupt condition will be generated if TCR5 = "0" and the period of the pulse (single pulse or measured separately repetative pulses) at the $\overline{CTG}$ input is less than the Counter Time Out period. If TCR5 = "1", an interrupt is generated if the reverse is true.

Assume now with TCR5 = "1" that a Counter Initialization has occurred and that the $\overline{CTG}$ input has returned "Low" prior to Counter Time Out. Since there is no Individual Interrupt Flag generated, this automatically starts a new Counter Initialization Cycle. The process will continue with frequency comparison being performed on each $\overline{CTG}$ input cycle until the mode is changed, or a cycle is determined to be above the predetermined limit.

- **Pulse Width Comparison Mode (TCR3 = 1, TCR4 = 1)**

This mode is similar to the Frequency Comparison Mode except for the limiting factor being a positive, rather than negative, transition of the $\overline{CTG}$ input. With TCR5 = "0", an Individual Interrupt Flag will be generated if the zero level pulse applied to the $\overline{CTG}$ input is less than the time period required for Counter Time Out. With TCR5 = "1", the interrupt is generated when the reverse condition is true.

As can be seen in Table 3, a positive transition of the $\overline{CTG}$ input disables the counter. With TCR5 = "0", it is therefore possible to directly obtain the width of any pulse causing an interrupt.

- **Composite Status Register**

The Composite Status Register (CSR) is a read-only register which is shared by the Timer and the Peripheral Data Port of the HD6846. Three individual interrupt flags in the register are set directly via the appropriate conditions in the timer or peripheral port. The composite interrupt flag – and the $\overline{IRQ}$ Output – respond to these individual interrupts only if corresponding enable bits are set in the appropriate Control Registers. (See Figure 17.) The sequence of assertion is not detected. Setting TCR6 while CSR0 is "High" will cause CSR7 to be set, for example.

The Composite Interrupt Flag (CSR7) is clear only if all enabled Individual Interrupt Flags are clear. The conditions for clearing CSR1 and CSR2 are detailed in a later section. The Timer Interrupt Flag (CSR0) is cleared under the following conditions:

1) Timer Reset – Internal Reset Bit (TCR0) = "1" or External $\overline{RES}$ = "0".
2) Any Counter Initialization condition.
3) A Write Timer Latches command if Time Interval modes (TCR3 = "1") are being used.
4) A Read Timer Counter command, provided this is preceded by a Read Composite Status Register while CSR0 is set. This latter condition prevents missing an Interrupt Request generated after reading the Status Register and prior to reading the counter.

The remaining bits of the Composite Status Register (CSR3~CSR6) are unused. They default to a logic zero when read.

■ I/O OPERATION

- **Parallel Peripheral Port**

The peripheral port of the HD6846 contains 8 Peripheral Data lines (P_0~P_7), two Peripheral Control lines (CP_1 and CP_2), a Data Direction Register, a Peripheral Data Register, and a Peripheral Control Register. The port also directly affects two bits (CSR1 and CSR2) of the Composite Status Register.

The Peripheral Port is similar to the "B" side of a PIA (HD6821) with the following exceptions:

1) All registers are directly accessible in the HD6846 Data Direction and Peripheral Data in the HD6821 are located at the same address with Bit Two of the Control Register used for register selection.
2) Peripheral Control Register Bit Two (PCR2) of the HD6846 is used to select an optional input latch function. This option is not available with HD6821 PIA's.
3) Interrupt Flags are located in the HD6846 composite status register rather than Bits 6 and 7 of the Control Register as used in the HD6821.
4) Interrupt Flags are cleared in the HD6821 by reading data from the Peripheral Data Register. HD6846 Interrupt Flags are cleared by either reading or writing to the Peripheral Data Register – provided that this sequence is followed a) Flag Set, b) Read Composite Status Register, c) Read/Write Peripheral Data Register is followed.

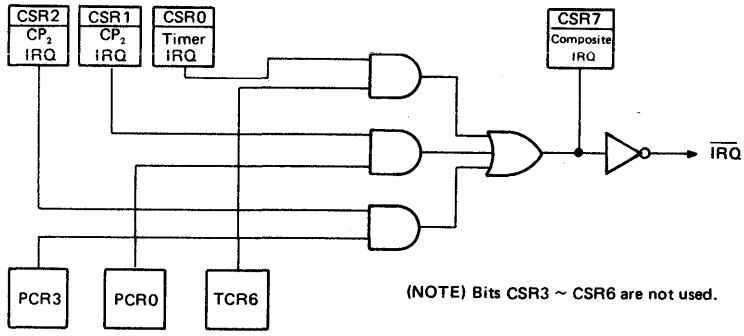


Figure 17 Composite Status Register & Associated Logic

5) Bit 6 of the HD6846 Peripheral Control Register is not used. Bit 7 (PCR7) is an Internal Reset Bit not available on the HD6821.
6) The Peripheral Data lines (and CP_2) of the HD6846 feature internal current limiting which allows them to directly drive the base of Darlington NPN transistors.

- **Data Direction Register**

The MPU can write directly to this eight-bit register to configure the Peripheral Data lines as either inputs or outputs. A particular bit within the register (DDRn) is used to control the corresponding Peripheral Data line (Pn). With DDRn = "0", Pn becomes an input; if DDRn = "1", Pn is an output. As an example, writing Hex $0F into the Data Direction Register results in P_0 thru P_3 becoming outputs and P_4 thru P_7 being inputs. Hex $55 in the Data Direction Register results in alternate outputs and inputs at the parallel port.

- **Peripheral Data Register**

This eight-bit register is used for transferring data between the peripheral data port and the MPU. Any bit corresponding to an output line will be used to drive the output buffer associated with that line. Data in these output bits is normally provided by an MPU Write function. (Input bits – those associated with input lines – are unchanged by a Write Command.) Any input bit will reflect the state of the associated input line if the input latch function is deselected. If the Control Register is programmed to provide input latching, the input bit will retain the state at the time CP_1 was activated until the Peripheral Data Register is read by the MPU.

- **Peripheral Control Register**

This eight-bit register is used to control the reset function as well as for selection of optional functions of the two peripheral control lines (CP_1 and CP_2). The Peripheral Control Register functions are outlined in Table 6.

- **Peripheral Port Reset (PCR7)**

Bit 7 of the Peripheral Control Register (PCR7) may be used to initialize the peripheral section of the HD6846. When this bit is set "High", the peripheral data register, the peripheral data direction register, and the interrupt flags associated with the peripheral port (CSR1 & CSR2) are all cleared. Other bits in the peripheral control register are not affected by PCR7.

PCR7 is set by either a logic zero at the External $\overline{RES}$ input or under program control by writing a "1" into the location. In any case, PCR7 may be cleared only by writing a zero into the location while $\overline{RES}$ is "High". The bit must be cleared to activate the port.

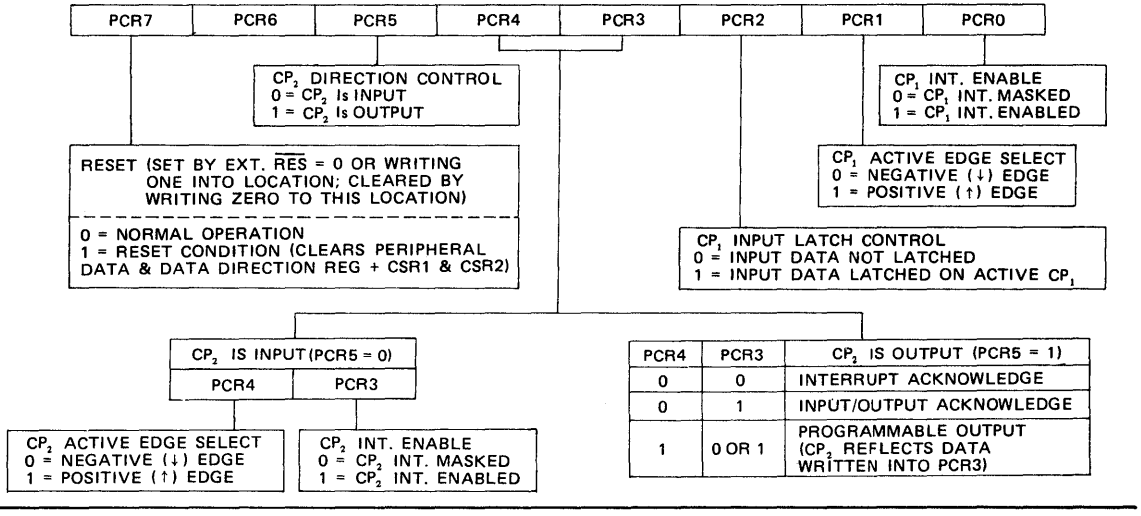
- **Control of CP_1 Peripheral Control Line**

CP_1 may be used an interrupt request to the HD6846, as a strobe to allow latching of input data, or both. In any case, the input can be programmed to be activated by either a positive or negative transition of the signal. Thes options are selected via Control Register Bits PCR0, RCR1 & PCR2.

Control Register Bit 0 (PCR0) is used to enable the interrupt transfer circuitry of the HD6846. Regardless of the state of PCR0, and active transition of CP_1 causes the Composite Status Register Bit One (CSR1) to be set. if PCR0 = "1", this interrupt will be reflected in the Composite Interrupt Flag (CSR7), and thus at the $\overline{IRQ}$ output. CSR1 is cleared by a Peripheral Port Reset condition or by either reading or writing to the peripheral data register after the Composite Status Register is read. The latter alternative is conditional – CSR1 must have been a logic one when the Composite Status Register was lart read. This precludes inadvertent clearing of interrupt flags generated between the time the Status Register is read and the manipulation of peripheral data.

Control Register Bit One (PCR1) is used to select the edge which activates CP_1. When PCR1 = "0", CP_1 is active on negative transitions ("High" to "Low"). "Low" to "High" transitions are sensed by CP_1 when PCR1 = "1"

Table 6 Peripheral Control Register Format (Expanded)



In addition to its use as an interrupt input, CP_1 can be used as a strobe to capture input data in an internal latch.This option is selected by writing a one into Peripheral Control Register Bit Two (PCR2). In operating, the data at the pins designated by the Data Direction Register as inputs will be captured by an active transition of CP_1. An MPU Read of the Peripheral Data Register will result in the captured data being transferred to the MPU – and it also releases the latch to allow capture of new data. Note that successive active transitions with no Read Peripheral Data Command between does not update the input latch. Also, it should be noted that use of the input latch function (which can be deselected by writing a zero into PCR2) has no effect on output data. It also does not affect Interrupt function of CP_1.

- **Control of CP_2 Peripheral Control Line**

CP_2 may be used as an input by writing a zero into PCR5. In this configuration, CP_2 becomes a dual of CP_1 in regard to generation of interrupts. An active transition (as selected by PCR4) causes Bit Two of the Composite Status Register to be set. PCR3 is then used to select whether the CP_2 transition is to cause CSR7 to be set – and thereby cause $\overline{IRQ}$ to go "Low". CP_2 has no effect on the input latch function of the HD6846.

Writing a one into PCR5 causes CP_2 to function as an output. PCR4 then determines whether CP_2 is to be used in a handshake or programmable output mode. With PCR4 = "1", CP_2 will merely reflect the data written into PCR3. Since this can readily be changed under program control, this mode allows CP_2 to be a programmable output line in much the same manner as those lines selected as outputs by the Data Direction Register.

The handshaking mode (PCR5 = "1", PCR4 = "0") allows CP_2 to perform one of two functions as selected by PCR3. With PCR3 = "1", CP_2 will go "Low" on the first Enable positive transition after a Read or Write to the Peripheral Data Register. This Input/Output Acknowledge signal is released (returns "High") on the next positive transition of the Enable signal.

In the Interrupt Acknowledge mode (PCR5 = "1", PCR4 = PCR3 = "0"), CP_2 is set when CSR1 is set by an active transition of CP_1. It is released (goes "Low") on the first positive transition of Enable after CSR1 has been cleared via an MPU Read or Write to the Peripheral Data Register. (Note that the previously described conditions for clearing CSR1 still apply.)

- **Restart Sequence**

A typical restart sequence for the HD6846 will include initialization of both the Peripheral Control & Data Direction Registers of the parallel port. It is necessary to set up the Peripheral Control Register first, since PCR7 = "0" is a condition for writing data into the Data Direction Register. (A logic zero at the external $\overline{RES}$ input automatically sets PCR7.)

- **Summary**

The HD6846 has several optional modes of operation which allow it to be used in a variety of applications. The following tables are provided for reference in selecting these modes.

Table 7 HD6846 Internal Register Addresses

R/$\overline{W}$	A_2	A_1	A_0	REGISTER SELECTED
R	0	0	0	Combination Status Register
R/$\overline{W}$	0	0	1	Peripheral Control Register
R/$\overline{W}$	0	1	0	Data Direction Register
R/$\overline{W}$	0	1	1	Peripheral Data Register
R	1	0	0	Combination Status Register
R/$\overline{W}$	1	0	1	Timer Control Register
R/$\overline{W}$	1	1	0	Timer MSB Register
R/$\overline{W}$	1	1	1	Timer LSB Register
R	x	x	x	ROM Address

Table 8 Composite Status Register

CSR7	CSR3~CSR6 NOT USED DEFAULT TO ZERO WHEN READ	CSR2	CSR1	CSR0

COMPOSITE INTERRUPT FLAG
0 = NO ENABLED INTERRUPT FLAG SET
1 = ONE OR MORE ENABLED INTERRUPT FLAGS SET*

INVERSE OF THIS BIT APPEARS AT $\overline{IRQ}$ OUTPUT

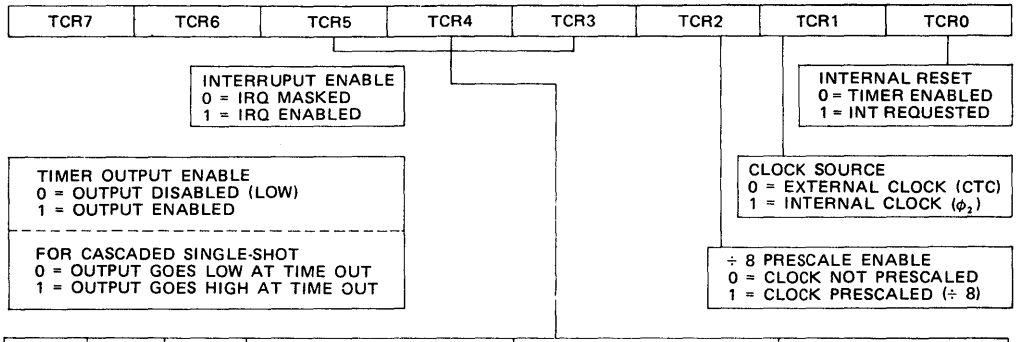
*STATUS OF THIS BIT CAN BE EXPRESSED AS:
CSR7 = CSR0·TCR6 + CSR1·PCR0 + CSR2·PCR3

CP_2 INTERRUPT FLAG
0 = NO INT REQ
1 = INT REQUESTED

TIMER INTERRUPT FLAG
0 = NO INT REQ.
1 = INT REQUESTED

CP_1 INTERRUPT
0 = NO INT REQ.
1 = INT REQUESTED

Table 9 Timer Control Register

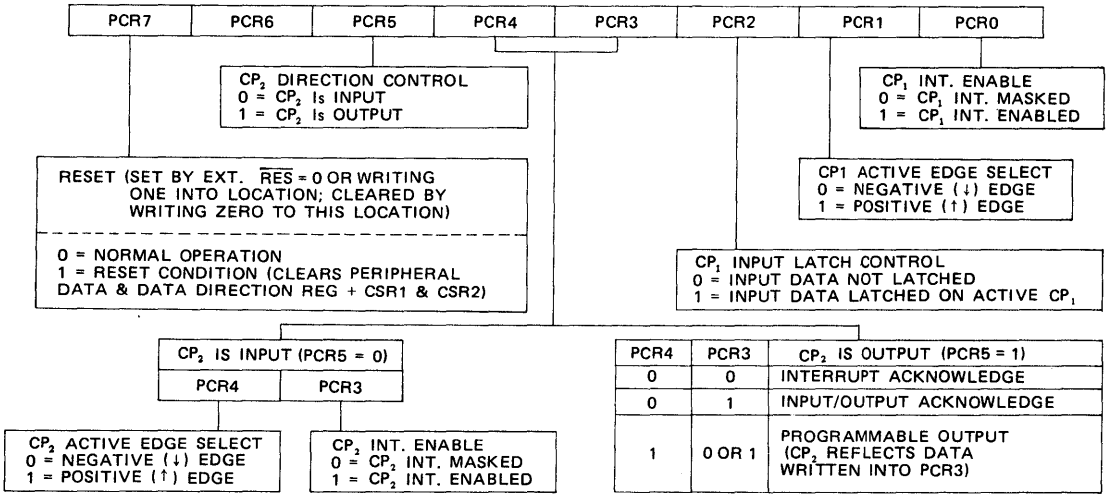


TCR3	TCR4	TCR5	TIME OPERATING MODE	COUNTER INITIALIZATION	INTERRUPT FLAG SET
0	0	0	CONTINUOUS	$\overline{CTG}\downarrow + W + R$	TO
0	0	1	CASCADED SINGLE SHOT	$\overline{CTG}\downarrow + W + R$	TO
0	1	0	CONTINUOUS	$\overline{CTG}\downarrow + R$	TO
0	1	1	NORMAL SINGLE SHOT	$\overline{CTG}\downarrow + R$	TO
1	0	0	FREQUENCY COMPARISON	$\overline{CTG}\downarrow \cdot \overline{T} \cdot (\overline{CE} + TOF \cdot CE) + R$	CTG ↓ BEFORE TO
1	0	1		$\overline{CTG}\downarrow \cdot \overline{T} + R$	TO BEFORE CTG
1	1	0	PULSE WIDTH COMPARISON	$\overline{CTG}\downarrow \cdot \overline{T} + R$	$\overline{CTG}\uparrow$ BEFORE TO
1	1	1		$\overline{CTG}\downarrow \cdot \overline{T} + R$	TO BEFORE $\overline{CTG}\uparrow$

R = RESET CONDITION
W = WRITE TIMER LATCHES
TO = COUNTER TIME OUT
CE = COUNTER ENABLE

$\overline{CTG}\downarrow$ = NEG TRANSISTION OF PIN 17
$\overline{CTG}\uparrow$ = POS TRANSITION OF PIN 17
$\overline{T}$ = INTERRUPT FLAG (CSR0) = 0

Table 10 Peripheral Control Register



PCR4	PCR3	CP2 IS OUTPUT (PCR5 = 1)
0	0	INTERRUPT ACKNOWLEDGE
0	1	INPUT/OUTPUT ACKNOWLEDGE
1	0 OR 1	PROGRAMMABLE OUTPUT (CP2 REFLECTS DATA WRITTEN INTO PCR3)

■ CUSTOM PROGRAMMING

By the programming of a single photomask for the HD6846, the customer may specify the content of the memory and the method of enabling the outputs.

Information on the general options of the HD6846 should be submitted on an Organizational Data form such as that shown in Figure 18 and Figure 19.

Information for custom memory content may be sent to HITACHI in one of two forms (shown in order of preference):

1) Paper tape output of the HMCS6800 Load Module Format or of the BNPF Format
2) Hexadecimal coding using IBM Punch Cards

ORGANIZATIONAL DATA
HD6846 COMBINATION ROM-I/O-TIMER

Customer:

Company _____

Part No. _____

Originator _____

Phone No. _____

Hitachi Use Only:

Quote:

Part No.:

Specif. No.:

Enable Options: (ROM ENABLE MUST DIFFER FROM I/O-TIMER)

	ROM SECTION 1	ROM SECTION 0	I/O-TIMER SECTION 1	I/O-TIMER SECTION 0
CS_0	☐	☐	☐	☐
CS_1	☐	☐	☐	☐

I/O-TIMER SELECT — A_6 (1 0 x): ☐ ☐ ☐

	CHECK ONE COLUMN ONLY				
A_{10}	x	1	x	x	x
A_9	x	x	1	x	x
A_8	x	x	x	1	x
A_7	x	x	x	x	1

1 ≥ 2.0V
0 ≤ 0.8V
x = NOT USED

Figure 18 Format for Programming General Options

DATE

COMPANY ENGINEER SECTION

CUSTOMERS P/N (if you need) ☐☐☐☐☐☐☐☐☐☐☐☐

TYPE NO. OF ROM

DATA FORMAT
1. HMCS6800 load module format 2. BNPF format

coding media
☐ 1. paper tape
☐ 2. IBM 80 column card

total bytes of data (decimal)

initial ROM address (decimal)

parity (for paper tape)
☐ 1. even ☐ 2. odd ☐ 3. none

total number of cards

for HITACHI reference only

ref. No.	
mask ROM No.	
processed data	
approved data	

designed _____

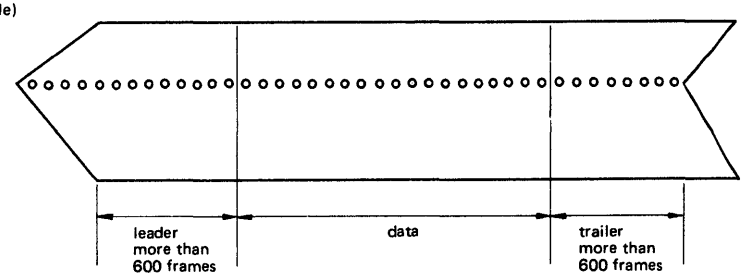
approved _____

Figure 19 Confirmation sheet of specification for HD6846 series ROM

■ PAPER TAPE

1) Any one inch width tape usually available in market can be used but tape in black color is recommended.
2) Both leader and trailer have more than 600 frames.

(Example)



3) One file data of each chip shall be contained in one reel of paper tape. One file data shall not be divided into more than two reels.
4) Parity
   Parity shall be indicated in "Confirmation sheet of specification". Parity forms are grouped;
   (1) With parity EVEN or ODD
   (2) Without parity
5) 8-bit ASCII code shall be used.

■ CARD

1) Use IBM 80 column card.
2) Use EBCDIC code.
3) Card format is as follows;

column	contents
1 to 71	Free format of data column
72	Blank
73 to 80	Sequential card number, not free format. Least significant digit of decimal sequential number is located in column 80. No alphabet letters. Any sequential number more than 1 can be used.

4) Total number of cards shall be written in "Confirmation sheet of specification".

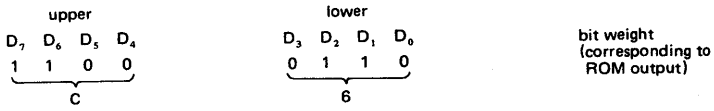
■ DATA FORMAT

• HMCS6800 LOAD MODULE FORMAT

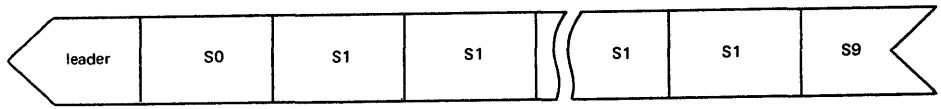
This is object format obtained from HMCS6800 assembler.

1) 8-bit code is divided into upper and lower 4 bits and transformed into hexadecimal number.

(Example) Binary number if 1100 0010 is transformed as follows.

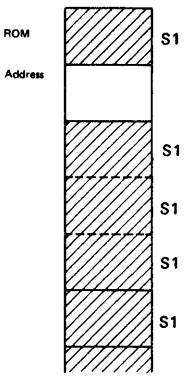


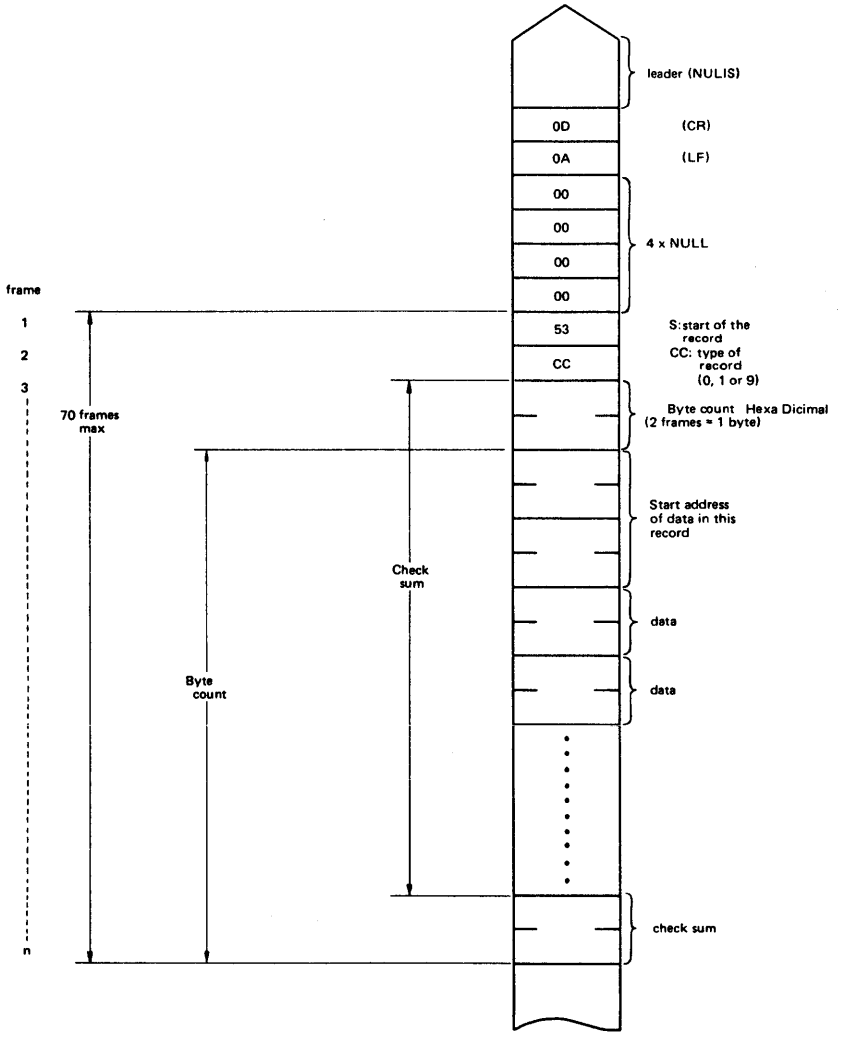
2) Load module structure of paper tape is shown as an example.



S0 is the header record, S1 is data record and S9 is end of file record. Each data record corresponds to each ROM data as shown below. Continuous memory address shall be devided into several records due to limitation of maximum frame number (70 frames = 35 bytes) in one record.

S0, S1 or S9 is distinguished by CC following start of the record S.





Check sum is complement of 1 for sum of each 8-bit.

3) Example of load module format

	frame	CC=30 header record		CC=31 data record		CC=39 end of file record	
1	start of record	53	S	53	S	53	S
2	type of record	30	0	31	1	39	9
3 4	byte count	30 36	06	31 36	16	30 33	03
5 6 7 8	start address of data in this record	30 30 30 30	0000	31 31 30 30	1100	30 30 30 30	0000
9 10	data	34 38	48-H	39 38	98	46 43	FC (check sum)
	data	34 34	44-D	30 32	02		
	data	35 32	52-R	≈			
n	check sum	32 42	2B (check sum)	41 38	A8 (check sum)		

Check sum of header record above is complement of 1 of $(06 + 00 + 00 + 48 + 44 + 52)_{16}$ i.e., 2B.

The start address of data record is incremented for each one byte data, then is compared to the next address in data record and is checked to be sequential or not.

When it is not sequential, hexadecimal 00 is filled as data for that address automatically.

A example of type out of paper tape in HMCS6800 load module format is shown below.

```
header record ... S00600004844522B
data record .....S113F0007EF5587EF7897EFAA77EF9C07EF9C47E24
data record .....S112F010FA657EFA8B7EFAA07EF9DC7EFA247E06
end of file record .... S9030000FC
```

4) Four types of data of ROM code are able to be processed. In any case, header record before data record is needed and so as end of file record after data record.

(a) No vacancy in ROM

Data record is filled with full ROM record of one chip. Therefore address is sequential. Initial ROM address in "Confirmation sheet of specification" is 0.

(b) Vacancy in former part of ROM

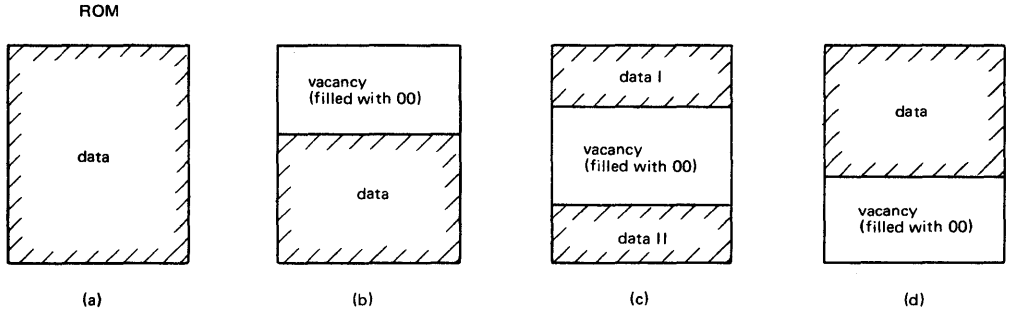
Desired initial address shall be filled in initial ROM address column in "Confirmation sheet of specification". Data of 00 are filled automatically for vacant address.

(c) Vacancy in the middle of ROM

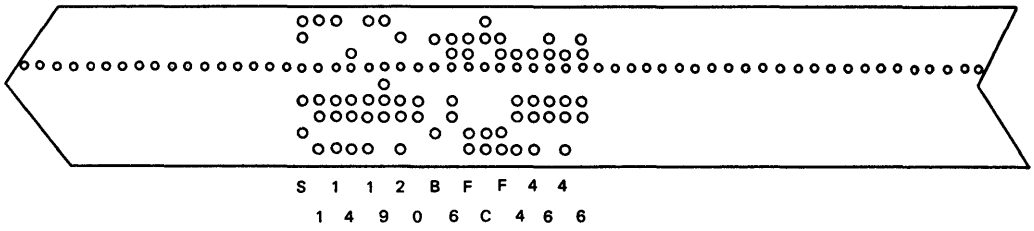
Data of 00 are filled in for vacant address. Initial ROM address for data I is 0 and desired initial address for data II shall be described in "Confirmation sheet of specification".

(d) Vacancy in later part of ROM

When end of file record is read out, data of 00 are filled in thereafter.



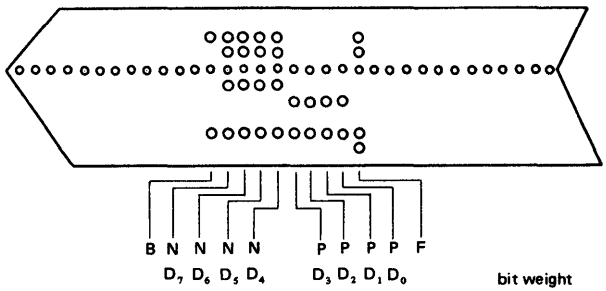
(Example) Paper tape whose data record is S1141920B6FC .....



- **BNPF format**

1) Each word is expressed as BNPF slice which begins word opening mark B, has 8 character bit contents shown by P or N and finishes with end mark F.

(Example) 0F in hexadecimal code is expressed as shown below (paper tape).



2) Any contents between F of the first slice and B of next slice are disregarded.
3) Bit pattern (BNPF) slice for all ROM address shall be indicated. Initial ROM address in "Confirmation sheet of specification" is, therefore always 0 for BNPF.
   - B ..... shows beginning of the word
   - N ..... shows 0 of one bit data
   - P ..... shows 1 of one bit data
   - F ..... shows end of the word

Note 1) X can be used expect for P and N for indication of word contents of BNPF slice. This X means that bit can be either P or N (don't care). X shall be determined by HITACHI for testing and shall be informed to the customer in confirmation table.

Note 2) Expression of B*nF can be used for indicating that the same contents of foregoing slice are applicable from this word to following n words.
For example, when B*4F is indicated at 10th word position, the contents of 9th word are repeated for 10, 11, 12 and 13th word.
(Content of X is not always repeated even in this case.)
n is grater than 1 and less than final address of ROM.

Note 3) When vacancy of ROM exists, combination of Note 1) and Note 2) is usefull.

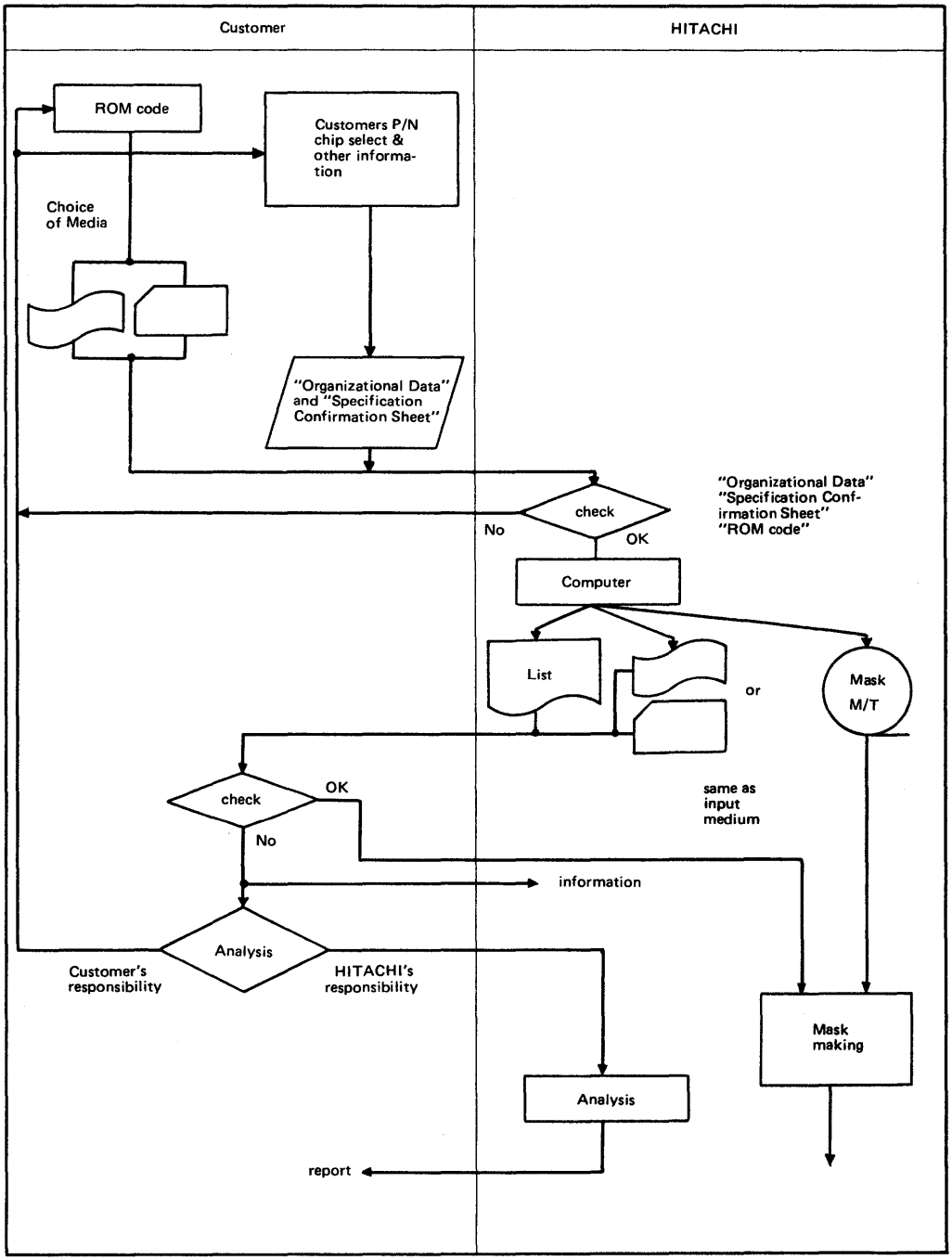


Figure 20 Flow chart of Mask ROM Development

HD6850, HD68A50

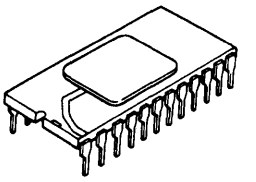
ACIA (Asynchronous Communications Interface Adapter)

The HD6850 Asynchronous Communications Interface Adapter provides the data formatting and control to interface serial asynchronous data communications information to bus organized systems such as the HMCS6800 Microprocessing Unit.

The bus interface of the HD6850 includes select, enable, read/write, interrupt and bus interface logic to allow data transfer over an 8-bit bi-directional data bus. The parallel data of the bus system is serially transmitted and received by the asynchronous data interface, with proper formatting and error checking.

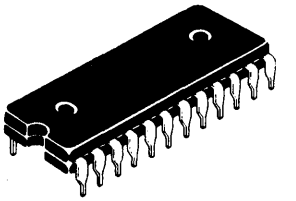
The functional configuration of the ACIA is programmed via the data bus during system initialization. A programmable Control Register provides variable word lengths, clock division ratios, transmit control, receive control, and interrupt control. For peripheral or modem operation three control lines are provided.

HD6850, HD68A50



(DC-24)

HD6850P, HD68A50P

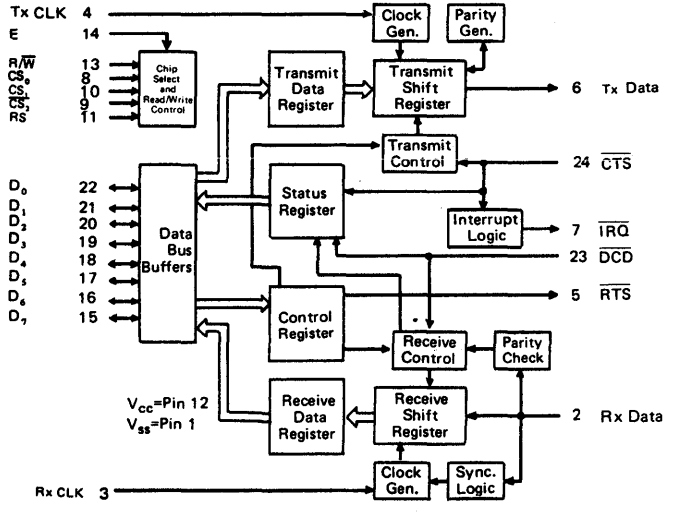


(DP-24)

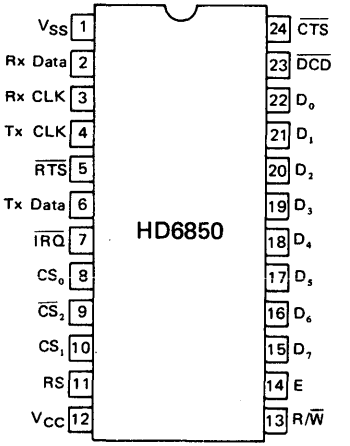
■ FEATURES

- Serial/Parallel Conversion of Data
- Eight and Nine-bit Transmission
- Insertion and Deleting of Start and Stop Bit
- Optional Even and Odd Parity
- Parity, Overrun and Framing Error Checking
- Peripheral/Modem Control Functions (Clear to Send $\overline{CTS}$, Request to Send $\overline{RTS}$, Data Carier Detect $\overline{DCD}$)
- Optional ÷ 1, ÷ 16, and ÷ 64 Clock Modes
- Up to 500kbps Transmission
- Programmable Control Register
- N-channel Silicon Gate Process
- Compatible with MC6850 and MC68A50

■ BLOCK DIAGRAM



■ PIN ARRANGEMENT



(Top View)

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC}*	-0.3 ~ +7.0	V
Input Voltage	V_{in}*	-0.3 ~ +7.0	V
Operating Temperature	T_{opr}	-20 ~ +75	°C
Storage Temperature	T_{stg}	-55 ~ +150	°C

* With respect to V_{SS} (SYSTEM GND)

(NOTE) Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ RECOMMENDED OPERATING CONDITIONS

Item	Symbol	min	typ	max	Unit
Supply Voltage	V_{CC}*	4.75	5.0	5.25	V
Input Voltage	V_{IL}*	-0.3	–	0.8	V
	V_{IH}*	2.0	–	V_{CC}	V
Operating Temperature	T_{opr}	-20	25	75	°C

* With respect to V_{SS} (SYSTEM GND)

■ ELECTRICAL CHARACTERISTICS

● DC CHARACTERISTICS (V_{CC}=5V±5%, V_{SS}=0V, Ta=-20~+75°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ*	max	Unit
Input "High" Voltage	All Inputs	V_{IH}		2.0	–	V_{CC}	V
Input "Low" Voltage	All Inputs	V_{IL}		-0.3	–	0.8	V
Input Leakage Current	R/$\overline{W}$, CS_0, CS_1, $\overline{CS_2}$, E	I_{in}	V_{in}=0~5.25V	-2.5	–	2.5	μA
Three-State (Off State) Input Current	D_0~D_7	I_{TSI}	V_{in}=0.4~2.4V	-10	–	10	μA
Output "High" Voltage	D_0~D_7	V_{OH}	I_{OH}=-205μA, Enable Pulse Width ≦ 25μs	2.4	–	–	V
	TxData, $\overline{RTS}$		I_{OH}=-100μA, Enable Pulse Width ≦ 25μs	2.4	–	–	
Output "Low" Voltage	All outputs	V_{OL}	I_{OL}=1.6mA, Enable Pulse Width ≦ 25μs	–	–	0.4	V
Output Leakage Current (Off State)	$\overline{IRQ}$	I_{LOH}	V_{OH}=2.4V	–	–	10	μA
Power Dissipation		P_D		–	300	525	mW
Input Capacitance	D_0~D_7	C_{in}	V_{in}=0V, T_a=25°C, f=1.0MHz	–	–	12.5	pF
	E, TxCLK, RxCLK, R/$\overline{W}$, $\overline{RS}$, RxData, CS_0, CS_1, $\overline{CS_2}$, $\overline{CTS}$, $\overline{DCD}$			–	–	7.5	
Output Capacitance	$\overline{RTS}$, TxData	C_{out}	V_{in}=0V, T_a=25°C, f=1.0MHz	–	–	10	pF
	$\overline{IRQ}$			–	–	5.0	

* T_a=25°C, V_{CC}=5V

• AC CHARACTERISTICS

1. TIMING OF DATA TRANSMISSION

Item		Symbol	Test Condition	min	typ	max	Unit
Minimum Clock Pulse Width	÷16, ÷64 Modes	PW_{CL}	Fig. 1	600	–	–	ns
	÷16, ÷64 Modes	PW_{CH}	Fig. 2	600	–	–	ns
Clock Frequency	÷1 Mode	f_c		–	–	500	kHz
	÷16, ÷64 Modes			–	–	800	
Clock-to-Data Delay for Transmitter		t_{TDD}	Fig. 3	–	–	1.0	μs
Receive Data Setup Time	÷ 1 Mode	t_{RDSU}	Fig. 4	500	–	–	ns
Receive Data Hold Time	÷ 1 Mode	t_{RDH}	Fig. 5	500	–	–	ns
$\overline{IRQ}$ Release Time		t_{IR}	Fig. 6	–	–	1.2	μs
$\overline{RTS}$ Delay Time		t_{RTS}	Fig. 6	–	–	1.0	μs
Rise Time and Fall Time	Except E	t_r, t_f		–	–	1.0*	μs

* 1.0 μs or 10% of the pulse width, whichever is smaller.

2. BUS TIMING CHARACTERISTICS

1) READ

Item	Symbol	Test Condition	HD6850			HD68A50			Unit
			min	typ	max	min	typ	max	
Enable Cycle Time	t_{cycE}	Fig. 7	1.0	–	–	0.666	–	–	μs
Enable "High" Pulse Width	PW_{EH}	Fig. 7	0.45	–	25	0.28	–	25	μs
Enable "Low" Pulse Width	PW_{EL}	Fig. 7	0.43	–	–	0.28	–	–	μs
Setup Time, Address and R/$\overline{W}$ valid to Enable positive transition	t_{AS}	Fig. 7	140	–	–	140	–	–	ns
Data Delay Time	t_{DDR}	Fig. 7	—	–	320	–	–	220	ns
Data Hold Time	t_H	Fig. 7	10	–	–	10	–	–	ns
Address Hold Time	t_{AH}	Fig. 7	10	–	–	10	–	–	ns
Rise and Fall Time for Enable Input	t_{Er}, t_{Ef}	Fig. 7	–	–	25	–	–	25	ns

2) WRITE

Item	Symbol	Test Condition	HD6850			HD68A50			Unit
			min	typ	max	min	typ	max	
Enable Cycle Time	t_{cycE}	Fig. 8	1.0	–	–	0.666	–	–	μs
Enable "High" Pulse Width	PW_{EH}	Fig. 8	0.45	–	25	0.28	–	25	μs
Enable "Low" Pulse Width	PW_{EL}	Fig. 8	0.43	–	–	0.28	–	–	μs
Setup Time, Address and R/$\overline{W}$ valid to Enable positive transition	t_{AS}	Fig. 8	140	–	–	140	–	–	ns
Data Setup Time	t_{DSW}	Fig. 8	195	–	–	80	–	–	ns
Data Hold Time	t_H	Fig. 8	10	–	–	10	–	–	ns
Address Hold Time	t_{AH}	Fig. 8	10	–	–	10	–	–	ns
Rise and Fall Time for Enable Input	t_{Er}, t_{Ef}	Fig. 8	–	–	25	–	–	25	ns

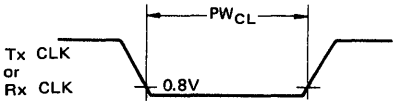


Figure 1 Clock Pulse Width, "Low" State

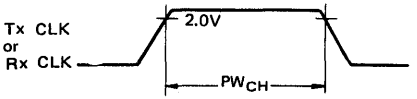


Figure 2 Clock Pulse Width, "High" State

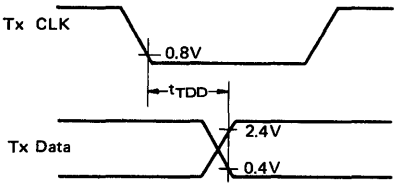


Figure 3 Transmit Data Output Delay

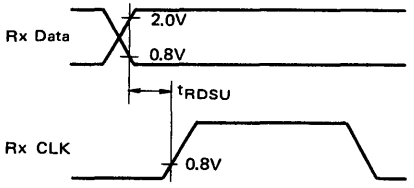


Figure 4 Receive Data Setup Time (÷1 Mode)

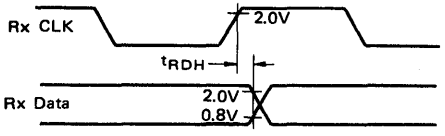


Figure 5 Receive Data Hold Time (÷1 Mode)

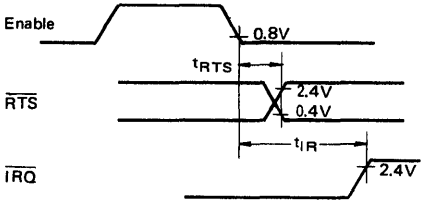


Figure 6 $\overline{RTS}$ Delay and $\overline{IRQ}$ Release Time

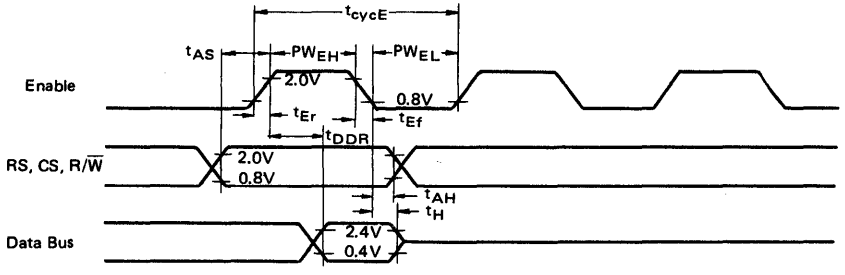


Figure 7 Bus Read Timing Characteristics
(Read information from ACIA)

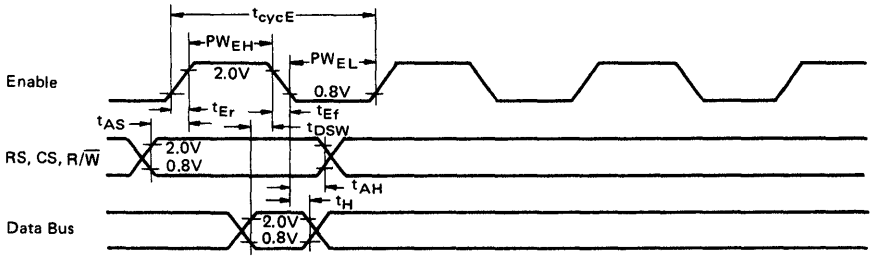
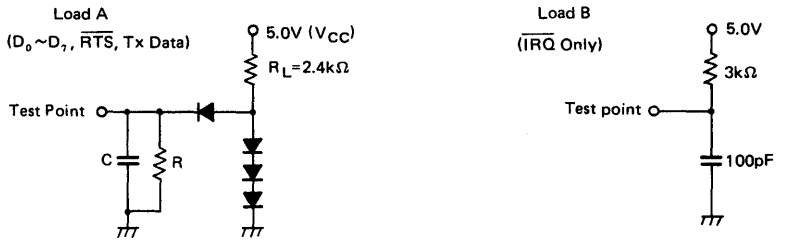


Figure 8 Bus Write Timing Characteristics
(Write information into ACIA)



C = 130pF for D_0~D_7
= 30pF for $\overline{RTS}$ and Tx Data
All diodes are 1S2074 Ⓗ or Equivalent.

R = 11kΩ for D_0~D_7
= 24kΩ for $\overline{RTS}$ and Tx Data

Figure 9 Bus Timing Test Loads

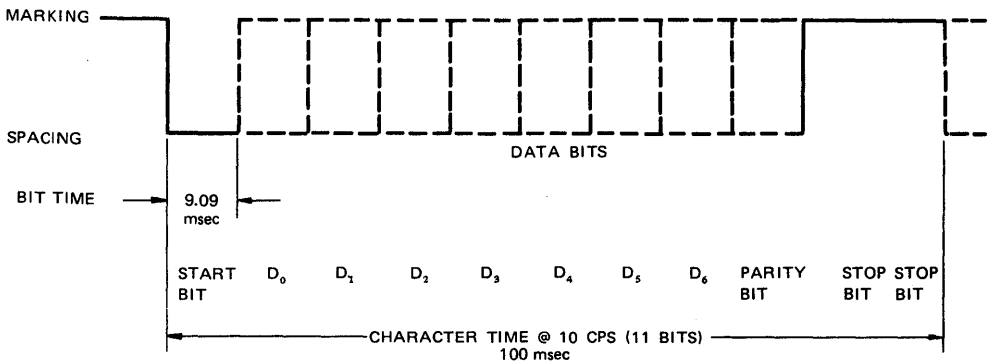
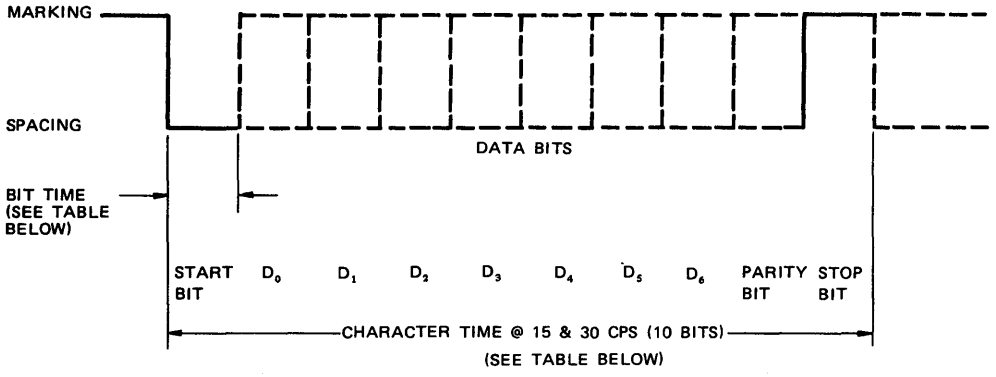


Figure 10 110 Baud Serial ASCII Data Timing



BAUD RATE	150	300
CHARACTERS/SEC	15	30
BIT TIME (msec)	6.67	3.33
CHARACTER TIME (msec)	66.7	33.3

$$\text{BIT TIME} = \frac{\text{SEC}}{\text{BAUD RATE}}$$

Figure 11 150 & 300 Baud Serial ASCII Data Timing

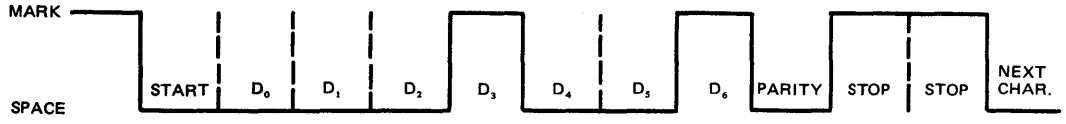


Figure 12 Send a 7 Bit ASCII Char. "H" Even Parity
– 2 Stop Bits $H = 48_{16} = 1001000_2$

■ DATA OF ACIA

HD6850 is an interface adapter which controls transmission and reception of Asynchronous serial data. Some examples of serial data are shown in Figs. 10 ~ 12.

■ INTERNAL STRUCTURE OF ACIA

HD6850(ACIA) provides the following; 8-bit Bi-directional Data Buses (D_0~D_7), Receive Data Input (Rx Data), Transmit Data Output (Tx Data), three Chip Selects (CS_0, CS_1, $\overline{CS_2}$), Register Select Input (RS), Two Control Input (Read/Write (R/$\overline{W}$), Enable(E) , Interrupt Request Output($\overline{IRQ}$), Clear-to-Send ($\overline{CTS}$) to control the modem, Request-to-Send ($\overline{RTS}$), Data Carrier Detect($\overline{DCD}$) and Clock Inputs(Tx CLK, Rx CLK) used for synchronization of received and transmitted data. This ACIA also provides four registers; Status Register, Control Register, Receive Register and Transmit Register.

24-pin dual-in-line type package is used for the ACIA. Internal Structure of ACIA is illustrated in Fig. 13.

■ ACIA OPERATION

• Master Reset

The master reset (CR0, CR1) should be set during system initialization to insure the reset condition and prepare for programming the ACIA functional configuration when the communications channel is required. Control bits CR5 and CR6 should also be programmed to define the state of $\overline{RTS}$ whenever master reset is utilized. After master resetting the ACIA, the programmable Control Register can be set for a number of options such as variable clock divider ratios, variable word length, one or two stop bits, parity (even, odd, or none), etc.

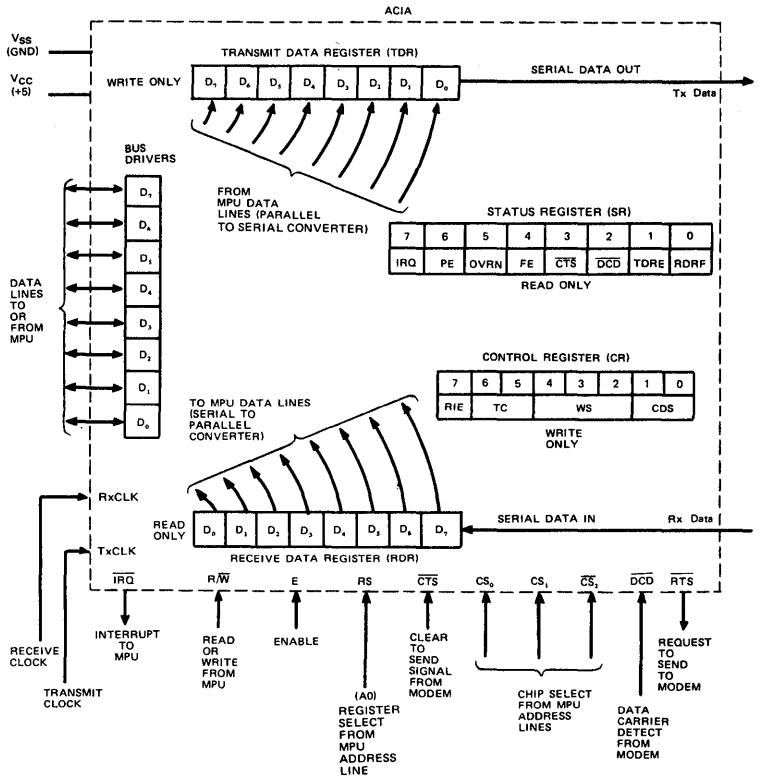


Figure 13 Internal Structure of ACIA

- **Transmit**

A typical transmitting sequence consists of reading the ACIA Status Register either as a result of an interrupt or in the ACIA's turn in a polling sequence. A character may be written into the Transmit Data Register if the status read operation has indicated that the Transmit Data Register is empty. This character is transferred to a Shift Register where it is serialized and transmitted from the Transmit Data output preceded by a start bit and followed by one or two stop bits. Internal parity (odd or even) can be optionally added to the character and will occur between the last data bit and the first stop bit. After the first character is written in the Data Register, the Status Register can be read again to check for a Transmit Data Register Empty condition and current peripheral status. If the register is empty, another character can be loaded for transmission even though the first character is in the process of being transmitted (because of double buffering). The second character will be automatically transferred into the Shift Register when the first character transmission is completed. This sequence continues until all the characters have been transmitted.

- **Receive**

Data is received from a peripheral by means of the Receive Data input. A divide-by-one clock ratio is provided for an externally synchronized clock (to its data) while the divide-by-16 and 64 ratios are provided for internal synchronization. Bit synchronization in the divide-by-16 and 64 modes is initiated by the detection of the leading mark-space transition of the start bit. False start bit delection capability insures that a full half bit of a start bit has been received before the internal clock is synchronized to the bit time. As a character is being received, parity (odd or even) will be checked and the error indication will be available in the Status Register along with framing error, overrun error, and Receive Data Register full. In a typical receiving sequence, the Status Register is read to determine if a character has been received from a peripheral. If the Receiver Data Register is full, the character is placed on the 8-bit ACIA bus when a Read Data command is received from the MPU. When parity has been selected for an 8-bit word (7 bits plus parity), the receiver strip the parity bit (D_7="0") so that data alone is transferred to the MPU. This feature reduces MPU programming. The Status Register can continue to be read again to determine when another character is available in the Receive Data Register. The receiver is also double buffered so that a character can be read from the data register as another character is being received in the Shift register. The above sequence continues until all characters have been received.

■ ACIA INTERNAL REGISTERS

The ACIA provides four registers; Transmit Data Register (TDR), Receive Data Register(RDR), Control Register(CR) and Status Register(SR). The content of each of the registers is summarized in Table 1.

Table 1 Definition of ACIA Register Contents

Buffer Address	RS=1 · R/$\overline{W}$=0 ****	RS=1 · R/$\overline{W}$=1	RS=0 · R/$\overline{W}$=0	RS=0 · R/$\overline{W}$=1
Data Bus	Transmit Data Register	Receiver Data Register	Control Register	Status Register
	(Write Only)	(Read Only)	(Write Only)	(Read Only)
0	Data Bit 0*	Data Bit 0	Counter Divide Select (CR0)	Rx Data Reg. Full (RDRF)
1	Data Bit 1	Data Bit 1	Counter Divide Select (CR1)	Tx Data Reg. Empty (TDRE)
2	Data Bit 2	Data Bit 2	Word Select 1 (CR2)	Data Carrier Detect ($\overline{DCD}$)
3	Data Bit 3	Data Bit 3	Word Select 2 (CR3)	Clear to Send ($\overline{CTS}$)
4	Data Bit 4	Data Bit 4	Word Select 3 (CR4)	Framing Error (FE)
5	Data Bit 5	Data Bit 5	Tx Control 1 (CR5)	Overrun (OVRN)
6	Data Bit 6	Data Bit 6	Tx Control 2 (CR6)	Parity Error (PE)
7	Data Bit 7***	Data Bit 7**	Rx Interrupt Enable (CR7)	Interrupt Request (IRQ)

* Leading bit = LSB = Bit 0
** Data bit will be zero in 7-bit plus parity modes.
*** Data bit is "don't care" in 7-bit plus parity modes.
**** 1 ... "High" level, 0 ... "Low" level

- **Transmit Data Register (TDR)**

Data is written in the Transmit Data Register during the negative transition of the enable (E) when the ACIA has been addressed and RS · R/$\overline{W}$ is selected. Writing data into the register causes the Transmit Data Register Empty bit in the Status Register to go "0". Data can then be transmitted. If the transmitter is idling and no character is being transmitted, then the transfer will take place within 2 bit time + several E cycles of the trailing edge of the Write command. If a character is being transmitted, the new data character will commence as soon as the previous character is complete. The transfer of data causes the Transmit Data Register Empty (TDRE) bit to indicate empty.

- **Receive Data Register (RDR)**

Data is automatically transferred to the empty Receive Data Register (RDR) from the receiver deserializer (a shift register) upon receiving a complete character. This event causes the Receive Data Register Full bit (RDRF) on the status buffer to go "1" (full). Data may then be read through the bus by addressing the ACIA and R/$\overline{W}$ "High" when the ACIA is enabled. The non-destructive read cycle causes the RDRF bit to be cleared to empty although the data is retained in the RDR. The status is maintained by RDRF as to whether or not the data is current. When the Receive Data Register is full, the automatic transfer of data from the Receiver Shift Register to the Data Register is inhibited and the RDR contents remain valid with its current status stored in the Status Register.

- **Control Register**

The ACIA Control Register consists of eight bits of write-only buffer that are selected when RS and R/$\overline{W}$ are "Low". This register controls the function of the receiver, transmitter, interrupt enables, and the Request-to-Send ($\overline{RTS}$) peripheral/modem control output.

Counter Divide Select Bits (CR0 and CR1)

The Counter Divide Select Bits (CR0 and CR1) determine the divide ratios utilized in both the transmitter and receiver section of the ACIA. Additionally, these bits are used to provide a master reset for the ACIA which clears the Status Register (except for external conditions on $\overline{CTS}$ and $\overline{DCD}$) and initializes both the receiver and transmitter. Master reset does not affect other Control Register bits. Note that after power-on or a power fail/restart, these bits must be set "1" to reset the ACIA. After resetting, the clock divide ratio may be selected. These counter select bits provide for the following clock divide ratios:

Table 2 Function of Counter Devide Select Bit

CR1	CR0	Function
0	0	÷1
0	1	÷16
1	0	÷64
1	1	Master Reset

Word Select Bits (CR2, CR3, and CR4)

The Word Select bits are used to select word length, parity, and the number of stop bits. The encoding format is as follows:

Table 3 Function of Word Select Bit

CR4	CR3	CR2	Function
0	0	0	7 Bits + Even Parity + 2 Stop Bits
0	0	1	7 Bits + Odd Parity + 2 Stop Bits
0	1	0	7 Bits + Even Parity + 1 Stop Bit
0	1	1	7 Bits + Odd Parity + 1 Stop Bit
1	0	0	8 Bits + 2 Stop Bits
1	0	1	8 Bits + 1 Stop Bit
1	1	0	8 Bits + Even Parity + 1 Stop Bit
1	1	1	8 Bits + Odd Parity + 1 Stop Bit

Word length, Parity Select, and Stop Bit changes are not buffered and therefore become effective immediately.

Transmitter Control Bits (CR5 and CR6)

Two Transmitter Control bits provide for the control of the interrupt from the Transmit Data Register Empty condition, the Request-to-Send ($\overline{RTS}$) output, and the transmission of a Break level (space). The following encoding format is used:

Table 4 Function of Transmitter Control-Bit

CR6	CR5	Function
0	0	$\overline{RTS}$ = "Low", Transmitting Interrupt Disabled.
0	1	$\overline{RTS}$ = "Low", Transmitting Interrupt Enabled.
1	0	$\overline{RTS}$ = "High", Transmitting Interrupt Disabled.
1	1	$\overline{RTS}$ = "Low", Transmits a Break level on the Transmit Data Output. Transmitting Interrupt Disabled.

Receive Interrupt Enable Bit (CR7)

The following interrupts will be enabled by a "1" in bit position 7 of the Control Register (CR7): Receive Data Register Full, Overrun, or a "Low" to "High" transistion on the Data Carrier Detect ($\overline{DCD}$) signal line.

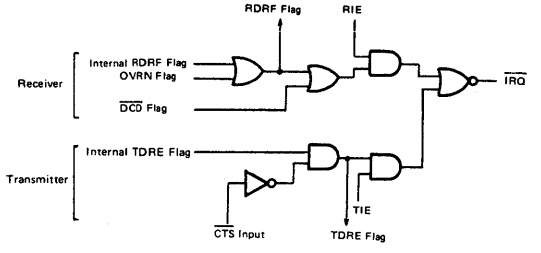


Fig. 14 $\overline{IRQ}$ Internal Circuit

- **Status Register**

Information on the status of the ACIA is available to the MPU by reading the ACIA Status Register. This read-only register is selected when RS is "Low" and R/$\overline{W}$ is "High". Information stored in this register indicates the status of the Transmit Data Register, the Receive Data Register and error logic, and the peripheral/modem status inputs of the ACIA.

Receive Data Register Full (RDRF), Bit 0

RDRF indicates that received data has been transferred to the Receive Data Register. RDRF is cleared after an MPU read of the Receive Data Register or by a master reset. The cleared or empty state indicates that the contents of the Receive Data Register are not current. Data Carrier Detect ($\overline{DCD}$) being "High" also causes RDRF to indicate empty.

Transmit Data Register Empty (TDRE), Bit 1

The Transmit Data Register Empty bit being set "1" indicates that the Transmit Data Register contents have been transferred and that new data may be entered. The "0" state indicates that the register is full and that transmission of a new character has not begun since the last write data command.

Data Carrier Detect ($\overline{DCD}$), Bit 2

The $\overline{DCD}$ bit will be "1" when the $\overline{DCD}$ input from a modem has gone "High" to indicate that a carrier is not present. This bit going "1" causes an Interrupt Request to be generated when the Receive Interrupt Enable is set. It remains "1" after the $\overline{DCD}$ input is returned "Low" until cleared by first reading the Status Register and then the Data Register or until a master reset occurs. If the $\overline{DCD}$ input remains "High" after read status and read data or master reset has occurred, the interrupt is cleared, the $\overline{DCD}$ status bit remains "1" and will follow the $\overline{DCD}$ input.

Clear-to-Send ($\overline{CTS}$), Bit 3

The $\overline{CTS}$ bit indicates the state of the $\overline{CTS}$ input from a modem. A "Low" $\overline{CTS}$ indicates that there is a $\overline{CTS}$ from the modem. In the "High state, the Transmit Data Register Empty bit is inhibited and the $\overline{CTS}$ status bit will be "1". Master reset does not affect the Clear-to-Send Status bit.

Framing Error (FE), Bit 4

FE indicates that the received character is improperly framed by a start and a stop bit and is detected by the absence of the 1st stop bit. This error indicates a synchronization error, faulty transmission, or a break condition. The FE flag is set or reset during the receive data transfer time. Therefore, this error indicator is present throughout the time that the associated character is available.

Receiver Overrun (OVRN), Bit 5

Overrun is an error flag that indicates that one or more characters in the data stream were lost. That is, a character or a number of characters were received but not read from the Receive Data Register (RDR) prior to subsequent characters being received. The overrun condition begins at the midpoint of the last bit of the second character received in succession without a read of the RDR having occurred. The overrun does not occur in the Status Register until the valid character prior to Overrun has been read. The RDRF bit remains set until the Overrun is reset. Character synchronization is maintained during the Overrun condition. The Overrun indication is reset after the reading of data from the Receive Data Register or by a Master Reset.

Parity Error (PE), Bit 6

The PE flag indicates that the number of "1"s (highs) in the character does not agree with the preselected odd or even parity. Odd parity is defined to be when the total number of ones is odd. The parity error indication will be present as long as the data character is in the RDR. If no parity is selected, then both the transmitter parity generator output and the receiver parity check results are inhibited.

Interrupt Request (IRQ), Bit 7

The IRQ bit indicates the state of the $\overline{IRQ}$ output. Any interrupt condition with its applicable enable will be indicated in this status bit. Anytime the $\overline{IRQ}$ output is "Low" the IRQ bit will be "1" to indicate the interrupt or service request status. IRQ is cleared by a read operation to the Receive Data Register or a write operation to the Transmit Data Register.

■ SIGNAL FUNCTIONS

• Interface Signal for MPU

Bi-Directional Data Bus (D_0~D_7)

The bi-directional data bus (D_0~D_7) allow for data transfer between the ACIA and the MPU. The data bus output drivers are three-state devices that remain in the high impedance (off) state except when the MPU performs an ACIA read operation.

Enable (E)

The Enable signal, E, is a high impedance TTL compatible input that enables the bus input/output data buffers and clocks data to and from the ACIA. This signal will normally be a derivative of the HMCS6800 ϕ_2 Clock.

Read/Write (R/$\overline{W}$)

The R/$\overline{W}$ line is a high impedance input that is TTL compatible and is used to control the direction of data flow through the ACIA's input/output data bus interface. When R/$\overline{W}$ is "High" (MPU Read cycle), ACIA output drivers are turned on and a selected register is read. When it is "Low", the ACIA output drivers are turned off and the MPU writes into a selected register. Therefore, the R/$\overline{W}$ signal is used to select read-only or write-only registers within the ACIA.

Chip Select (CS_0, CS_1, $\overline{CS_2}$)

These three high impedance TTL compatible input lines are used to address the ACIA. The ACIA is selected when CS_0 and CS_1 are "High" and $\overline{CS_2}$ is "Low". Transfers of data to and from the ACIA are then performed under the control of the Enable signal, Read/Write, and Register Select.

Register Select (RS)

The RS line is a high impedance input that is TTL compatible. A "High" level is used to select the Transmit/Receive Data Registers and a "Low" level the Control/Status Registers. The R/$\overline{W}$ signal line is used in conjunction with Register Select to select the read-only or write-only register in each register pair.

Interrupt Request ($\overline{IRQ}$)

$\overline{IRQ}$ is a TTL compatible, open-drain (no internal pullup), active "Low" output that is used to interrupt the MPU. The $\overline{IRQ}$ output remains "Low" as long as the cause of the interrupt is present and the appropriate interrupt enable within the ACIA is set.

Clock Inputs

Separate high impedance TTL compatible inputs are provided for clocking of transmitted and received data. Clock frequencies of 1, 16 or 64 times the data rate may be selected.

Transmit Clock (Tx CLK)

The Tx CLK input is used for the clocking of transmitted data. The transmitter initiates data on the negative transition of the clock.

Receive Clock (Rx CLK)

The Rx CLK input is used for synchronization of received data. (In the ÷ 1 mode, the clock and data must be synchronized externally.) The receiver samples the data on the positive transition of the clock.

• Serial Input/Output Lines

Receive Data (Rx Data)

The Rx Data line is a high impedance TTL compatible input through which data is received in a serial format. Synchronization with a clock for detection of data is accomplished internally when clock rates of 16 or 64 times the bit rate are used. Data rates are in the range of 0 to 500 kbps when external synchronization is utilized.

Transmit Data (Tx Data)

The Tx Data output line transfers serial data to a modem or other peripheral. Data rates in the range of 0 to 500 kbps when external synchronization is utilized.

Modem Control

The ACIA includes several functions that permit limited control of a peripheral or modem. The functions included are $\overline{CTS}$, $\overline{RTS}$ and $\overline{DCD}$.

Clear-to-Send ($\overline{CTS}$)

This high impedance TTL compatible input provides automatic control of the transmitting end of a communications link via the modem $\overline{CTS}$ active "Low" output by inhibiting the Transmit Data Register Empty (TDRE) status bit.

Request-to-Send ($\overline{RTS}$)

The $\overline{RTS}$ output enables the MPU to control a peripheral or modem via the data bus. The $\overline{RTS}$ output corresponds to the state of the Control Register bits CR5 and CR6. When CR6=0 or both CR5 and CR6=1, the $\overline{RTS}$ output is "Low" (the active state). This output can also be used for Data Terminal Ready ($\overline{DTR}$).

Data Carrier Detect ($\overline{DCD}$)

This high impedance TTL compatible input provides automatic control, such as in the receiving end of a communications link by means of a modem $\overline{DCD}$ output. The $\overline{DCD}$ input inhibits and initializes the receiver section of the ACIA when "High". A "Low" to "High" transition of the $\overline{DCD}$ initiates an interrupt to the MPU to indicate the occurrence of a loss of carrier when the Receive Interrupt Enable bit is set.

HD6852, HD68A52

SSDA (Synchronous Serial Data Adapter)

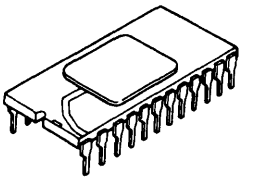
The HD6852 Synchronous Serial Data Adapter provides a bi-directional serial interface for synchronous data information interchange. It contains interface logic for simultaneously transmitting and receiving standard synchronous communications characters in bus organized systems such as the HMCS6800 Microprocessor systems.

The bus interface of the HD6852 includes select, enable, read/write, interrupt, and bus interface logic to allow data transfer over an 8-bit bi-directional data bus. The parallel data of the bus system is serially transmitted and received by the synchronous data interface with synchronization, fill character insertion/deletion, and error checking. The functional configuration of the SSDA is programmed via the data bus during system initialization.

Programmable control registers provide control for variable word length, transmit control, receive control, synchronization control and interrupt control. Status, timing and control lines provide peripheral or modem control.

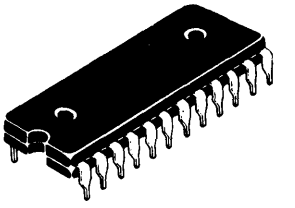
Typical applications include data communications terminals, floppy disk controllers, cassette or cartridge tape controllers and numerical control systems.

HD6852, HD68A52



(DC-24)

HD6852P, HD68A52P

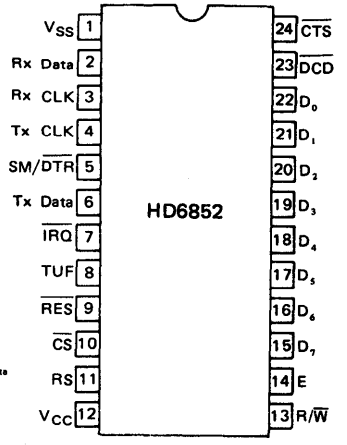


(DP-24)

■ FEATURES

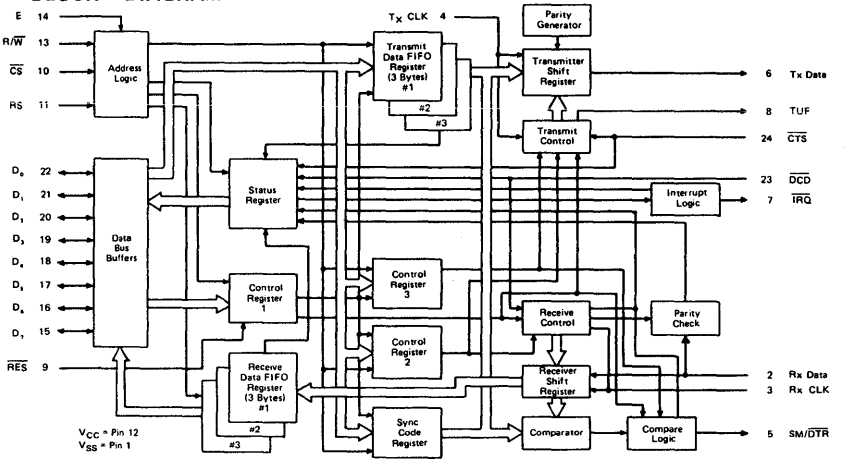
- Programmable Interrupts from Transmitter, Receiver, and Error Detection Logic
- Character Synchronization on One or Two Sync Codes
- External Synchronization Available for Parallel-Serial Operation
- Programmable Sync Code Register
- Up to 600kbps Transmitter
- Peripheral/Modem Control Functions
- Three Bytes of FIFO Buffering on Both Transmit and Receive
- 6, 7, or 8 Bit Data Transmission
- Optional Even and Odd Parity
- Parity, Overrun, and Underflow Status
- Compatible with MC6852 and MC68A52

■ PIN ARRANGEMENT



(Top View)

■ BLOCK DIAGRAM



■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC}*	−0.3 ~ +7.0	V
Input Voltage	V_{in}*	−0.3 ~ +7.0	V
Operating Temperature	T_{opr}	− 20 ~ + 75	°C
Storage Temperature	T_{stg}	−55 ~ +150	°C

* With respect to V_{SS} (SYSTEM GND)

(NOTE) Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ RECOMMENDED OPERATING CONDITIONS

Item	Symbol	min	typ	max	Unit
Supply Voltage	V_{CC}*	4.75	5.0	5.25	V
Input Voltage	V_{IL}*	−0.3	–	0.8	V
	V_{IH}*	2.0	–	V_{CC}	V
Operating Temperature	T_{opr}	− 20	25	75	°C

* With respect to V_{SS} (SYSTEM GND)

■ ELECTRICAL CHARACTERISTICS

• DC CHARACTERISTICS (V_{CC} = 5V ± 5%, V_{SS} = 0V, Ta = −20~+75°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ*	max	Unit
Input "High" Voltage	All Input	V_{IH}	–	2.0	–	–	V
Input "Low" Voltage	All Input	V_{IL}	–	−0.3	–	0.8	V
Output "High" Voltage	D_0~D_7	V_{OH}	I_{OH} = −205 μA, PW_{EH}, $PW_{EL} \leqq$ 25 μs	2.4	–	–	V
	Tx Data $\overline{DTR}$, TUF	V_{OH}	I_{OH} = −100 μA, PW_{EH}, $PW_{EL} \leqq$ 25 μs	2.4	–	–	V
Output "Low" Voltage	All Output	V_{OL}	I_{OL} = 1.6 mA, PW_{EH}, $PW_{EL} \leqq$ 25 μs	–	–	0.4	V
Input Leakage Current	TxCLK, RxCLK, Rx Data, E, $\overline{RES}$, RS, R/$\overline{W}$ $\overline{CS}$, $\overline{DCD}$, $\overline{CTS}$	I_{in}	V_{in} = 0 ~ 5.25 V	−2.5	–	2.5	μA
Three-State Input Current (Off State)	D_0~ D_7	I_{TSI}	V_{in} = 0.4 ~ 2.4V, V_{CC} = 5.25V	−10	–	10	μA
Output Leakage Current (Off State)	$\overline{IRQ}$	I_{LOH}	V_{OH} = 2.4V	–	–	10	μA
Power Dissipation		P_D		–	300	525	mW
Input Capacitance	D_0~ D_7	C_{in}	V_{in} = 0V, Ta = 25°C, f = 1 MHz	–	–	12.5	pF
	RxData, RxCLK, TxCLK, $\overline{RES}$, $\overline{CS}$, RS, R/$\overline{W}$, E, $\overline{DCD}$, $\overline{CTS}$			–	–	7.5	
Output Capacitance	TxData, $\overline{DTR}$, TUF,	C_{out}	V_{in} = 0V, Ta = 25°C f = 1 MHz	–	–	10	pF
	$\overline{IRQ}$			–	–	5.0	

* Ta = 25°C, V_{CC} = 5V

- **AC CHARACTERISTICS** (V_{CC}=5V±5%, V_{SS}=0V, Ta=-20~+75°C, unless otherwise noted.)

1. TIMING OF THE DATA TRANSFER

Item	Symbol	Test Condition	HD6852			HD68A52			Unit
			min	typ	max	min	typ	max	
Clock "Low" Pulse Width	PW_{CL}	Fig. 1	700	–	–	400	–	–	ns
Clock "High" Pulse Width	PW_{CH}	Fig. 2	700	–	–	400	–	–	ns
Clock Frequency	f_C		–	–	600	–	–	1,000	kHz
Receive Data Setup Time	t_{RDSU}	Fig. 3,7	350	–	–	200	–	–	ns
Receive Data Hold Time	t_{RDH}	Fig. 3	350	–	–	200	–	–	ns
Sync Match Delay Time	t_{SM}	Fig. 3	–	–	1.0	–	–	0.666	μs
Clock-to-Data Delay for Transmitter	t_{TDD}	Fig. 4,6	–	–	1.0	–	–	0.666	μs
Transmitter Underflow	t_{TUF}	Fig. 4	–	–	1.0	–	–	0.666	μs
$\overline{DTR}$ Delay Time	t_{DTR}	Fig. 5	–	–	1.0	–	–	0.666	μs
$\overline{IRQ}$ Release Time	t_{IR}	Fig. 5	–	–	1.2	–	–	0.8	μs
$\overline{RES}$ Pulse Width	t_{RES}		1.0	–	–	0.666	–	–	μs
$\overline{CTS}$ Setup Time	t_{CTS}	Fig. 6	200	–	–	150	–	–	ns
$\overline{DCD}$ Setup Time	t_{DCD}	Fig. 7	500	–	–	350	–	–	ns
Input Rise and Fall Times(Except E)	t_r, t_f	0.8V to 2.0V	–	–	1.0*	–	–	1.0*	μs

* 1.0μ or 10% of the pulse width, whichever is smaller.

2. BUS TIMING

1) READ

Item	Symbol	Test Condition	HD6852		HD68A52		Unit
			min	max	min	max	
Enable Cycle Time	t_{cycE}	Fig. 8	1.0	–	0.666	–	μs
Enable "High" Pulse Width	PW_{EH}		0.45	25	0.28	25	μs
Enable "Low" Pulse Width	PW_{EL}		0.43	–	0.28	–	μs
Setup Time, Address and R/$\overline{W}$ valid to Enable positive transition	t_{AS}		140	–	140	–	ns
Data Delay Time	t_{DDR}		–	320	–	220	ns
Data Hold Time	t_H		10	–	10	–	ns
Address Hold Time	t_{AH}		10	80	10	80	ns
Rise and Fall Time for Enable input	t_{Er}, t_{Ef}		–	25	–	25	ns

2) WRITE

Item	Symbol	Test Condition	HD6852		HD68A52		Unit
			min	max	min	max	
Enable Cycle Time	t_{cycE}	Fig. 9	1.0	–	0.666	–	μs
Enable Pulse Width, "High"	PW_{EH}		0.45	25	0.28	25	μs
Enable Pulse Width, "Low"	PW_{EL}		0.43	–	0.28	–	μs
Setup Time, Address and R/$\overline{W}$ valid to Enable positive transition	t_{AS}		140	–	140	–	ns
Data Setup Time	t_{DSW}		195	–	80	–	ns
Data Hold Time	t_H		10	–	10	–	ns
Address Hold Time	t_{AH}		10	–	10	–	ns
Rise and Fall Time for Enable input	t_{Er}, t_{Ef}		–	25	–	25	ns

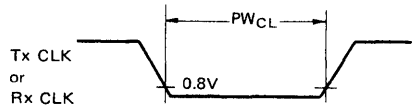


Figure 1 Clock Pulse Width ("Low" level)

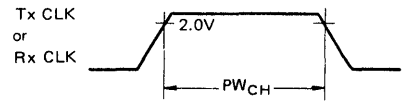


Figure 2 Clock Pulse Width ("High" level)

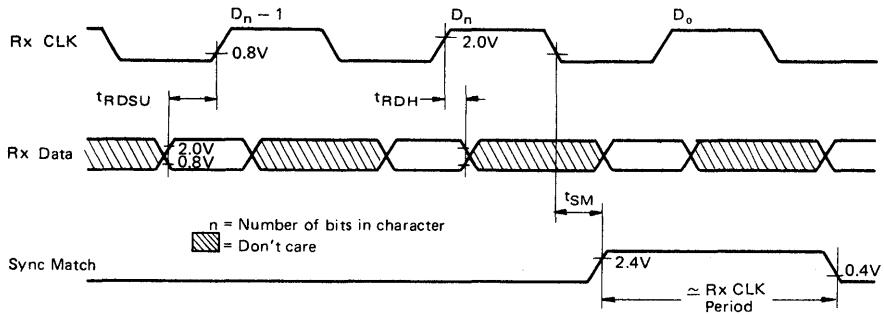


Figure 3 Receive Data Setup and Hold Times and Sync Match Delay Time

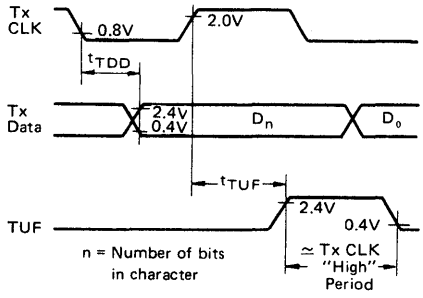
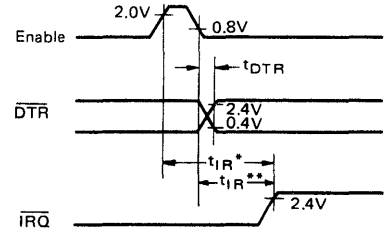


Figure 4 Transmit Data Output Delay and Transmitter Underflow Delay Time



* $\overline{IRQ}$ Release Time applied to TxData FIFO write operation and RxData FIFO read operation.

** $\overline{IRQ}$ Release Time applied to write "1" operation to RxRS, TxRS, CTUF, Clear $\overline{CTS}$ bits.

Figure 5 $\overline{DTR}$ and $\overline{IRQ}$ Release Time

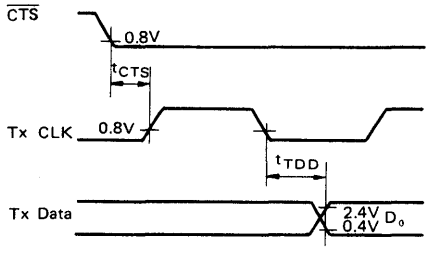
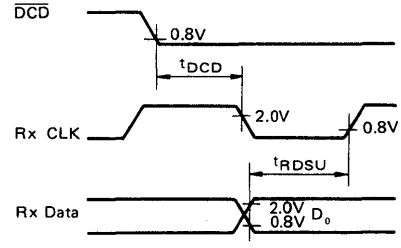


Figure 6 $\overline{CTS}$ Setup Time



At least two Rx CLK pulse should be input after the last bit of the last data before the next falling edge of $\overline{DCD}$ occurs.

Figure 7 $\overline{DCD}$ Setup Time

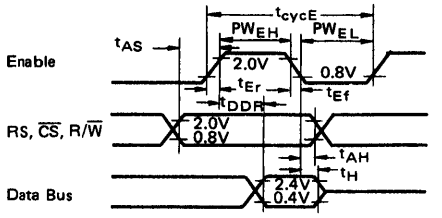


Figure 8 Bus Read Timing Characteristics (Read information from SSDA)

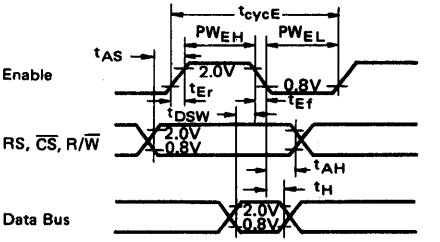


Figure 9 Bus Write Timing Characteristics (Write information into SSDA)

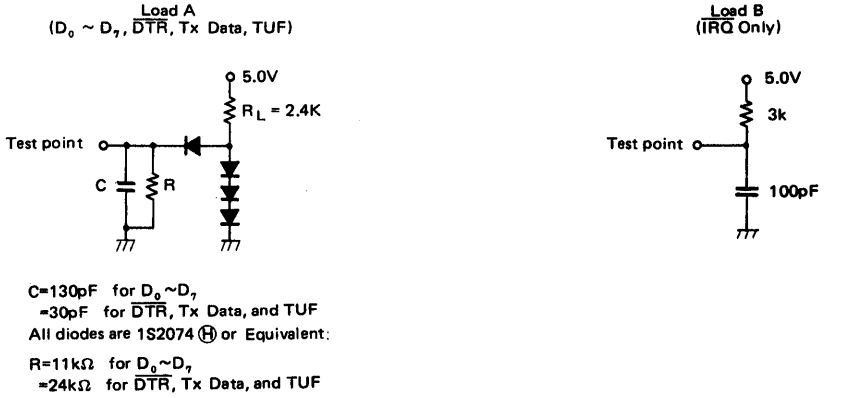


Figure 10 Test Loads

■ DEVICE OPERATION

At the bus interface, the SSDA appears as two addressable memory locations. Internally, there are seven registers: two read-only and five write-only registers. The read-only registers are Status and Receive Data; the write-only registers are Control 1, Control 2, Control 3, Sync Code and Transmit Data. The serial interface consists of serial input and output lines with independent clocks, and four peripheral/modem control lines.

Data to be transmitted is transferred directly into the 3-byte Transmit Data First-In First-Out (FIFO) Register from the data bus. Availability of the input to the FIFO is indicated by a bit in the Status Register; once data is entered, it moves through the FIFO to the last empty location. Data at the output of the FIFO is automatically transferred from the FIFO to the Transmitter Shift Register as the shift register becomes available to transmit the next character. If data is not available from the FIFO (underflow condition), the Transmitter Shift Register is automatically loaded with either a sync code or an all "1"s character. The transmit section may be programmed to append even, odd, or no parity to the transmitted word. An external control line ($\overline{CTS}$) is provided to inhibit the transmitter without clearing the FIFO.

Serial data is accumulated in the receiver based on the synchronization mode selected. In the external sync mode used for parallel-serial operation, the receiver is synchronized by the Data Carrier Detect ($\overline{DCD}$) input and transfers successive bytes of data to the input of the Receiver FIFO. The single-sync-character mode requires that a match occur between the Sync Code Register and one incoming character before data transfer to the FIFO begins. The two-sync-character mode requires that two sync codes be received in sequence to establish synchronization. Subsequent to synchronization in any mode, data is accumulated in the shift register, and parity is optionally checked. An indication of parity error is carried through the Receiver FIFO with each character to the last empty location. Availability of a word at the FIFO output is indicated by a bit in the Status Register, as is a parity error.

The SSDA and its internal registers are selected by the address bus, Read/Write (R/$\overline{W}$) and Enable control lines. To configure the SSDA, Control Registers are selected and the appropriate bits set. The Status Register is addressable for reading status.

Other I/O lines, in addition to Clear-to-Send ($\overline{CTS}$) and Data Carrier Detect ($\overline{DCD}$), include Sync Match/Data Terminal Ready (SM/$\overline{DTR}$) and Transmitter Underflow (TUF). The transmitter and receiver each have individual clock inputs allowing simultaneous operation under separate clock control. Signals to the microprocessor are the Data bus and Interrupt Request ($\overline{IRQ}$).

- **Initialization**

During a power-on sequence, the SSDA is reset via the $\overline{RES}$ input and internally latched in a reset condition to prevent erroneous output transmissions. The Sync Code Register, Control Register 2, and Control Register 3 should be programmed prior to the programmed release of the Transmitter and/or Receiver Reset bits; these bits in Control Register 1 should be cleared after the $\overline{RES}$ line has gone "High".

- **Transmitter Operation**

Data is transferred to the transmitter section in parallel form by means of the data bus and Transmit Data FIFO. The Transmit Data FIFO is a 3-byte register whose status is indicated by the Transmitter Data Register Available status bit (TDRA) and its associated interrupt enable bit. Data is transferred through the FIFO on negative edges of Enable (E) pulses. Two data transfer modes are provided in the SSDA. The 1-byte transfer mode provides for writing data to the transmitter section (and reading from the receiver section) one byte at a time. The 2-byte transfer mode provides for writing two data characters in succession.

Data will automatically transfer from the last register location in the Transmit Data FIFO (when it contains data) to the Transmitter Shift Register during the last half of the last bit of the previous character. A character is transferred into the Shift Register by the Transmitter Clock. Data is transmitted LSB first, and odd or even parity can be optionally appended. The unused bit positions in short word length characters from the data bus are "don't cares". (Note: The data bus inputs may be reversed for applications requiring the MSB to be transferred taken, e.g., IBM format for floppy disks; however, care must be taken to properly program the control registers – Table 1 will have its bit positions reversed.)

When the Shift Register becomes empty, and data is not available for transfer from the Transmit Data FIFO, an "underflow" occurs, and a character is inserted into the transmitter data stream to maintain character synchronization. The character transmitted on underflow will be either a "Mark" (all "1"s) or the contents of the Sync Code Register, depending upon the state of the Transmit Sync Code on Underflow control bit. The underflow condition is indicated by a pulse ($\approx$ Tx CLK "High" period) on the Underflow putput (when in Tx Sync on underflow mode). The Underflow output occurs coincident with the transfer of the last half of the last bit preceding the underflow character. The Underflow status bit is set until cleared by means of the Clear Underflow control bit. This output may be used in floppy disk systems to synchronize write operations and for appending CRCC.

Transmission is initiated by clearing the Transmitter Reset bit in Control Register 1. When the Transmitter Reset bit is cleared, the first full positive half-cycle of the Transmit Clock will initiate the transmit cycle, with the transmission of data or underflow characters beginning on the negative edge of the Transmit Clock pulse which started the cycle. If the Transmit Data FIFO was not loaded, an underflow character will be transmitted.

The Clear-to-Send ($\overline{CTS}$) input provides for automatic control of the transmitter by means of external system hardware; e.g., the modem $\overline{CTS}$ output provides the control in a data communications system. The $\overline{CTS}$ input resets and inhibits the transmitter section when "High", but does not reset the Transmit Data FIFO. The TDRA status bit is inhibited by $\overline{CTS}$ being "High" in either the one-sync character or two-sync-character mode of operation.

In the external sync mode, TDRA is unaffected by $\overline{CTS}$ in order to provide Transmit Data FIFO status for preloading and operating the transmitter under the control of the $\overline{CTS}$ input. When the Transmitter Reset bit (Tx Rs) is set, the Transmit Data FIFO is cleared and the TDRA status bit is cleared. After one E clock has occurred, the Transmit Data FIFO becomes available for new data with TDRA inhibited.

- **Receiver Operation**

Data and a presynchronized clock are provided to the SSDA receiver section by means of the Receive Data (Rx Data) and Receive Clock (Rx CLK) inputs. The data is a continuous stream of binary data bits without means for identifying character boundaries within the stream. It is, therefore, necessary to achieve character synchronization for the data at the beginning of the data block. Once synchronization is achieved, it is assumed to be retained for all successive characters within the block.

Data communications systems utilize the detection of sync codes during the initial portion of the preamble to establish character synchronization. This requires the detection of a single code or two successive sync codes. Floppy disk and cartridge tape units require sixteen bits of defined preamble and cassettes require eight bits of preamble to establish the reference for the start of record. All three are functionally equivalent to the detection of sync codes. Systems which do not utilize code detection techniques require custom logic external to the SSDA for character synchronization and use of the parallel-to-serial (external sync) mode.
(Note: The Receiver Shift Register is set to ones when reset)

- **Synchronization**

The SSDA provides three operating modes with respect to character synchronization: one-sync-character mode, two-sync-character mode, and external sync mode. The external sync mode requires synchronization and control of the receiving section through the Data Carrier Detect ($\overline{DCD}$) input. This external synchronization could consist of direct line control from the transmitting end of the serial data link or from external logic designed to detect the start of the message block. The one-sync-character mode searches on a bit-by-bit basis until a match is achieved between the data in the Shift Register and the Sync Code Register. The match indicates character synchronization is complete and will be retained for the message block. In the two-sync-character mode, the receiver searches for the first sync code match on a bit-by-bit basis and then looks for a second successive sync code character prior to establishing character synchronization. If the second sync code character is not received, the bit-by-bit search for the first sync code is resumed.

Sync codes received prior to the completion of synchronization (one or two character) are not transferred to the Receive Data FIFO. Redundant sync codes during the preamble or sync codes which occure as "fill characters" can automatically be stripped from the data, when the Strip Sync control bit is set, to minimize system loading. The character synchronization will be retained until cleared by means of the Clar Sync bit, which also inhibits synchronization search when set.

- **Receiving Data**

Once synchronization has been achieved, subsequent characters are automatically transferred into the Receive Data FIFO and clocked through the FIFO to the last empty location by E pulses (MPU System $\phi 2$). The Receiver Data Available status bit

(RDA) indicates when data is available to be read from the last FIFO location (#3) when in the 1-byte transfer mode. The 2-byte transfer mode causes the RDA status bit to indicate data is available when the last two FIFO register locations are full. Data being available in the Receive Data FIFO causes an interrupt request if the Receiver Interrupt Enable (RIE) bit is set. The MPU will then read the SSDA Status Register, which will indicate that data is available for the MPU read from the Receiver Data FIFO register. The $\overline{IRQ}$ and RDA status bits are reset by a read from the FIFO. If more than one character has been received and is resident in the Receive Data FIFO, subsequent E clocks will cause the FIFO to update and the RDA and $\overline{IRQ}$ status bits will again be set. The read data operation for the 2-byte transfer mode requires an intervening E clock between reads to allow the FIFO data to shift. Optional parity is automatically checked as data is received, and the parity status condition is maintained with each character until the data is read from the Receive Data FIFO. Parity errors will cause an interrupt request if the Error Interrupt Enable (EIE) has been set. The parity bit is not transferred to the data bus but must be checked in the Status Register. NOTE: In the 2-byte transfer mode, parity should be checked prior to reading the second byte, since a FIFO read clears the error bit.

Other status bits which pertain to the receiver section are Receiver Overrun and Data Carrier Detect ($\overline{DCD}$). The Overrun status bit is automatically set when a transfer of a character to the Receive Data FIFO occurs and the first register of the Receive Data FIFO is full. Overrun causes an interrupt if Error Interrupt Enable (EIE) has been set. The transfer of the overrunning character into the FIFO causes the previous character in the FIFO input register location to be lost. The Overrun status bit is cleared by reading the Status Register (when the overrun condition is present), followed by a Receive Data FIFO Register read. Overrun cannot occur and be cleared without providing an opportunity to detect its occurrence via the Status Register.

A positive transition on the $\overline{DCD}$ input causes an interrupt if the EIE control bit has been set. The interrupt caused by $\overline{DCD}$ is cleared by reading the Status Register when the $\overline{DCD}$ status bit is "1", followed by a Receive Data FIFO read. The $\overline{DCD}$ status bit will subsequently follow the state of the $\overline{DCD}$ input when it goes "Low".

■ SSDA REGISTERS

Seven registers in the SSDA can be accessed by means of the bus. The registers are defined as read-only or write-only according to the direction of information flow. The Register Select (RS) input selects two registers in each state, one being read-only and the other write-only. The Read/Write (R/$\overline{W}$) input defined which of the two selected registers will actually be accessed. Four registers (two read-only and two write-only) can be addressed via the bus at any particular time. These registers and the required addressing are defined in Table 1.

• Control Register 1 (C1)

Control Register 1 is an 8-bit wirte-only register that can be directly addressed from the data bus. Control Register 1 is addressed when RS = "Low" and R/$\overline{W}$ = "Low".

Receiver Reset (Rx Rs), C1 Bit 0

The Receiver Reset control bit provides both a reset and inhibit function to the receiver section. When Rx Rs is set, it clears the receiver control logic, error logic, Rx Data FIFO Control, Parity Error status bit, and $\overline{DCD}$ interrupt. The Receiver Shift Register is set ones. The Rx Rs bit must be cleared after the occurrence of a "Low" level on $\overline{RES}$ in order to enable the receiver section of the SSDA.

Transmitter Reset (Tx Rs), C1 Bit 1

The Transmitter Reset control bit provides both a reset and inhibit to the transmitter section. When Tx Rs is set, it clears the transmitter control section, Transmitter Shift Register, Tx Data FIFO Control (the Tx Data FIFO can be reloaded after one E clock pulse), the Transmitter Underflow status bit, and the $\overline{CTS}$ interrupt, and inhibits the TDRA status bit (in the one-sync-character and two-sync-character modes). The Tx Rs bit must be cleared after the occurrence of a "Low" level on $\overline{RES}$ in order to enable the transmitter section of the SSDA. If the Tx FIFO is not preloaded, it must be loaded immediately after the Tx Rs release to prevent a transmitter underflow condition.

Strip Synchronization Characters (Strip Sync), C1 Bit 2

If the Strip Sync bit is set, the SSDA will automatically strip all received characters which match the contents of the Sync Code Register. The characters used for synchronization (one or two characters of sync) are always stripped from the received data stream.

Clear Synchronization (Clear Sync), C1 Bit 3

The Clear Sync control bit provides the capability of dropping receiver character synchronization and inhibiting resynchronization. The Clear Sync bit is set to clear and inhibit receiver synchronization in all modes and is reset to zero to enable resynchronization.

Transmitter Interrupt Enable (TIE), C1 Bit 4

TIE enable both the Interrupt Request ($\overline{IRQ}$) output and Interrupt Request status bit to indicate a transmitter service request. When TIE is set and the TDRA status bit is "1", the $\overline{IRQ}$ output will go "Low" (the active state) and the $\overline{IRQ}$ status bit will go "1".

Receiver Interrupt Enable (RIE), C1 Bit 5

RIE enable both the Interrupt Request output ($\overline{IRQ}$) and the Interrupt Request status bit to indicate a receiver service request. When RIE is set and the RDA status bit is "1", the $\overline{IRQ}$ output will go "Low" (the active state) and the $\overline{IRQ}$ status bit will go "1".

Address Control 1 (AC1) and Address Control 2 (AC2), C1 Bits 6 and 7

AC1 and AC2 select one of the write-only registers – Control 2, Control 3, Sync Code, or Tx Data FIFO – as shown in Table 1, when RS = "High" and R/$\overline{W}$ = "Low".

• Control Register 2 (C2)

Control Register 2 is an 8-bit write-only register which can be programmed from the bus when the Address Control bits in Control Register 1 (AC1 and AC2) are reset, RS = "High" and R/$\overline{W}$ = "Low".

Peripheral Control 1 (PC1) and Peripheral Control 2 (PC2), C2 Bits 0 and 1

Two control bits, PC1 and PC2, determine the operating characteristics of the Sync Match/$\overline{DTR}$ output. PC1, when "High", selects the Sync Match mode. PC2 provides the inhibit/

enable control for the SM/$\overline{DRT}$ output in the Sync Match mode. A one-bit-wide pulse is generated at the output when PC2 is "0", and a match occurs between the contents of the Sync Code Register and the incoming data even if sync is inhibited (Clear Sync bit = "1"). The Sync Match pulse is referenced to the negative edge of Rx CLK pulse causing the match.

The Data Terminal Ready ($\overline{DTR}$) mode is selected when PC1 is "0". When PC2 = "1" the SM/$\overline{DTR}$ output = "Low" and vice versa. The operation of PC2 and PC1 is summarized in Table 4.

1-Byte/2-Byte Transfer (1-Byte/2-Byte), C2 Bit 2

When 1-Byte/2-Byte is set, the TDRA and RDA status bits will indicate the availability if their respective data FIFO registers for a single byte data transfer. Alternately, if 1-Byte/2-Byte is reset, the TDRA and RDA status bits indicate when two bytes of data can be moved without a second status read. An intervening Enable pulse must occur between data transfers.

Word Length Selects (WS1, WS2, WS3), C2 Bits 3, 4, 5

Word length Select bits WS1, WS2, and WS3 select word length of 7, 8, or 9 bits including parity as shown in Table 3.

Transmit Sync Code on Underflow (Tx Sync), C2 Bit 6

When Tx Sync is set, the transmitter will automatically send a sync character when data is not available for transmission. If Tx Sync is reset, the transmitter will transmit a Mark character (including the parity bit position) on underflow. When the underflow is detected, a pulse approximately a Tx CLK "High" period wide will occur on the underflow output if the Tx Sync bit is "1". Internal parity generation is inhibited during underflow except for sync code fill character transmission in 8 bit plus parity word lengths.

Error Interrupt Enable (EIE), C2 Bit 7

When EIE is set, the $\overline{IRQ}$ status bit will go "1" and the $\overline{IRQ}$ output will go "Low" if:

1) A receiver overrun occurs. The interrupt is cleared by reading the Status Register and reading the Rx Data FIFO.
2) $\overline{DCD}$ input has gone to a "High". The interrupt is cleared by reading the Status Register and reading the Rx Data FIFO.
3) A parity error exists for the character in the last location (#3) of the Rx Data FIFO. The interrupt is cleared by reading the Rx Data FIFO. The interrupt is cleared by reading the Rx Data FIFO.
4) The $\overline{CTS}$ input has gone to a "High". The interrupt is cleared by writing a "1" in the Clear $\overline{CTS}$ bit, C3 bit 2, or by a Tx Reset.
5) The transmitter has underflowed (in the Tx Sync on Underflow mode). The interrupt is cleared by writing a "1" into the Clear Underflow, C3 bit 3, or Tx Reset.

When EIE is a "0", the $\overline{IRQ}$ status bit and the $\overline{IRQ}$ output are disabled for the above error conditions. A "Low" level on the $\overline{RES}$ input resets EIE to "0".

- **Control Register 3 (C3)**

Control Register 3 is a 4-bit write-only register which can be programmed from the bus when RS = "High" and R/$\overline{W}$ = "Low" and Address Control bit AC1 = "1" and AC2 = "0".

External/Internal Sync Mode Control (E/I Sync), C3 Bit 0

When the E/I Sync Mode bit is "1", the SSDA is in the external sync mode and the receiver synchronization logic is disabled. Synchronization can be achieved by means of the $\overline{DCD}$ input or by starting Rx CLK at the midpoint of data bit "0" of a character with $\overline{DCD}$ "Low". Both the transmitter and receiver sections operate as parallel – serial converters in the External Sync mode. The Clear Sync bit in Control Register 1 acts as a receiver sync inhibit when "High" to provide a bus controllable inhibit. The Sync Code Register can serve as a transmitter fill character register and a receiver match register in this mode. A "Low" on the $\overline{RES}$ input resets the E/I Sync Mode bit placing the SSDA In the internal sync mode.

One-Sync-Character/Two-Sync-Character Mode, Control (1 Sync/2 Sync), C3 Bit 1

When the 1 Sync/2 Sync bit is set, the SSDA will synchronize on a single match between the received data and the contents of the Sync Code Register. When the 1 Sync/2 Sync bit is reset, two successive sync characters must be received prior to receiver synchronization. If the second sync character is not detected, the bit by bit search resumes from the first bit in the second character. See the description of the Sync Code Register for more details.

Clear $\overline{CTS}$ Status (Clear $\overline{CTS}$), C3 Bit 2

When a "1" is written into the Clear $\overline{CTS}$ bit, the stored status and interrupt are cleared. Subsequently, the $\overline{CTS}$ status bit reflects the state of the $\overline{CTS}$ input. The Clear $\overline{CTS}$ control bit does not affect the $\overline{CTS}$ input nor its inhibit of the transmitter secton. The Clear $\overline{CTS}$ command bit is self-clearing, and writing a "0" into this bit is a nonfunctional operation.

Clear Transmit Underflow Status (CTUF), C3 Bit 3

When a "1" is written into the CTUF status bit, the CTUF bit and its associated interrupt are reset. The CTUF command bit is self-clearing and writing a "0" into this bit is a nonfunctional operation.

- **Sync Code Register**

The Sync Code Register is an 8-bit register for storing the programmable sync code required for received data character synchronization in the one-sync-character and two-sync-character modes. The Sync Code Register also provides for stripping the sync/fill characters from the received data (a programmable option) as well as automatic insertion of fill characters in the transmitted data stream. The Sync Code Register is not utilized for teceiver character synchronization in the external sync mode; however, it provides storage of receiver match and transmit fill characters.

The Sync Code Register can be loaded when AC2 and AC1 are a "1" and "0" respectively, and R/$\overline{W}$ = "Low" and RS = "High".

The Sync Code Register may be changed after the detection of a match with the received data (the first sync code having been detected) to synchronize with a double-word sync pattern. (This sync code change must occur prior to the completion of the second character.) The sync match (SM) output can be used to interrupt the MPU system to indicate that the first eight bits have matched. The service routine would then change the sync match register to the second half of the pattern. Alternately, the one-sync-character mode can be used for sync codes for 16 or more bits by using software to check the second and subsequent bytes after reading them from the FIFO.

The detection of the sync code can be programmed to appear on the Sync Match/$\overline{DTR}$ output by writing a "1" in PC1 (C2 bit 0) and a "0" in PC2 (C2 bit 1). The Sync Match output will go "High" for one bit time beginning at the character interface between the sync code and the next character.

- **Parity for Sync Character**

Transmitter

Transmitter does not generate parity for the sync character except 9-bit mode.

9-bit (8-bit + parity) – 8-bit sync character + parity
8-bit (7-bit + parity) – 8-bit sync character (no parity)
7-bit (6-bit + parity) – 7-bit sync character (no parity)

Receiver

At Synchronization

Receiver automatically strips the sync character(s) (two sync characters if '2 sync' mode is selected) which is used to establish synchronization. And parity is not checked for these sync characters.

After Synchronization is Established

When 'strip sync' bit is selected, the sync characters (fill characters) are stripped and parity is not checked for the stripped sync (fill) characters. When 'strip sync' bit is not selected (0), the sync character is assumed to be normal data and it is transferred into FIFO after parity checking. (When non-parity format is selected, parity is not checked.)

Strip Sync (C1 Bit 2)	Data Format (C2 Bit 3-5)	Operation
1	x	No transfer of sync code. No parity check of sync code.
0	With Parity	*Transfer data and sync codes. Parity check.
0	Without Parity	*Transfer data and sync codes. No parity check.

* Subsequent to synchronization
x ..... don't care

It is necessary to pay attention to the selected sync character in the following cases.

1) Data format is (6 + parity), (7 + parity),
2) Strip sync is not selected ("0").
3) After synchronization when sync code is used as a fill character.

Transmitter sends sync character without parity, but receiver checks the parity as if it is normal data. Therefore, the sync character should be chosen to match the parity check selected for the receiver in this special case.

- **Receive Data First-In First-Out Register (Rx Data FIFO)**

The Receive Data FIFO Register consists of three 8-bit registers which are used for buffer storage of received data. Each 8-bit register has an internal status bit which monitors its full or empty condition. Data is always transferred from a full register to an adjacent empty register. The transfer from register to register occurs on E pulses. The RDA status bit will be "1" when data is available in the last location of the Rx Data FIFO.

In an Overrun condition, the overrunning character will be transferred into the full first stage of the FIFO register and will cause the loss of that data character. Successive overruns continue to overwite the first register of the FIFO. This destruction of data is indicated by means of the Overrun status bit. The Overrun bit will be set when the overrun occurs and remains set until the Status Register is read, followed by a read of the Rx Data FIFO.

Unused data bits for short word lengths (including the parity bit) will appear as "0"s on the data bus when the Rx Data FIFO is read.

- **Transmit Data First-In First-Out Register (Tx Data FIFO)**

The Transmit Data FIFO Register consists of three 8-bit registers which are used for buffer storage of data to be transmitted. Each 8-bit register has an internal status bit which monitors its full or empty condition. Data is always transferred from a full register to an adjacent empty register. The transfer is clocked by E pulses.

The TDRA status bit will be "High" if the Tx Data FIFO is available for data.

Unused data bits for short word lengths will be handled as "don't cares". The parity bit is not transferred over the data bus since the SSDA generates parity at transmission.

When an Underflow occurs, the Underflow character will be either the contents of the Sync Code Register or an all "1"s character. The underflow will be stored in the Status Register until cleared and will appear on the Underflow output as a pulse approximately a Tx CLK "High" period wide.

- **Status Register**

The Status Register is an 8-bit read-only register which provides the real-time status of the SSDA and the associated serial data channel. Reading the Status Register is a non-destructive process. The method of clearing status bits depends upon the function each bit represents and is discussed for each bit in the register.

Receiver Data Available (RDA), S Bit 0

The Receiver Data Available status bit indicates when receiver data can be read from the Rx Data FIFO. The receiver data being present in the last register (#3) of the FIFO causes RDA to be "1" for the 1-byte transfer mode. The RDA bit being "1" indicates that the last two registers (#2 and #3) are full when in the 2-byte transfer mode. The second character can be read without a second status rad (to determine that the character is available). And E pulse must occur between reads of the Rx Data FIFO to allow the FIFO to shift. Status must be read on a word-by-word basis if receiver data error checking is important. The RDA status bit is reset automatically when data is not available.

Transmitter Data Register Available (TDRA), S Bit 1

The TDRA status bit indicates that data can be loaded into the Tx Data FIFO Register. The first register (#1) of the Tx Data FIFO being empty will be indicated by a "1" in the TDRA status bit in the 1-byte transfer mode. The first two registers (#1 and #2) must be empty for TDRA to be "1" when in the 2-byte transfer mode. The Tx Data FIFO can be loaded with two bytes without an intervening status read; however, one E pulse must occur between loads. TDRA is inhibited by the Tx Reset or $\overline{RES}$. When Tx Reset is set, the Tx Data FIFO is cleared and then released on the next E clock pulse. The Tx Data FIFO can then be loaded with up to three characters of data, even though TDRA is inhibited. This feature allows preloading data prior to the release of Tx Reset. A "High" level on the $\overline{CTS}$ input inhibits the TDRA status bit in either sync mode of operation (one-sync-character or two-sync-character). $\overline{CTS}$ does not affect TDRA in the external sync mode. This

enables the SSDA to operate under the control of the $\overline{CTS}$ input with TDRA indicating the status of the Tx Data FIFO. The $\overline{CTS}$ input does not clear the Tx Data FIFO in any operating mode.

Data Carrier Detect ($\overline{DCD}$), S Bit 2

A positive transition on the $\overline{DCD}$ input is stored in the SSDA until cleared by reading both Status and Rx Data FIFO. A "1" written into Rx Rs also clears the stored $\overline{DCD}$ status. The $\overline{DCD}$ status bit, when set, indicates that the $\overline{DCD}$ input has gone "High", The reading of both Status and Receive Data FIFO allows Bit 2 of subsequent Status reads to indicate the state of the $\overline{DCD}$ input until the next positive transition.

Clear-to-Send ($\overline{CTS}$), S Bit 3

A positive transiton on the $\overline{CTS}$ input is stored in the SSDA until cleared by writing a "1" into the Clear $\overline{CTS}$ control bit or the Tx Rs bit. The $\overline{CTS}$ status bit, when set, indicates that the $\overline{CTS}$ input has gone "High". The Clear $\overline{CTS}$ command (a "1" into C3 Bit 2) allows Bit 3 of subsequent Status reads to indicate the state of the $\overline{CTS}$ input until the next positive transition.

Transmitter Underflow (TUF), S Bit 4

When data is not available for the transmitter, an underflow occurs and is so indicated in the Status Register (in the Tx Sync on underflow mode). The underflow status bit is cleared by writing a "1" into the Clear Underflow (CTUF) control bit or the Tx Rs bit. TUF indicates that a sync character will be transmitted as the next character. A TUF is indicated on the output only when the contents of the Sync Code Register is to be transferred (transmit sync code on underflow = "1").

Receiver Overrun (Rx Ovrn), S Bit 5

Overrun indicates data has been received when the Rx Data FIFO is full, resulting in data loss. The Rx Ovrn status bit is set when Overrun occurs. The Rx Ovrn status bit is cleared by reading Status followed by reading the Rx Data FIFO or by setting the Rx Rs control bit.

Receiver Parity Error (PE), S Bit 6

The parity error status bit indicates that parity for the character in the last register of the Rx Data FIFO did not agree with selected parity. The parity error is cleared when the character to which it pertains is read from the Rx Data FIFO or when Rx Rs occurs. The $\overline{DCD}$ input does not clear the Parity Error or Rx Data FIFO status bits.

Interrupt Request ($\overline{IRQ}$), S Bit 7

The Interrupt Request status bit indicates when the $\overline{IRQ}$ output is in the active state ($\overline{IRQ}$ output = "Low"). The IRQ status bit is subject to the same interrupt enables (RIE, TIE, and EIE) as the $\overline{IRQ}$ output. The IRQ status but simplifies status inquiries for polling systems by providing single bit indication of service requests.

Table 1 SSDA Programming Model

Register	Control* Inputs		Address Control		Register Content							
	RS	R/$\overline{W}$	AC2	AC1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Status (S)	0	1	X	X	Interrupt Request ($\overline{IRQ}$)	Receiver Parity Error (PE)	Receiver Overrun (Rx Ovrn)	Transmitter Underflow (TUF)	Clear-to-Send ($\overline{CTS}$)	Data Carrier Detect ($\overline{DCD}$)	Transmitter Data Register Available (TDRA)	Receiver Data Available (RDA)
Control 1 (C1)	0	0	X	X	Address Control 2 (AC2)	Address Control 1 (AC1)	Receiver Interrupt Enable (RIE)	Transmitter Interrupt Enable (TIE)	Clear Sync	Strip Sync Characters (Strip Sync)	Transmitter Reset (Tx Rs)	Receiver Reset (Rx Rs)
Receive Data FIFO ***	1	1	X	X	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
Control 2 (C2)	1	0	0	0	Error Interrupt Enable (EIE)	Transmit Sync Code on Underflow (Tx Sync)	Word Length Select 3 (WS3)	Word Length Select 2 (WS2)	Word Length Select 1 (WS1)	1-Byte/2-Byte Transfer (1-Byte/2-Byte)	Peripheral Control 2 (PC2)	Peripheral Control 1 (PC1)
Control 3 (C2)	1	0	0	1	Not Used	Not Used	Not Used	Not Used	Clear Transmitter Underflow Status (CTUF)	Clear $\overline{CTS}$ Status (Clear $\overline{CTS}$)	One-Sync-Character/ Two-Sync Character Mode Control (1 Sync/ 2 Sync)	External/ Internal Sync Mode Control (E/I Sync)
Sync Code **	1	0	1	0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
Transmit Data FIFO ***	1	0	1	1	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0

* 0 ; "Low" level, 1 ; "High" level

** "FF" should not be used as Sync Code.

*** When the SSDA is used in applications requiring the MSB of data to be receive and transmitted first, the data bus inputs to the SSDA may be reversed (D_0 to D_7, etc.). Caution must be used when this is done since the bit positions in this table will be reversed, and the parity should not be selected.

Table 2 Functions of SSDA Register

Register	Bit	Symbol	Function			
Status Register (S)	7	IRQ	The IRQ flag is cleared when the source of the IRQ is cleared. The source is determined by the enables in the Control Registers: TIE, RIE, EIE.			
	6	PE	Conditions for Set	When parity error is detected in receive data.	Conditions for Reset	Read Rx Data FIFO, or a "1" into Rx Rs (C1 Bit 0).
	5	Rx Ovrn		When receive data FIFO overruns.		Read Status and then Rx Data FIFO, or a "1" into Rx Rs (C1 Bit 0).
	4	TUF		When under flow is occurred in the transmitter.		A "1" into CTUF (C3 Bit 3) or into Tx Rs (C1 Bit 1).
	3	$\overline{CTS}$		When $\overline{CTS}$ signal rises.		A "1" into Clear $\overline{CTS}$ (C3 Bit 2) or a "1" into Tx Rs (C1 Bit 1)
	2	$\overline{DCD}$		When $\overline{DCD}$ signal rises.		Read Status and then Rx Data FIFO or a "1" into Rx Rs (C1 Bit 0)
	1	TDRA		1 Byte Transfer Mode; when the transmit data FIFO (#1) is empty.		Write into Tx Data FIFO.
				2 Byte Transfer Mode; when the transmit data FIFO (#1, #2) is empty.		
	0	RDA		1 Byte Transfer Mode; when the data is received in the receive data FIFO (#3).		Read Rx Data FIFO.
				2 Byte Transfer Mode; when the data is received in the receive data FIFO (#2, #3).		
Control Register 1 (C1)	7 6	AC2 AC1	Used to access other registers, as shown Table 1.			
	5	RIE	When "1", enables interrupt on RDA (S Bit 0).			
	4	TIE	When "1", enables interrupt on TDRA (S Bit 1).			
	3	Clear Sync	When "1", clears receiver character synchronization.			
	2	Strip Sync	When "1", strips all sync codes from the received data stream.			
	1	Tx Rs	When "1", resets and inhibits the transmitter section.			
	0	Rx Rs	When "1", resets and inhibits the receiver section.			
Control Register 2 (C2)	7	EIE	When "1", enables the PE, Rx Ovrn, TUF, $\overline{CTS}$, and $\overline{DCD}$ interrupt flags (S Bits 6 through 2).			
	6	Tx Sync	When "1", allows sync code contents to be transferred on underflow, and enables the TUF Status bit and output. When "0", an all mark character is transmitted on underflow.			
	5 4 3	WS3 WS2 WS1	Word Length Select			
	2	1-Byte/2-Byte	When "1", enables the TDRA and RDA bits to indicate when a 1-byte transfer can occur; when "0", the TDRA and RDA bits indicate when a 2-byte transfer can occur.			
	1 0	PC2 PC2	SM/$\overline{DTR}$ Output Control			
Control Register 3 (C3)	3	CTUF	When "1", clears TUF (S Bit 4), and $\overline{IRQ}$ if enabled.			
	2	Clear $\overline{CTS}$	When "1", clears $\overline{CTS}$ (S Bit 3), and $\overline{IRQ}$ if enabled.			
	1	1-Sync/2-Sync	When "1", selects the one-sync-character mode; when "0", selects the two-sync-character mode.			
	0	E/I Sync	When "1", selects the external sync mode; when "0", selects the internal sync mode.			

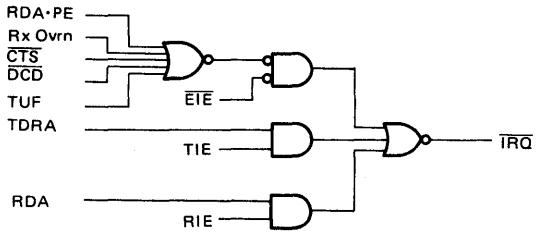
Table 3 Word Length

Bit 5 WS3	Bit 4 WS2	Bit 3 WS1	Word Length
0	0	0	6 Bits + Even Parity
0	0	1	6 Bits + Odd Parity
0	1	0	7 Bits
0	1	1	8 Bits
1	0	0	7 Bits + Even Parity
1	0	1	7 Bits + Odd Parity
1	1	0	8 Bits + Even Parity
1	1	1	8 Bits + Odd Parity

Table 4 SM/$\overline{DTR}$ Output Control

Bit 1 PC2	Bit 0 PC1	SM/$\overline{DTR}$ Output at Pin 5
0	0	"High" Level*
0	1	Pulse ⎍ 1-Bit Wide, on SM
1	0	"Low" Level*
1	1	SM Inhibited, "Low"*

* OUTPUT level is fixed by the data written into PC2, PC1.
** When "10" or "11", output is fixed at "Low".



■ INTERFACE SIGNALS FOR MPU

The SSDA interfaces to the HD6800 MPU with an 8-bit bi-directional data bus, a chip select line, a register select line, an interrupt request line, read/write line, an enable line, and a reset line. These signals, in conjunction with the HD6800 VMA output, permit the MPU to have complete control over the SSDA.

• Bi-Directional Data Bus (D_0~D_7)

The bi-directional data bus (D_0~D_7) allow for data transfer between the SSDA and the MPU. The data bus output drivers are three-state devices that remain in the high impedance (off) state except when the MPU performs an SSDA read operation.

• Enable (E)

The Enable signal, E, is a high impedance TTL compatible input that enables the bus input/output data buffers, clocks data to and from the SSDA, and moves data through the FIFO Registers. This signal is normally the continuous HMCS6800 System $\phi 2$ clock, so that incoming data characters are shifted through the FIFO.

• Read/Write (R/$\overline{W}$)

The Read/Write line is a high impedance input that is TTL compatible and is used to control the direction of data flow through the SSDA's input/output data bus interface. When Read/Write is "High" (MPU read cycle), SSDA output drivers are turned on if the chip is selected and a selected register is read. When it is "Low", the SSDA output drivers are turned off and the MPU writes into a selected register. The Read/Write signal is also used to select read-only or write-only registeres within the SSDA.

• Chip Select ($\overline{CS}$)

This high impedance TTL compatible input line is used to address the SSDA. The SSDA is selected when $\overline{CS}$ is "Low". VMA should be used in generating the $\overline{CS}$ input to insure that false selects will not occur. Transfers of data to and from the SSDA are then performed under the control of the Enable signal, Read/Write, and Register Select.

• Register Select (RS)

The Register Select line is a high impedance input that is TTL compatible. A "High" level is used to select Control Registers C2 and C3, the Sync Code Register, and the Transmit/Receive Data Registers. A "Low" level selects the Control 1 and Status Registers (see Table 1).

• Interrupt Request ($\overline{IRQ}$)

$\overline{IRQ}$ is a TTL compatible, open-drain (no internal pullup), active "Low" output that is used to interrupt the MPU. The $\overline{IRQ}$ remains "Low" until cleared by the MPU.

• Reset ($\overline{RES}$)

The $\overline{RES}$ input provides a means of resetting the SSDA from an external source. In the "Low" state, the $\overline{RES}$ input causes the following:

1) Receiver Reset (Rx Rs) and Transmitter Reset (Tx Rs) bits are set causing both the receiver and transmitter sections to be held in a reset condition.
2) Peripheral Control bits PC1 and PC2 are reset to zero, causing the SM/$\overline{DTR}$ output to be "High".
3) The Error Interrupt Enable (EIE) bit is reset.
4) An internal synchronization mode is selected.
5) The Transmitter Data Register Available (TDRA) status bit is cleared and inhibited.

When $\overline{RES}$ returns "High" (the inactive state), the transmitter and receiver sections will remain in the reset state until the Receiver Reset and Transmitter Reset bits are cleared via the bus under software control. The control Register bits affected by $\overline{RES}$ (Rx Rs, Tx Rs, PC1, PC2, EIE, and E/I Sync) cannot be changed when $\overline{RES}$ is "Low".

■ CLOCK INPUTS

Separate high impedance TTL compatible inputs are provided for clocking of transmitted and received data.

• Transmit Clock (Tx CLK)

The Transmit Clock input is used for the clocking of transmitted data. The transmitter shifts data on the negative transition of the clock.

• Receive Clock (Rx CLK)

The Receive Clock input is used for clocking in received data. The clock and data must be synchronized externally. The receiver samples the data on the positive transition of the clock.

■ SERIAL INPUT/OUTPUT LINES

• Receive Data (Rx Data)

The Receive Data line is a high impedance TTL compatible input through which data is received in a serial format. Data rates are from 0 to 600 kbps.

• Transmit Data (Tx Data)

The Transmit Data output line transfers serial data to a modem or other peripheral. Data rates are from 0 to 600 kbps.

■ PERIPHERAL/MODEM CONTROL

The SSDA includes several functions that permit limited control of a peripheral or modem. The functions included are $\overline{CTS}$, SM/$\overline{DTR}$, $\overline{DCD}$, and TUF.

• Clear-to-Send ($\overline{CTS}$)

The $\overline{CTS}$ input provides a real-time inhibit to the transmitter

section (the Tx Data FIFO is not disturbed). A positive $\overline{CTS}$ transition resets the Tx Shift Register and inhibits the TDRA status bit and its associated interrupt in both the one-sync-character and two-sync-character modes of operation. TDRA is not affected by the $\overline{CTS}$ input in the external sync mode.

The positive transition of $\overline{CTS}$ is stored within the SSDA to insure that its occurrence will be acknowledged by the system. The stored $\overline{CTS}$ information and its associated $\overline{IRQ}$ (if enabled) are cleared by writing a "1" in the Clear $\overline{CTS}$ bit. The $\overline{CTS}$ status bit subsequently follows the $\overline{CTS}$ input when it goes "Low".

The $\overline{CTS}$ input provides character timing for transmitter data when in the external sync mode. Transmission is initiated on the negative transition of the first full positive clock pulse of the transmitter clock (Tx CLK) after the release of $\overline{CTS}$ (see Figure 6).

- **Data Carrier Detect ($\overline{DCD}$)**

The $\overline{DCD}$ input provides a real-time inhibit to the receiver section (the Rx FIFO is not disturbed). A positive $\overline{DCD}$ transition resets and inhibits the receiver section except for the Receive FIFO and the RDRA status bit and its associated $\overline{IRQ}$.

The positive transition of $\overline{DCD}$ is stored within the SSDA to insure that its occurrence will be acknowledged by the system. The stored $\overline{DCD}$ information and its associated $\overline{IRQ}$ (if enabled) are cleared by reading the Status Register and then the Receiver FIFO, or by writing a "1" into the Receiver Reset bit. The $\overline{DCD}$ status bit subsequently follows the $\overline{DCD}$ input when it goes "Low". The $\overline{DCD}$ input provides character synchronization timing for the receiver during the external sync mode of operation. The receiver will be initialized and data will be sampled on the positive transition of the first full Receive Clock cycle after release of $\overline{DCD}$ (see Figure 7).

- **Sync Mach/Data Terminal Ready (SM/$\overline{DTR}$)**

The SM/$\overline{DTR}$ output provides four functions (see Table 4) depending on the state of the PC1 and PC2 control bits. When the Sync Match mode is selected (PC1 = "1", PC2 = "0"), the output provides a one-bit-wide pulse when a sync code is detected. This pulse occurs for each sync code match even if the receiver has already attained synchronization. The SM output is inhibited when PC2 = "1". The $\overline{DTR}$ mode (PC1 = "0") provides an output level corresponding to the complement of PC2 ($\overline{DTR}$ = "0" when PC2 = "1".) (see Table 4.)

- **Transmitter Underflow (TUF)**

The Underflow output indicates the occurrence of a transfer of a "fill character" to the Transmitter Shift Register when the last location (#3) in the Transmit Data FIFO is empty. The Underflow output pulse is approximately a Tx CLK "High" period wide and occurs during the last half of the last bit of the character preceding the "Underflow" (see Figure 4). The Underflow output pulse does not occur when the Tx Sync bit is in the reset state.

- **NOTE FOR USAGE**

If the hold time of $\overline{CS}$ signal and R/$\overline{W}$ signal is within 50~230 ns, there is a case that Transmit Data FIFO is not cleared and TDRA flag is not set when software reset using TxRS (TxRS=1) is executed. Usual program for data transmission will start to send the data as shown in Fig. 11 and Fig. 12.

In this case, the data of the first three bytes are not preset and unexpected data which is remaining in Transmit Data FIFO are sent in the first two bytes.

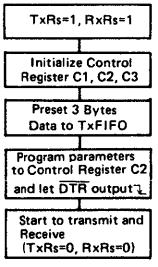


Figure 11 Normal Flow of Starting the Transmission and Reception

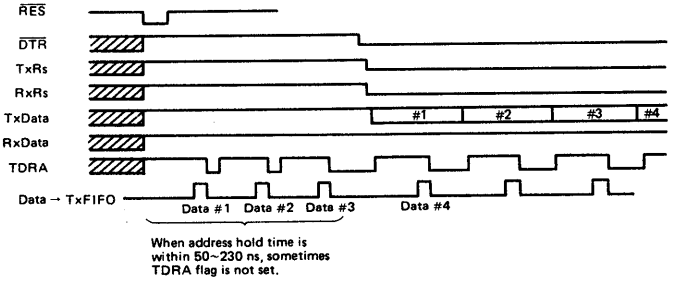


Figure 12 Transmission Start Sequence

In case of SSDA, Address Hold Time should be from 20 to 50 ns or over 230 ns.

- **$\overline{DCD}$ Input in External Synchronization Mode**

In case of receiving data in External Synchronization Mode, Receive data is put off by one bit at times, when $\overline{DCD}$ is drived like ⎍ in RxCLK cycle in which RDA flag is set.

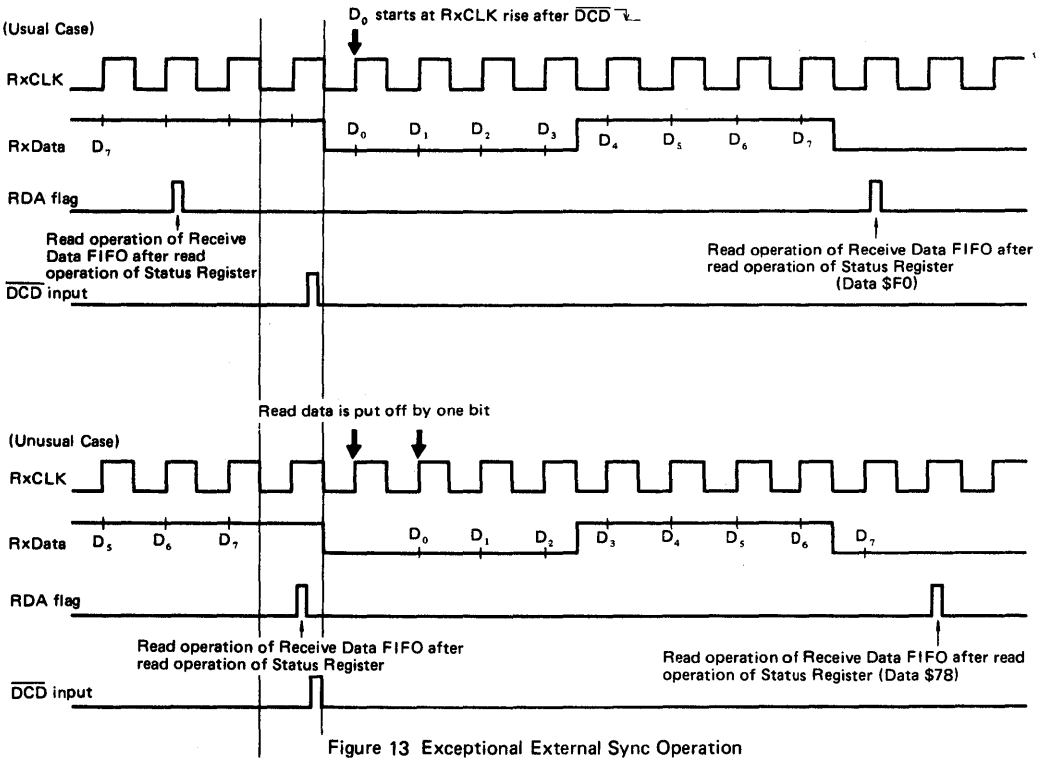


Figure 13 Exceptional External Sync Operation

To avoid this case, use SSDA in the following method.

(1) $\overline{DCD}$ ↓ and RxCLK ↓ should meet the relation shown in Fig. 14.

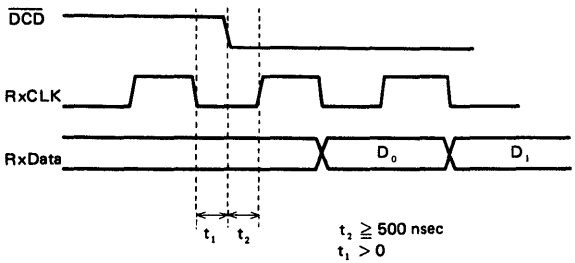


Figure 14 $\overline{DCD}$ Input Timing in External Sync Mode

(2) RxData should be input regarding the second RxCLK rise as D_0 bit, after $\overline{DCD}$ ↓.

HD46508, HD46508-1, HD46508A, HD46508A-1

ADU (Analog Data Acquisition Unit)

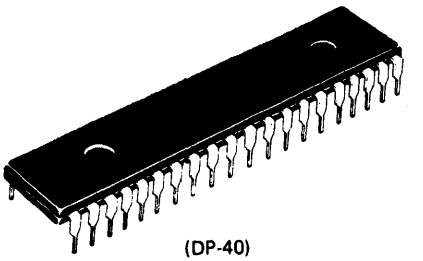
The HD46508 is a monolithic NMOS device with a 10-bit analog-to-digital converter, a programmable voltage comparator, a 16-channel analog multiplexer and HMCS6800 microprocessor family compatible interface.

Each of 16 analog inputs is either converted to a digital data by the analog-to-digital converter or compared with the specified value by the programmable comparator. The analog-to-digital converter uses successive approximation method as the conversion technique. It's intrinsic resolution is 10 bits but it can be 8 bits if the programmer so desires. The programmable voltage comparator compares the input voltage with the value specified by the programmer. The result (greater than, or smaller than) is reflected to the flag in the status register.

The device can expand its capability by controlling the external circuits such as sample holder, pre-amplifier and external multiplexer.

With these features, this device is ideally suited to applications such as process control, machine control and vehicle control.

HD46508P, HD46508P-1, HD46508PA, HD46508PA-1

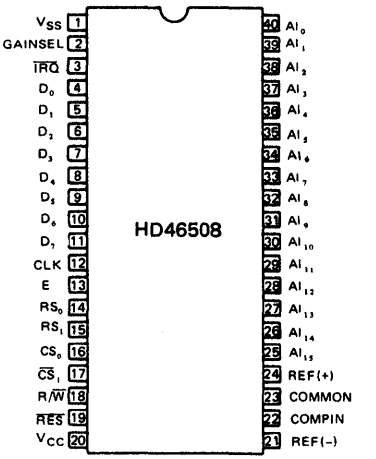


(DP-40)

■ FEATURES

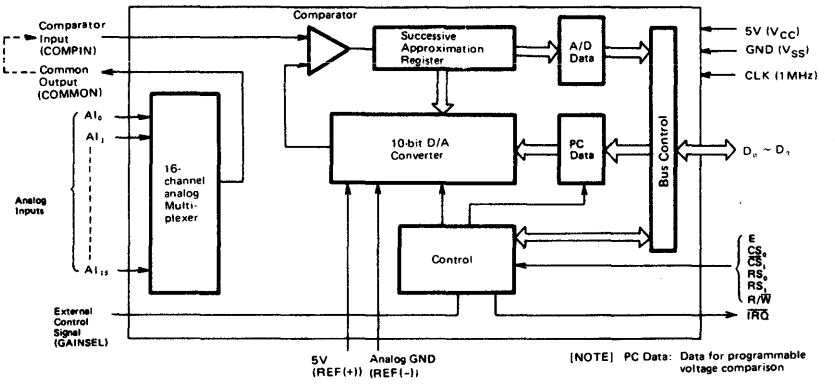
- 16-channel Analog multiplexer
- Programmable A/D Converter resolution (10-bit or 8-bit)
- Programmable Voltage comparison (PC)
- Conversion Time 100μs (A/D), 13μs(PC)
- External Sample and Hold Circuit Control
- Auto Range-switching Control of External Amplifier
- Waiting Function for the Settling Time of External Amplifier
- Interrupt Control (Only for A/D conversion)
- Single +5V Power Supply
- Compatible with HMCS6800 Bus (The connection with other Asynchronous Buses possible)

■ PIN ARRANGEMENT



(Top View)

■ BLOCK DIAGRAM



[NOTE] PC Data: Data for programmable voltage comparison

■ ORDERING INFORMATION

ADU	Bus Timing	Non Linearity*
HD46508A	1 MHz	±1 LSB
HD46508A-1	1.5 MHz	
HD46508	1 MHz	±3 LSB
HD46508-1	1.5 MHz	

* Specification for 10 bit A/D conversion

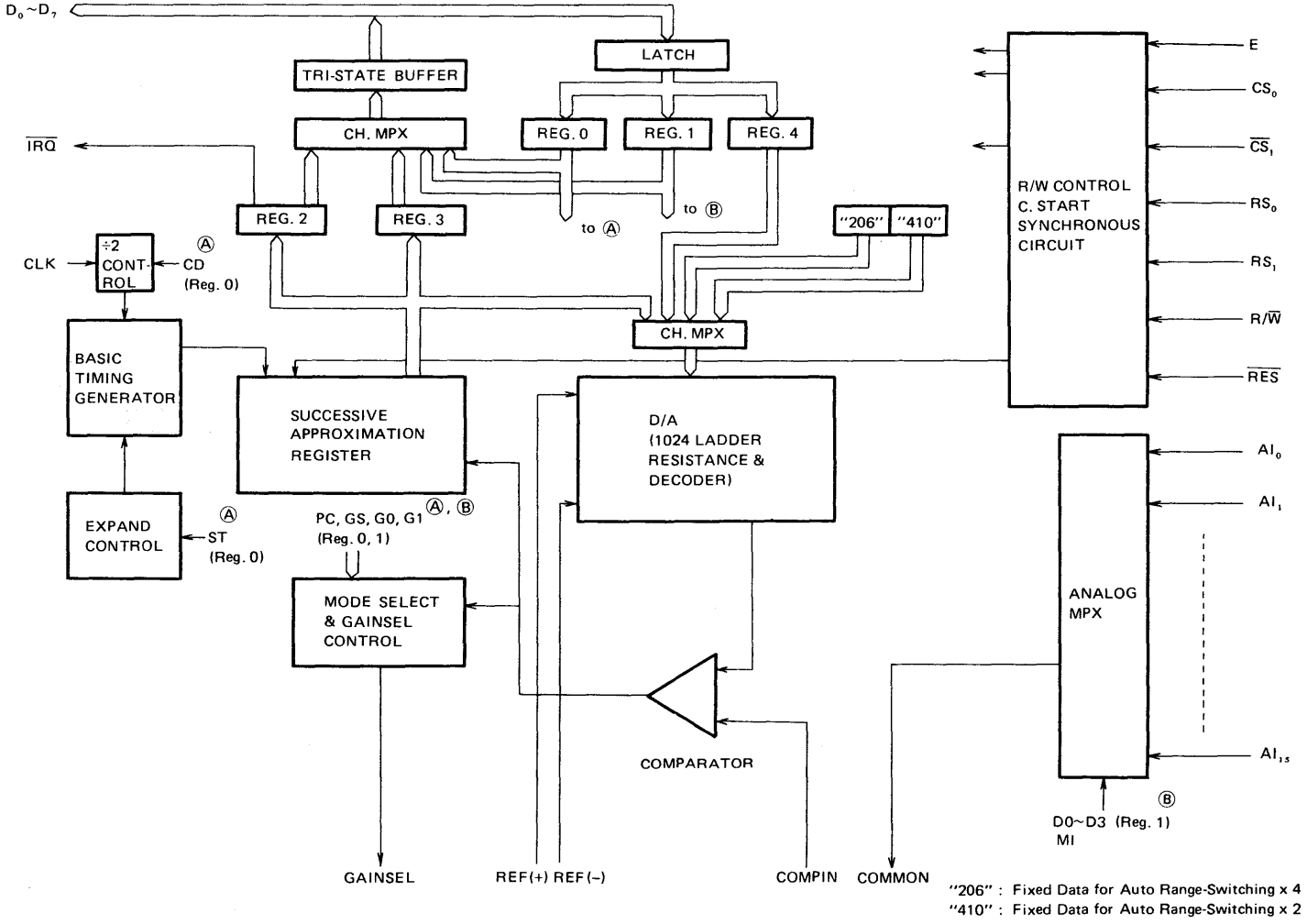


Figure 1 Internal Block Diagram

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC}*	−0.3 ~ +7.0	V
Input Voltage	V_{in}*	−0.3 ~ +7.0	V
Analog Input Voltage	V_{Ain}*	−0.3 ~ +7.0	V
Operating Temperature	T_{opr}	− 20 ~ + 75	°C
Storage Temperature	T_{stg}	− 55 ~ +150	°C

* With respect to V_{SS} (SYSTEM GND)

[NOTE] Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ RECOMMENDED OPERATING CONDITIONS

Item	Symbol	min	typ	max	Unit
Supply Voltage	V_{CC}*	4.75	5	5.25	V
Input "High" Voltage	V_{IH}*	2.0	–	V_{CC}	V
Input "Low" Voltage	V_{IL}*	−0.3	–	0.8	V
Analog Input Voltage	V_{Ain}*	0	–	5.0	V
Reference Voltage	$V_{REF(+)}$*	–	5.0	V_{CC}+0.25	V
	$V_{REF(-)}$*	−0.1	0	–	
Voltage Center of Ladder	$\frac{V_{REF(+)} + V_{REF(-)}}{2}$*	–	$\frac{V_{CC}}{2}$	$\frac{V_{CC}}{2}$+0.25	V
Operating Temperature	T_{opr}	− 20	25	75	°C

*With respect to V_{SS} (SYSTEM GND)

■ ELECTRICAL CHARACTERISTICS

• DC CHARACTERISTICS <1> (V_{CC} = 5V ± 5%, V_{SS} = 0V, Ta = −20 ~ +75°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Input "High" Voltage		V_{IH}		2.0	–	V_{CC}	V
Input "Low" Voltage		V_{IL}		−0.3	–	0.8	V
Output "High" Voltage	$D_0 \sim D_7$	V_{OH}	I_{OH} = −205μA	2.4	–	–	V
	GAINSEL		I_{OH} = −200μA	2.4	–	–	
			I_{OH} = −10μA	V_{CC}−1.0	–	–	
Output "Low" Voltage	$D_0 \sim D_7$, GAINSEL	V_{OL}	I_{OL} = 1.6 mA	–	–	0.4	V
	$\overline{IRQ}$		I_{OL} = 3.2 mA	–	–	0.4	
Input Leakage Current	E, CLK, R/$\overline{W}$ $\overline{RES}$, RS_0, RS_1 CS_0, $\overline{CS_1}$	I_{in}	V_{in} = 0 ~ 5.25V	-2.5	–	2.5	μA
Three-State (off state) Input Current	$D_0 \sim D_7$	I_{TSI}	V_{in} = 0.4 ~ 2.4V	-10	–	10	μA
Output Leakage Current	$\overline{IRQ}$	I_{LOH}	V_{OH} = 2.4V	–	–	10	μA
Power Dissipation		P_D		–	–	500	mW
Input Capacitance	$D_0 \sim D_7$	C_{in}	V_{in} = 0V, Ta = 25°C f = 1 MHz	–	–	12.5	pF
	E, CLK, R/$\overline{W}$ $\overline{RES}$, RS_0, RS_1 CS_0, $\overline{CS_1}$			–	–	10.0	pF
Output Capacitance	$\overline{IRQ}$, GAINSEL	C_{out}	V_{in} = 0V, Ta = 25°C f = 1 MHz	–	–	10.0	pF

- **DC CHARACTERISTICS <2>** (V_{CC} = 5V ± 5%, V_{SS} = 0V, Ta = -20~+75°C, unless otherwise noted.)

Item	Test Condition	min	typ	max	Unit
Analog Multiplexer ON Resistance	V_{Ain} = 5.0V, V_{CC} = 4.75V, Ta = 25°C	–	–	1	kΩ
OFF Channel Leakage Current	V_{Ain} = 5.0V, V_{CC} = 4.75V, Ta = 25°C, COMMON = 0V	–	10	100	nA
	V_{Ain} = 0V, Ta = 25°C, V_{CC} = 4.75V, COMMON = 5V	-100	-10	–	nA
Analog Multiplexer Input Capacitance		–	–	7.5	pF
Ladder Resistance (from REF(+) to REF(–))	$V_{REF\ (+)}$ = 5.0V, $V_{REF\ (-)}$ = 0V, Ta = 25°C	10	–	40	kΩ

- **CONVERTER SECTION** (Ta = 25°C, V_{CC} = $V_{REF(+)}$ = 5.0V, t_{cycC} = 1μs, unless otherwise noted.)

1. 10-BIT A/D CONVERSION

Item	HD46508A, HD46508A-1			HD46508, HD46508-1			Unit
	min	typ	max	min	typ	max	
Resolution	–	10	–	–	10	–	bits
Non-linearity Error *	–	±1/2	±1	–	±1	±3	LSB
Zero-Error	–	±1/2	±3/4	–	±1/2	±1	LSB
Full-Scall Error	–	±1/4	±1/2	–	±1/2	±1	LSB
Quantization Error	–	–	±1/2	–	–	±1/2	LSB
Absolute Accuracy *	–	±1	±3/2	–	±2	±4	LSB

2. 8-BIT A/D CONVERSION

Item	HD46508A, HD46508A-1			HD46508, HD46508-1			Unit
	min	typ	max	min	typ	max	
Resolution	–	8	–	–	8	–	bits
Non-linearity Error *	–	±1/8	±1/4	–	±1/4	±3/4	LSB
Zero-Error	–	±1/4	±3/8	–	±3/8	±1/2	LSB
Full-Scall Error	–	±1/4	±3/8	–	±3/8	±1/2	LSB
Quantization Error	–	–	±1/2	–	–	±1/2	LSB
Absolute Accuracy *	–	±5/8	±3/4	–	±3/4	±5/4	LSB

3. PROGRAMMABLE VOLTAGE COMPARISON (PC)

Item	HD46508A, HD46508A-1			HD46508, HD46508-1			Unit
	min	typ	max	min	typ	max	
Resolution	–	8	–	–	8	–	bits
Non-linearity Error *	–	±1/8	±1/4	–	±1/4	±3/4	LSB
Zero-Error	–	±1/4	±3/8	–	±3/8	±1/2	LSB
Full-Scall Error	–	±1/4	±3/8	–	±3/8	±1/2	LSB
Absolute Accuracy *	–	±3/8	±5/8	–	±1/2	±1	LSB

*Temperature Coefficient; 25 ppm of FSR/°C (max)

- **AC CHARACTERISTICS (V_{CC} = 5.0V ± 5%, V_{SS} = 0V, Ta = -20 ~ +75°C, unless otherwise noted.)**

1. CLOCK WAVEFORM

Item	Symbol	Test Conditions	CD* = 0			CD* = 1			Unit
			min	typ	max	min	typ	max	
CLK Cycle Time	t_{cycc}	Fig. 2	1.0	–	10	0.5	–	5	μs
CLK "High" Pulse Width	PW_{CH}		0.45	–	4.5	0.22	–	2.2	μs
CLK "Low" Pulse Width	PW_{CL}		0.40	–	4.0	0.21	–	2.1	μs
Rise and Fall Time of CLK	t_{Cr}, t_{Cf}		–	–	25	–	–	25	ns

* CD : CLK Divider bit

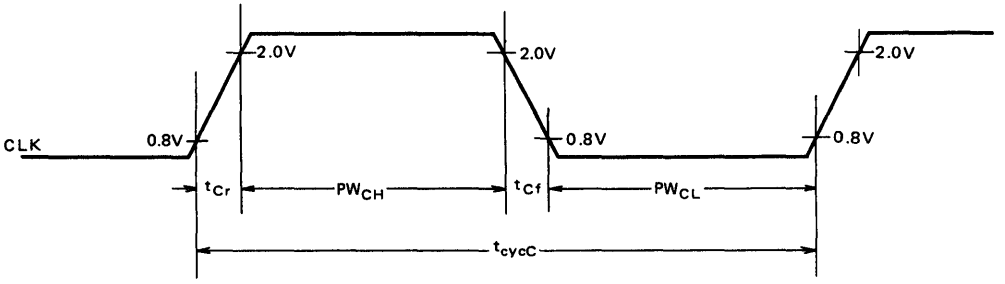


Figure 2 CLK Waveform

2. $\overline{IRQ}$, GAINSEL OUTPUT

Item	Symbol	Test condition	min	typ	max	Unit
$\overline{IRQ}$ Release Time	t_{IR}	Fig. 3	–	–	650	ns
GAINSEL Delay Time	t_{GSD1}	Fig. 4	–	–	750	ns
	t_{GSD2}		–	–	750	ns

t_{GSD1} : TTL Load
t_{GSD2} : CMOS Load

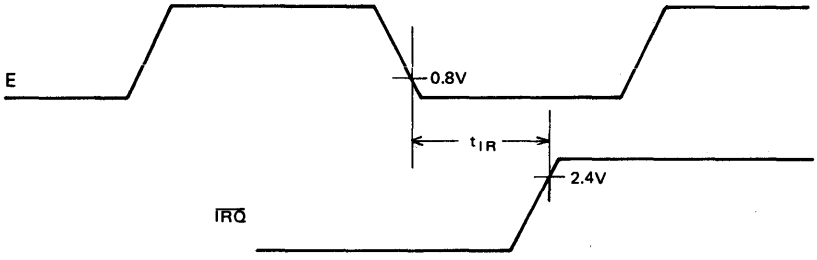
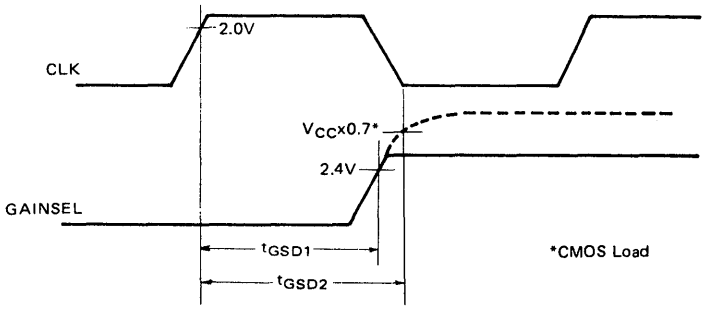


Figure 3 $\overline{IRQ}$ Release Time

(1) Sample & Hold



(2) x2, x4 Auto Range-Switching, Programmable Gain

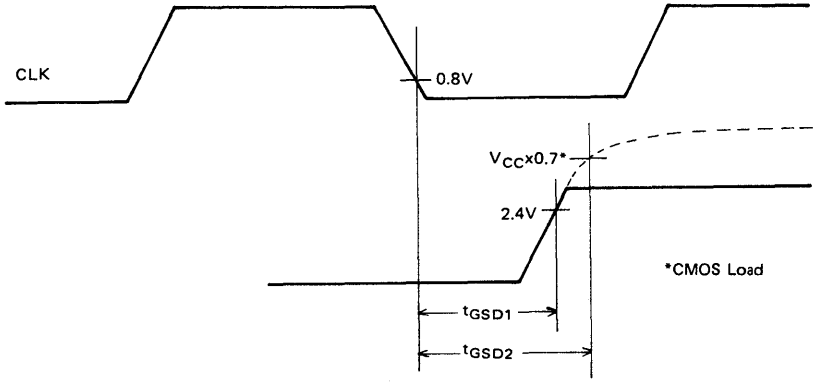


Figure 4 GAINSEL Delay Time

2. BUS TIMING CHARACTERISTICS

READ OPERATION SEQUENCE

Item	Symbol	Test Condition	HD46508 HD46508A			HD46508-1 HD46508A-1			Unit
			min	typ	max	min	typ	max	
Enable Cycle Time	t_{cycE}	Fig. 5	1.0	–	–	0.666	–	–	μs
Enable "High" Pulse Width	PW_{EH}		0.45	–	–	0.28	–	–	μs
Enable "Low" Pulse Width	PW_{EL}		0.40	–	–	0.28	–	–	μs
Rise and Fall Time of Enable	t_{Er},t_{Ef}		–	–	25	–	–	25	ns
Address Set Up Time	t_{AS}		140	–	–	140	–	–	ns
Data Delay Time	t_{DDR}		–	–	320	–	–	220	ns
Data Access Time	t_{ACC}		–	–	460	–	–	360	ns
Data Hold Time	t_H		10	–	–	10	–	–	ns
Address Hold Time	t_{AH}		10	–	–	10	–	–	ns

WRITE OPERATION SEQUENCE

Item	Symbol	Test Condition	HD46508 HD46508A			HD46508-1 HD46508A-1			Unit
			min	typ	max	min	typ	max	
Enable Cycle Time	t_{cycE}	Fig. 6	1.0	–	–	0.666	–	–	μs
Enable "High" Pulse Width	PW_{EH}		0.45	–	–	0.280	–	–	μs
Enable "Low" Pulse Width	PW_{EL}		0.40	–	–	0.280	–	–	μs
Rise and Fall Time of Enable	t_{Er}, t_{Ef}		–	–	25	–	–	25	ns
Address Set Up Time	t_{AS}		140	–	–	140	–	–	ns
Data Set Up Time	t_{DSW}		195	–	–	80	–	–	ns
Data Hold Time	t_H		10	–	–	10	–	–	ns
Address Hold Time	t_{AH}		10	–	–	10	–	–	ns

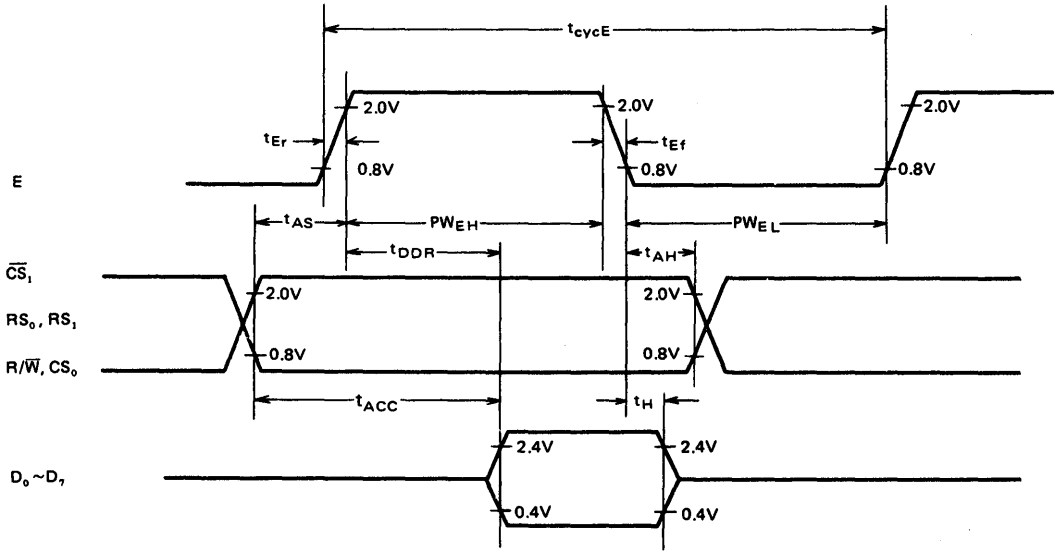


Figure 5 Read Timing

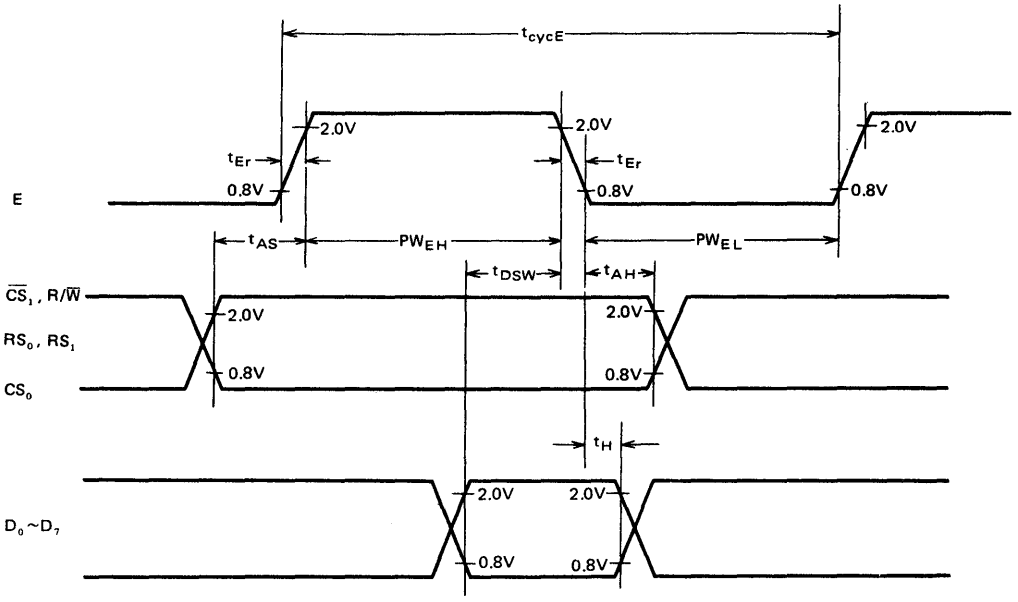


Figure 6 Write Timing

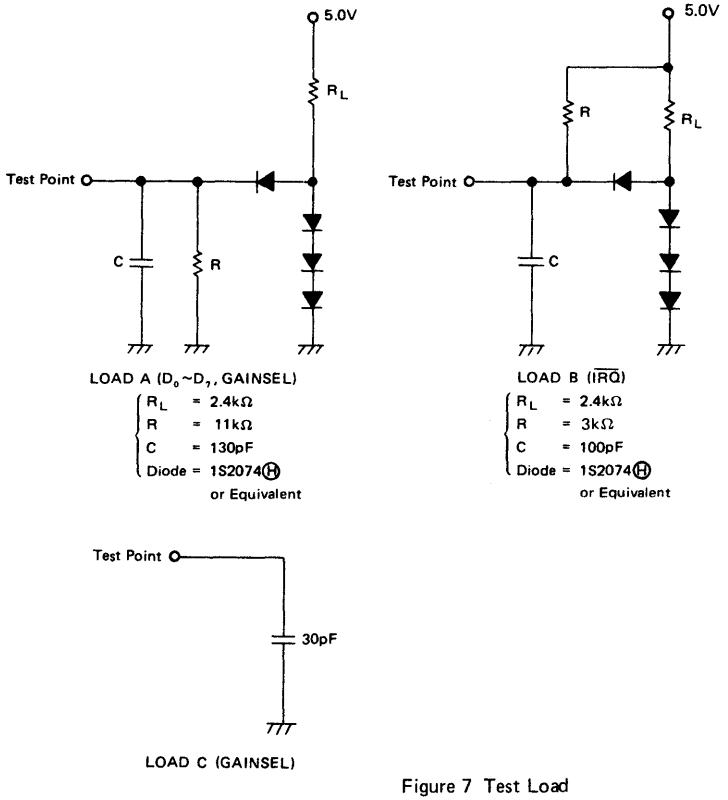


Figure 7 Test Load

■ SIGNAL DESCRIPTION

• Processor Interface

Data Bus (D_0~D_7)

The Bi-directional data lines (D_0~D_7) allow data transfer between the ADU and MPU. Data bus output drivers are three state buffers that remain in the high-impedance state except when MPU performs a ADU read operation.

Enable (E)

The Enable signal (E) is used as strobe signal in MPU R/W operation with the ADU internal registers. This signal is normally derived from the HMCS6800 system clock (ϕ_2).

Chip Select (CS_0, $\overline{CS_1}$)

The Chip Select lines (CS_0, $\overline{CS_1}$) are used to address the ADU. The ADU is selected when CS_0 is at "High" and $\overline{CS_1}$ is at "Low" level.

Read/Write (R/$\overline{W}$)

The R/$\overline{W}$ line controls the direction of data transfer between the ADU and MPU. When R/$\overline{W}$ is at "High" level, data of ADU is transferred to MPU. When R/$\overline{W}$ is at "Low" level, data of MPU is transferred to ADU.

Register Select (RS_0, RS_1)

The Register Select line (RS_0, RS_1) are used to select one of the 4 ADU internal registers. Table 1 shows the relation between (RS_0, RS_1) address and the selected register. The lowest 2 address lines of MPU are usually used for these signals.

Reset ($\overline{RES}$)

This input is used to reset the ADU. An input "Low" level on $\overline{RES}$ line forces the ADU into following status.

1) All the shift-registers in ADU are cleared and the conversion operation is stopped.
2) The GAINSEL output goes down to "Low" level. The $\overline{IRQ}$ output is made "Off" state and the D_0~D_7 are made high impedance state.

Interrupt Request ($\overline{IRQ}$) (Open Drain Output)

This output line is used to inform the A/D conversion end signal to the MPU. This signal becomes active "Low" level when IE bit in the control register 1 is "1" and IRQ bit in the control register 2 goes "1" at the end of conversion. And this signal returns to "High" right after The MPU reads the A/D Data Register (R3). Programmable voltage comparison does not affect this signal.

• Analog Data Interface

Analog Input (AI_0~AI_{15})

The Input Analog Data to be measured is applied to these Analog Input (AI_0~AI_{15}). These are multiplexed by internal 16 channel multiplexer and output to COMMOM pin. A particular input channel is selected when the multiplexer channel address is programmed into the control Register 1 (R1).

Multiplexer Common Output (COMMON)

This signal is the output of the 16 channel analog multiplexer, and may be connected to the input of pre-amplifier or sample/hold circuit according to user's purposes. When no external circuit needed, this output should be connected to the COMPIN input.

Comparator Input (COMPIN)

This is a high impedance input line that is used to transmit selected analog data to comparator. The COMMON line is usually connected to this input. When external Pre-amplifier or Sample/hold circuit is used, output of these circuits may be connected to this input.

Reference Voltage (+) (REF (+))

This line is used to apply the standard voltage to the internal ladder resistors.

Reference Voltage (–) (REF (–))

This line is connected to the analog ground.

• ADU Control

Conversion Clock (CLK)

The CLK is a standard clock input signals which defines internal timing for A/D conversion and PC operation.

Gain Select (GAINSEL) (CMOS Compatible Output)

This output is used to control the external circuit. The function of this signal is programmable and it is specified by (G1, G0) bits in Control Register 0. By using this output, user can control the auto-range-switching of external pre-amplifier, also control external sample & hold circuit, etc. as well.

[NOTE] This LSI is different from other HMCS6800 family LSIs in following function
- $\overline{RES}$ doesn't affect IE bit of R0

■ FUNCTION OF INTERNAL REGISTERS

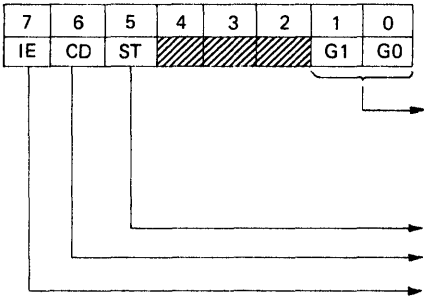
• Structure

Table 1 Internal Registers of the ADU

$\overline{CS_1}$	CS_0	RS_1	RS_0	Reg. #	Register Name	Read	Write	Data Bit 7	6	5	4	3	2	1	0
0	1	0	0	R0	Control Register 0	O	O	IE	CD	ST				G1	G0
0	1	0	1	R1	Control Register 1	O	O	SC	GS	PC	MI	D3	D2	D1	D0
0	1	1	0	R2	Status & A/D Data Register (H)	O	x	IRQ	BSY	PCO		OV	DW	C9	C8
0	1	1	1	R3	A/D Data Register (L)	O	x	C7	C6	C5	C4	C3	C2	C1	C0
0	1	1	1	R4	PC Data Register	x	O	B7	B6	B5	B4	B3	B2	B1	B0

(Note) O---YES
x---NO

Control Register 0 (R0)

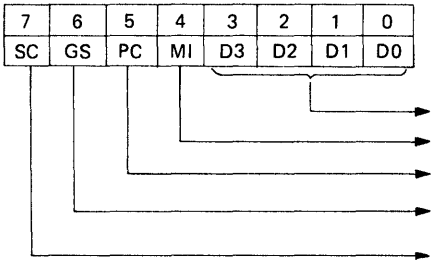


	"1"	"0"
Mode Select	See Table 2	
Not Used		
Not Used		
Not Used		
Settling Time	Available	Not Available
CLK Divider	CLK/2	CLK
Interrupt Enable*	Enable IRQ	Mask IRQ

Figure 8 Control Register 0

*$\overline{RES}$ doesn't affect IE bit.

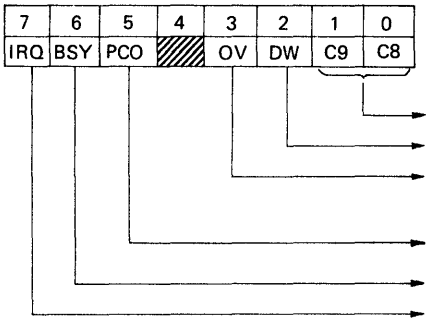
Control Register 1 (R1)



	"1"	"0"
MPX Channel Address	See Table 3	
MPX Inhibit	Inhibited	Not Inhibited
Prog. Comparator Select	Prog. Comparator mode	A/D Converter mode
GAINSEL Enable	GAINSEL Enable	GAINSEL Disable
Short-cycle Conversion	8-bit Length	10-bit Length

Figure 9 Control Register 1

Status & A/D Data Register (H)



	"1"	"0"
Upper bit (10 bit data)		
Data Weight	See Table 4.	
Data Over Scale flag	Data is over scale	Within the scale
Not Used		
Programmable Comparator Output	$V_{Ain} > V_p$	$V_{Ain} < V_p$
Busy flag	Under Conversion	Conversion Completed
IRQ flag	Requested	Not Requested

V_{Ain} : Unknown Input Voltage
V_p : Programmed Voltage by R4
C9, C8 bits are cleared when 8 bit A/D conversion is performed.

Figure 10 Status & A/D Data Register (H)

A/D Data Register (L)

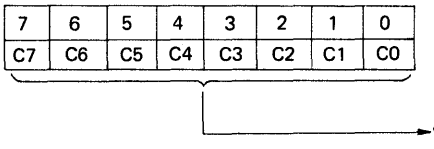


Figure 11 A/D Data Register (L)

PC Data Register

7	6	5	4	3	2	1	0
B7	B6	B5	B4	B3	B2	B1	B0

8 bit Data for Programmable Voltage Comparison

Figure 12 PC Data Register

- **Description for the Internal Registers**

Control Register 0 (R0)

This Register is a 5-bit read/write register that is used to specify Interrupt Enable (IE), CLK Divider (CD), Settling Time (ST) and Mode Select (G0, G1). This Register should be written before writing R1.

IE bit: (Interrupt Enable)
- IE = "1", Interrupt is requested through the $\overline{IRQ}$ output.
- IE = "0", Interrupt is masked.

CD bit: (Clock Divider)
- CD = "1", CLK ÷ 2 is used as internal clock.
- CD = "0", CLK is used directly.

ST bit: (Settling Time)
- ST = "1", First comparison is executed after 1 expanded cycle in order to compensate external amplifiers settling delay.
- ST = "0", Cycle is not delayed.

G0, G1 bit; (Mode select) These bits are used to specify the function of GAINSEL signal when GS bit is "1".

Table 2 Function of G0, G1

G1	G0	Mode Select
0	0	Sample & Hold
0	1	Auto Range-Switching x 2
1	0	Auto Range-Switching x 4
1	1	Programmable Gain Control

Control Register 1 (R1)

This register is an 8-bit read/write register that is used to store the command for A/D conversion mode and programmable comparison mode. This register includes MPX channel address (D_0~D_3), MPX inhibit (MI), programmable comparator select (PC), GAINSEL enable (GS) and short-cycle conversion (SC) bits. When this register (R1) is programmed, each conversion mode starts.

SC bit (Short-cycle)
- SC = "1", Short-cycle conversion (8 bit length)
- SC = "0", Normal conversion (10 bit length)

GS bit (GAINSEL Enable)
- GS = "1", GAINSEL signal is enabled. The function of GAINSEL is specified by (G0, G1) bits.
- GS = "0", GAINSEL signal is disabled. ("Low" level)

PC bit (Program comparator)
- PC = "1", Programmable voltage comparator mode
- PC = "0", A/D conversion mode

MI bit (MPX Inhibit)
- MI = "1", Internal MPX channel is inhibited in order to use external MPX channel.
- MI = "0", Internal MPX channel is used.

D0~D3 (MPX channel) These bits are used to select the particular MPX channel.

Table 3 MPX Channel Addressing

Channel #1	D3	D2	D1	D0	Analog Input
0	0	0	0	0	AI_0
1	0	0	0	1	AI_1
2	0	0	1	0	AI_2
3	0	0	1	1	AI_3
4	0	1	0	0	AI_4
5	0	1	0	1	AI_5
6	0	1	1	0	AI_6
7	0	1	1	1	AI_7
8	1	0	0	0	AI_8
9	1	0	0	1	AI_9
10	1	0	1	0	AI_{10}
11	1	0	1	1	AI_{11}
12	1	1	0	0	AI_{12}
13	1	1	0	1	AI_{13}
14	1	1	1	0	AI_{14}
15	1	1	1	1	AI_{15}

Table 4 Function Select

PC	SC	Function	GS	(G0, G1)
0	0	10 bit AD CONV.	0	DISABLE
			1	ENABLE*
	1	8 bit AD CONV.	0	DISABLE
			1	ENABLE*
1	x	PROG. COMP (8 bit)	x	DISABLE

x = Do not care
* = See Table 6
[NOTE] CD bit and ST bit are effective in every case.

Status & A/D Data Register (H) (R2)

This register is a 7-bit read only register that is used to store the upper 2-bit data (C8, C9), data weight (DW), data overscale (OV), programmable comparator output (PCO), busy (BSY) and interrupt request(IRQ).

(C8, C9) (Upper bit data) : These bits store upper 2-bit data measured by 10 bit length conversion.

DW bit (Data weight) : This bit indicates data weight when Auto range-switching mode is selected. This bit is set or reset when the conversion has completed. The conditions are shown in following Table.
In this mode GAINSEL output also goes "High" or "Low" on the same condition shown in Table 5.
Other status of DW bit is shown in Table 6.

OV bit (Over scale) : This bit is set when analog data is greater than or equal to reference Voltage ($V_{REF(+)}$).

PCO bit (Programmable comparator Output) : This bit indicates the result of programmable voltage comparison.
"1" → PCO $V_{Ain} > Vp$
"0" → PCO $V_{Ain} < Vp$
V_{Ain} : Analog Input Voltage to be compared
Vp : Programmed Voltage (R4)

BSY bit (Busy) : This bit indicates that the ADU is now under conversion.

IRQ bit (Interrupt Request) : This bit is set when the A/D conversion has completed and cleared by reading the R3.

A/D Data Register (L) (R3)

This register is an 8-bit read-only register that is used to store the lower 8 bits data of 10-bit conversion or full 8 bits data of the 8-bit conversion.

PC Data Register (R4)

This register is an 8-bit write-only register prepared for Programmable Voltage comparison. Stored data is converted to digital voltage, and compared with analog input to be measured. The result of comparison is set into PCO bit.

Table 5 Data Weight (DW) Set or Reset Condition

Mode \ Condition	Set ("1")	Reset ("0")
Auto Range-Switching (x2)	$V_{Ain} < \frac{410}{1024} \cdot V_{REF(+)}$	$V_{Ain} > \frac{410}{1024} \cdot V_{REF(+)}$
Auto Range-Switching (x4)	$V_{Ain} < \frac{206}{1024} \cdot V_{REF(+)}$	$V_{Ain} > \frac{206}{1024} \cdot V_{REF(+)}$

V_{Ain} : Analog Input Voltage to be measured
$V_{REF(+)}$: Voltage Applied to REF(+)

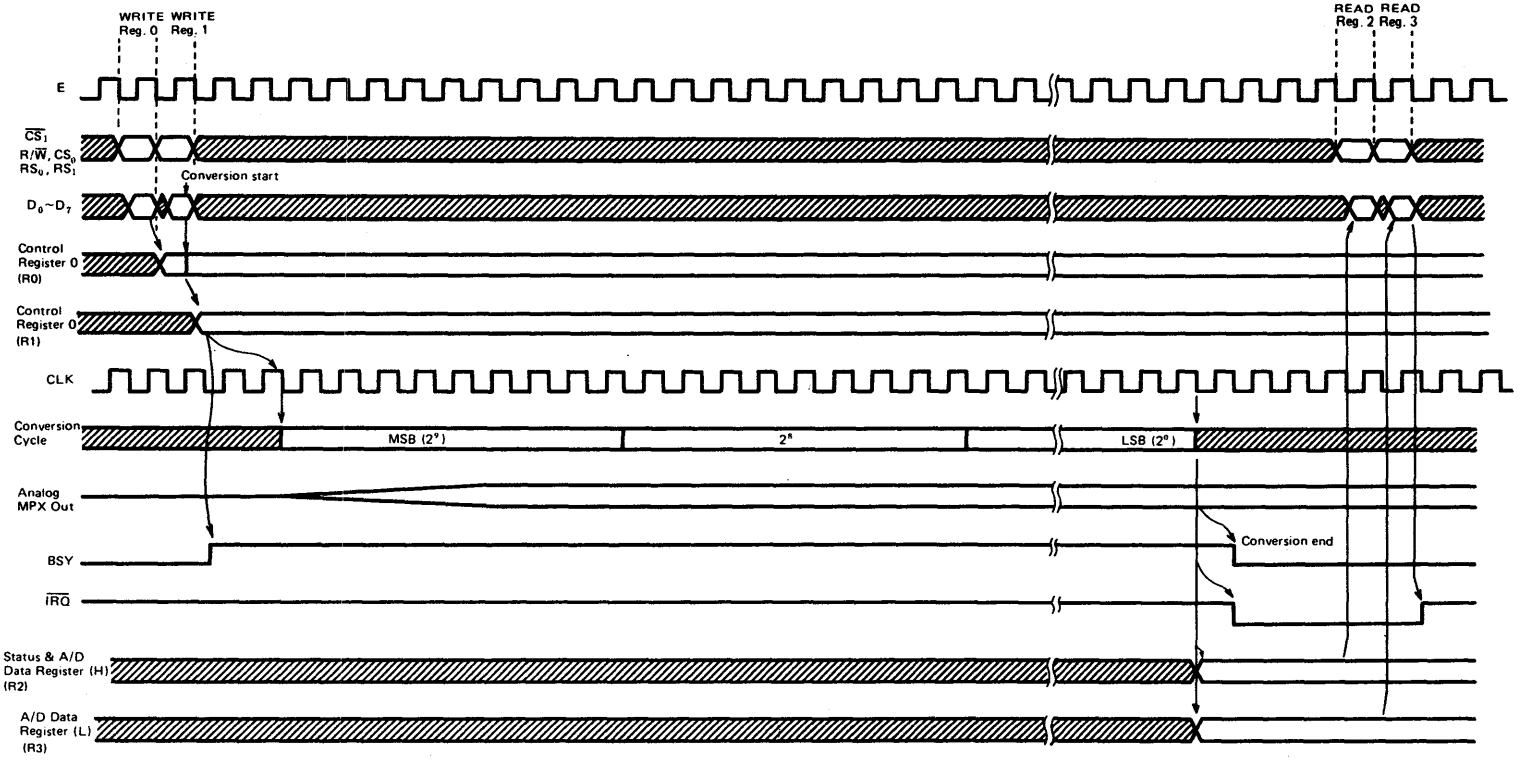
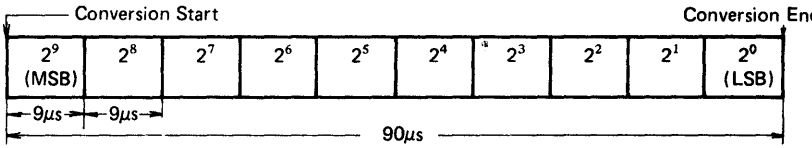


Figure 13 A/D Conversion Timing Chart (Basic Sequence)

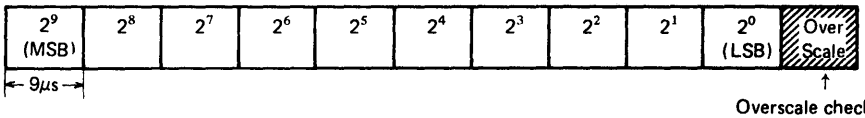
- **A/D Conversion and PC sequence** (t_{cyc}=1μs)

10 bits A/D Conversion

1) Basic Sequence (SC = "0", ST = "0", GS = "0")

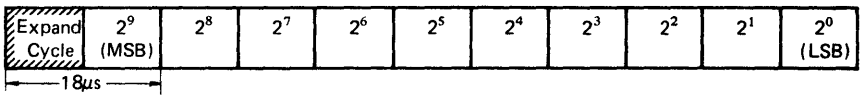


2) Basic Sequence (When overscale is detected)



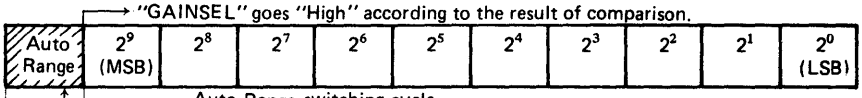
Overscale check Cycle (Analog Input is compared with $V_{REF(+)}$.)

3) Expanded Sequence (SC = "0", ST = "1", GS = "0")



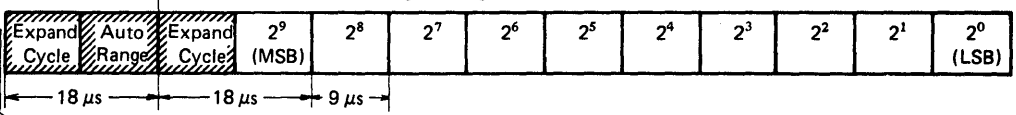
MSB cycle is expanded to compensate external amplifier's settling delay.

4) Auto Range-Switching Control Sequence (SC = "0", ST = "0", GS = "1", G0 = "0", G1 = "1" or G0 = "1", G1 = "0")

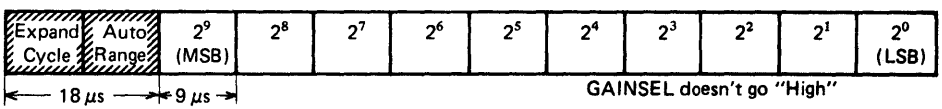


5) Auto Range-Switching & Expansion Control Sequence (SC = "0", ST = "1", GS = "1", G0 = "0", G1 = "1" or G0 = "1", G1 = "0")

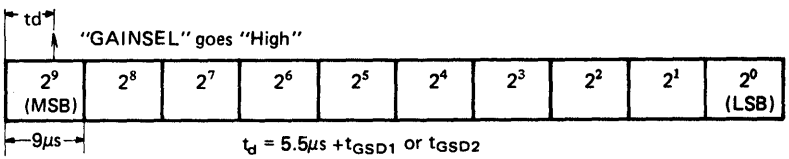
a) Analog Input < 1/2 $V_{REF(+)}$ or 1/4 $V_{REF(+)}$



b) Analog Input > 1/2$V_{REF(+)}$ or 1/4$V_{REF(+)}$



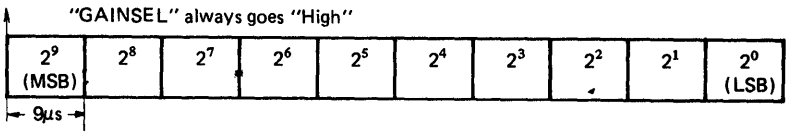
6) Sample & Hold Control Sequence (SC = "0", ST = "0", GS = "1", G0 = "0", G1 = "0")



t_d = 5.5μs + t_{GSD1} or t_{GSD2}

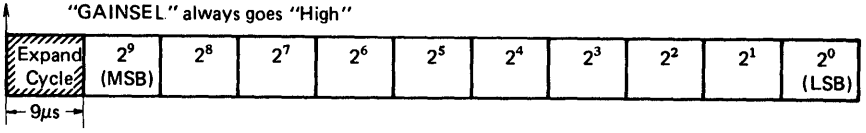
7) Programmable Gain Control Sequence

(SC = "0", ST = "0", GS = "1", G0 = "1", G1 = "1")



8) Programmable Gain & Expansion Control Sequence

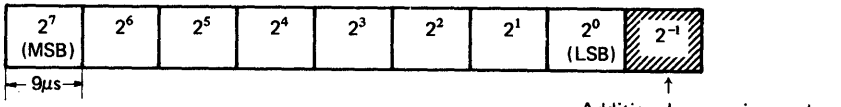
(SC = "0", ST = "1", GS = "1", G0 = "1", G1 = "1")



8 Bit A/D Conversion

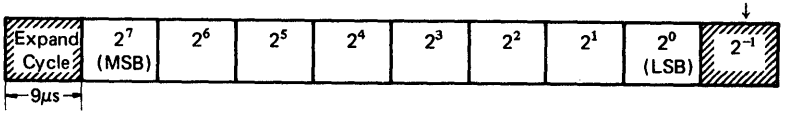
1) Basic Sequence

(SC = "1", ST = "0", GS = "0")



2) Expanded Sequence

(SC = "1", ST = "1", GS = "0")



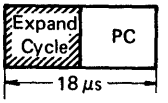
Programmable Voltage Comparison

1) Basic Sequence

(PC = "1", ST = "0")

2) Expanded Sequence

(PC = "1", ST = "1")



■ HOW TO USE THE ADU

• Functions of GAINSEL

The ADU is equipped with programmable GAINSEL output signal. By using GAINSEL output and external circuit, the ADU is able to implement following control.

1) Auto Range-Switching (Auto Gain) Control
2) Programmable Gain control
3) Sample & Hold control

GAINSEL output is controlled by Mode Select bit (G0, G1) when GAINSEL enable bit (GS) is "1".

Table 6 GAINSEL Control

GS	G1	G0	GAINSEL	Control Mode	DW
0	x	x	"Low"	Normal Use (GAINSEL is not used)	0
1	0	0	"High"	Sample & Hold control	0
1	0	1	*	Auto Range Switching x 2 control	**
1	1	0	*	Auto Range Switching x 4 control	**
1	1	1	"High"	Programmable Gain control	1

* GAINSEL goes "High" or "Low" according to the condition shown in Table 5.
** See, Table 5.

How to Control External Circuit

(1) Sample & Hold Control (G1=0, G0=0)

An example of Sample & Hold circuit is shown in Fig. 14. When ADU is set in Sample & Hold Control Mode, GAINSEL becomes "High" level on conversion and controls the data holding.

(2) Automatic Range Switching Control (G1=0, G0=1 or G1=1, G0=0)

The GAINSEL signal controls the external amplifier which can change the ratio of voltage amplification. (GAIN: 1 → 2 times or 1 → 4 times). Fig. 15 shows Automatic Range Switching Control. In this case, when the input voltage is lower than 206/1024 $V_{REF(+)}$, GAINSEL becomes "High" level. This makes the GAIN of the amplifier change from 1 to 4 times, and 4 times value of the input voltage is A/D converted. Using this function even if an input signal is small, it is possible to execute A/D conversion in nearly full scale. In this mode, when GAINSEL signal becomes "High", DW bit becomes "1" to show the range switching is in a progress.

(3) Programmable GAIN Control (G1=1, G0=1)

The GAINSEL signal is used for controlling the external amplifier of any GAIN which is fit to the system.

In this mode, GAINSEL always becomes "High" at the beginning of A/D conversion, so the change of range is controlled by GS bit. Converted data need to be corrected in software in accordance with GAIN of the amplifier.

This mode is effective in the case of converting very small input voltage.

(Note) Refer to "ADU Function Sequence" (A/D Conversion and PC Sequence) for the timing in which GAINSEL signal becomes "High". GAINSEL signal becomes "Low" in accordance with "1" → "0" change of BSY bit. Refer to Fig. 13.

x1 Sample & Hold

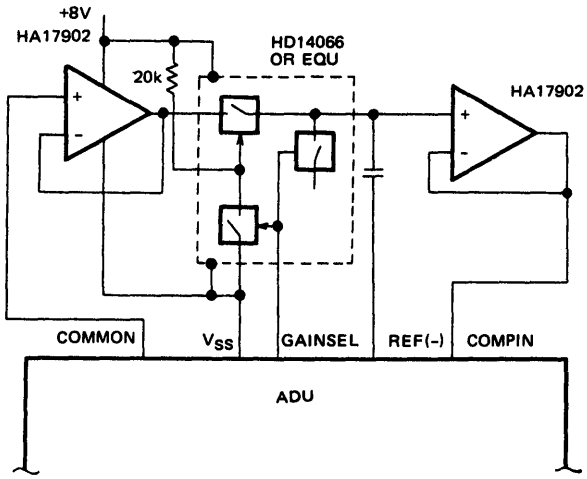


Figure 14 Sample & Hold Circuit

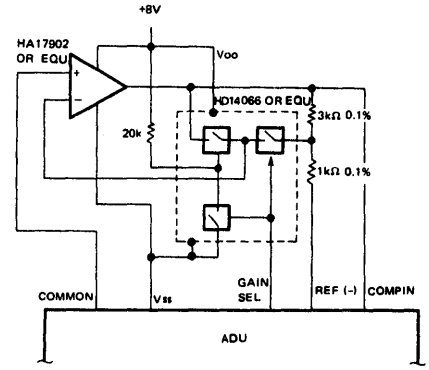
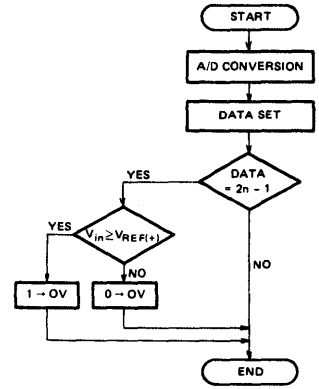


Figure 15 Pre-amplifier Circuit
(x1, x4 Auto-Range Switching)

- **Overscale Check**

ADU is equipped with hardware overscale detection function. The overscale detection is performed automatically when the result of A/D conversion is 2^n-1 (all bits = 1). When analog input V_{Ain} is higher than $V_{REF(+)}$, overscale bit (OV) is set to "1". The definition of the overscale is illustrated in Fig. 17. And the flow of overscale check is shown in Fig. 16.



OV	DATA	NOTE
0	11 1	NOT OVERSCALE
1	11 1	OVERSCALE

Figure 16 Overscale Check Flow

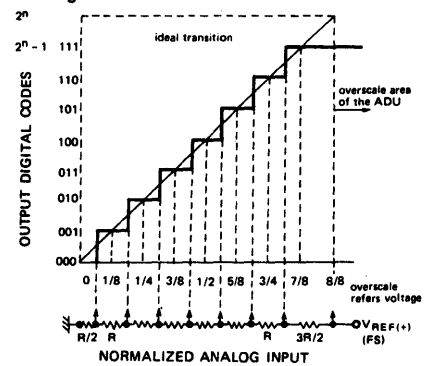


Figure 17 Definition ADU's Overscale

- **Usage of the PC**

The ADU has a programmable threshold voltage comparator (PC) function. The threshold voltage is pre-setable from 0V to 5V range with 8 bit resolution. The comparator's output is stored into PCO bit at the end of comarison.

The programmable voltage comparison time is so short that the interrupt is not requested at this mode. The end of comparison needs to be confirmed by reading the 1→0 transition of the BSY bit in R2.

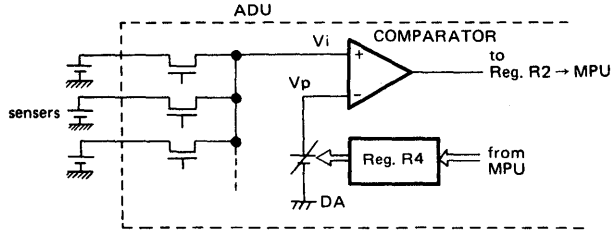
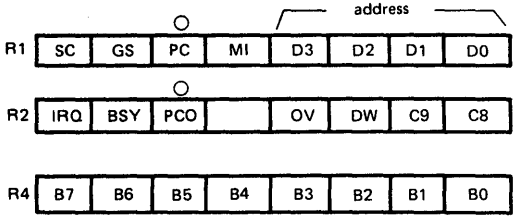


Figure 18 Function Diagram of the PC



PC=0 : A/D conversion mode
PC=1 : Programmable Voltage Comparison Mode

PCO : Programmable comparator output (1 bit data)

$B_0 \sim B_7$: V_p setting byte (upper byte of 10 bit D/A. Lower byte is set to 0)

Figure 19 Registers of the PC Mode

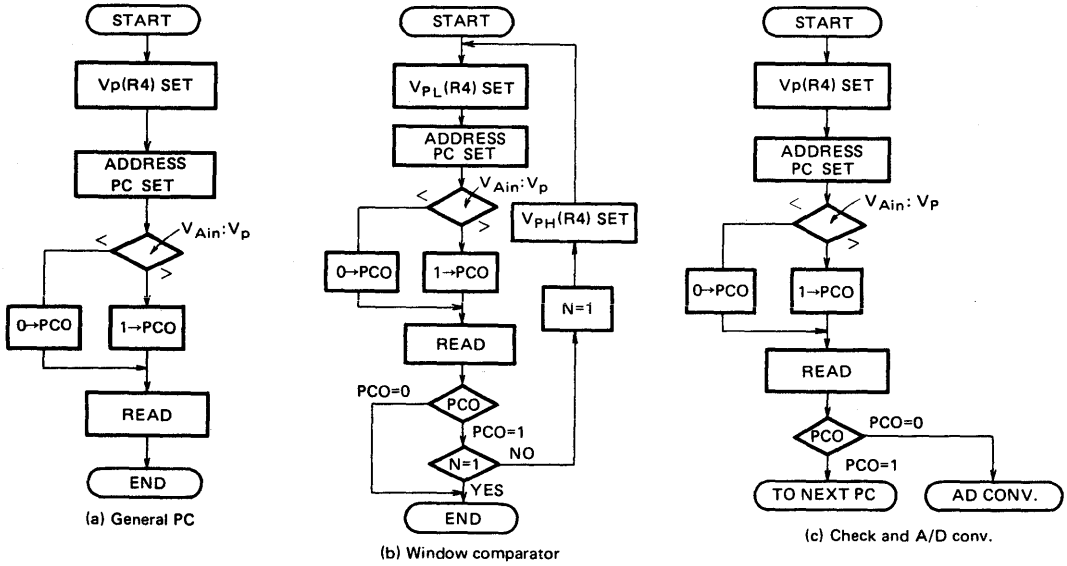
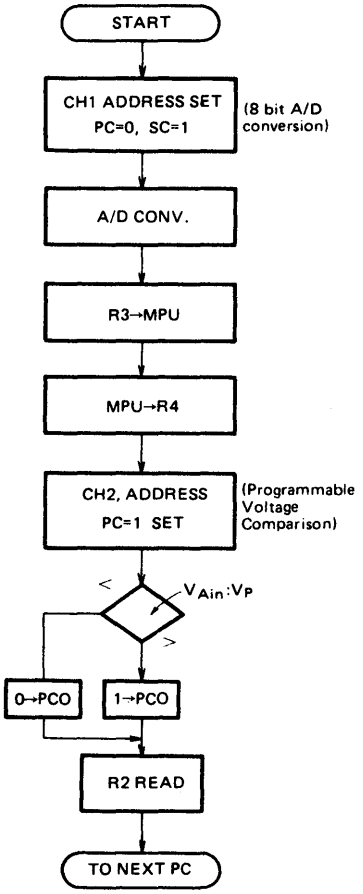


Figure 20 PC Application Flow Chart Examples



(d) Voltage Comparison between two channels.

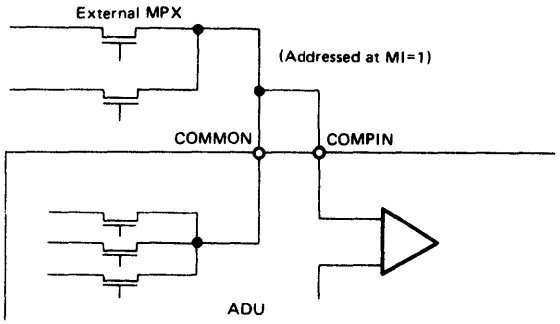
Figure 20 PC Application Flow Chart Examples (continued)

- **How to use MI bit**

MI bit (R1) functions as follows.

MI = 1: Internal MPX channel is inhibited in order to use attached external MPX channel.
MI = 0: Internal MPX channel is enabled.

MI bit used to select either of External MPX and Internal MPX. External MPX is connected as follows.



[NOTE] When external MPX is used as the way figure 20, 1 dammy AD conversion or PC at MI=1 should be performed.

Figure 21 How to use External MPX

■ EXAMPLE OF APPLIED CIRCUIT OF THE ADU

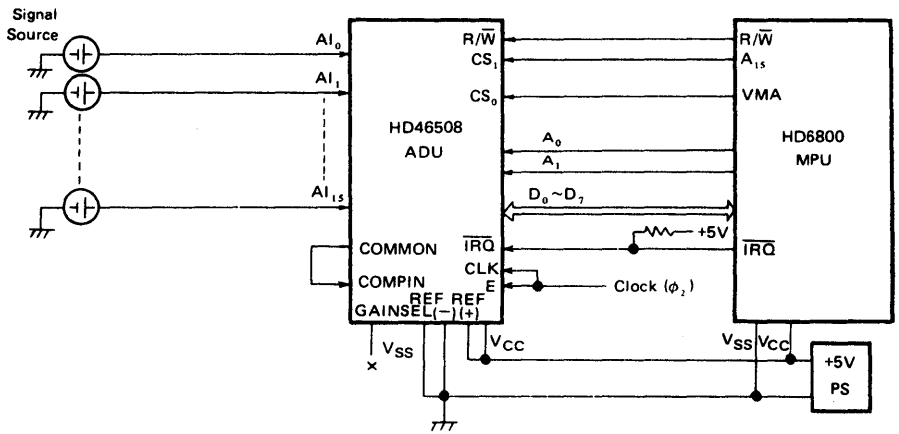


Figure 22 Single ADU System

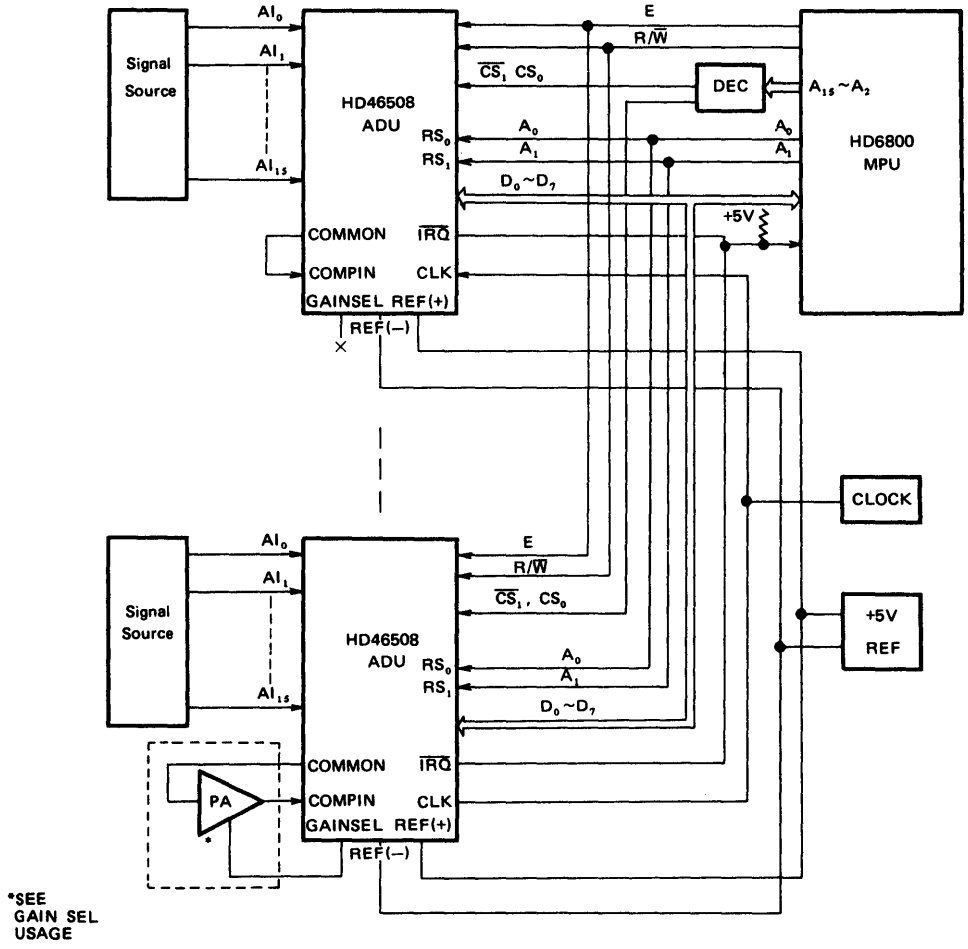


Figure 23 Multi ADU System

HD146818

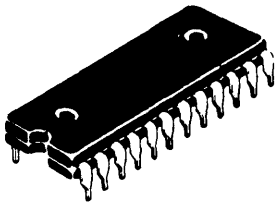
RTC (Real Time Clock Plus RAM) —PRELIMINARY—

The HD146818 is a HMCS6800 peripheral CMOS device which combines three unique features: a complete time-of-day clock with alarm and one hundred calendar, a programmable periodic interrupt and square-wave generator, and 50 bytes of Low-power static RAM.

This device includes HD6801, HD6301 multiplexed bus interface circuit and 8085's multiplexed bus interface as well, so it can be directly connected to HD6801, HD6301 and 8085.

The Real-Time Clock plus RAM has two distinct uses. First, it is designed as battery powered CMOS part including all the common battery backed-up functions such as RAM, time, and calender. Secondly, the HD146818 may be used with a CMOS microprocessor to relieve the software of timekeeping workload and to extend the available RAM of an MPU such as the HD6301.

HD146818P

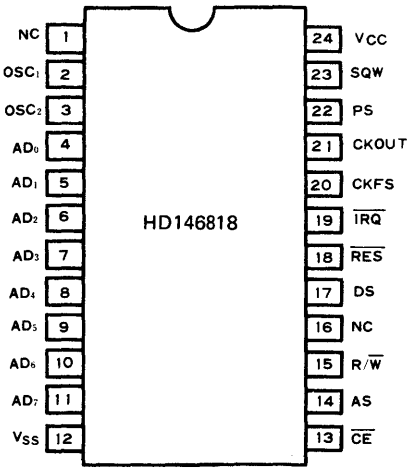


(DP-24)

■ FEATURES

- Time-of-Day Clock and Calendar
  - Counts Seconds, Minutes, and Hours of the Day
  - Counts Days of Week, Date, Month, and Year
- Binary or BCD Representation of Time, Calendar, and Alarm
- 12- or 24 Hour Clock with AM and PM in 12-Hour Mode
- Automatic End of Month Recognition
- Automatic Leap Year Compensation
- Interfaced with Software as 64 RAM Locations
  - 14 Bytes of Clock and Control Register
  - 50 Bytes of General Purpose RAM
- Three Interrupt are Separately Software Maskable and Testable
  - Time-of-Day Alarm, Once-per-Second to Once-per-Day
  - Periodic Rates from 30.5μs to 500ms
  - End-of-Clock Update Cycle
- Programmable Square-Wave Output Signal
- Three Time Base Input Options
  - 4.194304 MHz
  - 1.048576 MHz
  - 32.768 kHz
- Clock Output May be used as Microprocessor Clock Input
  - At Time Base Frequency ÷4 or ÷1
- Multiplexed Bus Interface Circuit of HD6801, HD6301 and 8085
- Low-Power, High-Speed, High-Density CMOS
- Motorola MC146818 Compatible

■ PIN ARRANGEMENT



(Top View)

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC} *	-0.3 ~ +7.0	V
Input Voltage	V_{in} *	-0.3 ~ +7.0	V
Operating Temperature	T_{opr}	0 ~ +70	°C
Storage Temperature	T_{stg}	-55 ~ +150	°C

* With respect to V_{SS} (SYSTEM GND)

(NOTE) Permanent LSI damage may occur if maximum rating are exceeded. Normal operation should be under recomended operating condition. If these conditions are exceeded. it could affect reliability of LSI.

■ RECOMENDED OPERATING CONDITIONS

Item	Symbol	min	typ	max	Unit
Supply Voltage	V_{CC} *	4.5	5.0	5.5	V
Input Voltage	V_{IL} *	-0.3	–	0.7	V
	V_{IH} *	V_{CC}-1.0	–	V_{CC}	V
Operating Temperature	T_{opr}	0	25	70	°C

* With respect to V_{SS} (SYSTEM GND)

■ ELECTRICAL CHARACTERISTICS

• DC CHARACTERISTICS (V_{CC} = 5.0V ± 10%, V_{SS} = 0V, Ta = 0 ~ +70°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Input "High" Voltage	AD_0~AD_7, $\overline{CE}$, AS, R/$\overline{W}$, DS, CKFS, PS	V_{IH}		V_{CC}-2.0	–	V_{CC}	V
	$\overline{RES}$			V_{CC}-1.0	–	V_{CC}	
	OSC_1			V_{CC}-1.0	–	V_{CC}	
Input "Low" Voltage	AD_0~AD_7, $\overline{CE}$, AS, R/$\overline{W}$, DS, CKFS, PS	V_{IL}		-0.3	–	0.7	V
	$\overline{RES}$			-0.3	–	0.8	
	OSC_1			-0.3	–	0.8	
Input Leakage Current	OSC_1, $\overline{CE}$, AS, R/$\overline{W}$, DS, $\overline{RES}$, CKFS, PS	$\lvert I_{in} \rvert$		–	–	2.5	μA
Three-state (off state) Input Current	AD_0~AD_7	$\lvert I_{TSI} \rvert$		–	–	10	μA
Output Leakage Current	$\overline{IRQ}$	I_{LOH}		–	–	10	μA
Output "High" Voltage	AD_0~AD_7	V_{OH}	I_{OH} = -1.6 mA	4.1	–	–	V
	SQW, CKOUT						
	AD_0~AD_7		I_{OH} < -10 μA	V_{CC}-0.1	–	–	V
	SQW, CKOUT						
Output "Low" Voltage	AD_0~AD_7	V_{OL}	I_{OL} = 1.6 mA	–	–	0.5	V
	CKOUT		I_{OL} = 1.6 mA				
	$\overline{IRQ}$, SQW		I_{OL} = 1.6 mA				
Input Capacitance	AD_0~AD_7	C_{in}	V_{in} = 0V, Ta = 25°C, f = 1 MHz	–	–	12.5	pF
	All inputs except AD_0~AD_7			–	–	12.5	pF
Output Capacitance	SQW, CKOUT, $\overline{IRQ}$	C_{out}		–	–	12.5	pF
Supply Current (MPU Read/Write operating) — Crystal Oscillation	f_{OSC} = 4 MHz	I_{CC} *	V_{CC} = 5.0V, SQW: disable, CKOUT = f_{OSC} (No Load), t_{cyc} = 1 μs, Circuit: Fig. 10, Parameter: Table 1	–	–	10	mA
	f_{OSC} = 1 MHz			–	–	7	
	f_{OSC} = 32 kHz			–	–	5	
Supply Current (MPU not operating) — Crystal Oscillation	f_{OSC} = 4 MHz			–	–	5	mA
	f_{OSC} = 1 MHz			–	–	2	
	f_{OSC} = 32 kHz			–	300	500	μA
Supply Current (MPU Read/Write operating) — External Clock	f_{OSC} = 4 MHz	I_{CC} *	V_{CC} = 5.0V, SQW: disable, CKOUT = f_{OSC} (No Load), OSC_2: open, t_{cyc} = 1 μs, Circuit: Fig. 15	–	–	10	mA
	f_{OSC} = 1 MHz			–	–	7	
	f_{OSC} = 32 kHz			–	–	5	
Supply Current (MPU not operating) — External Clock	f_{OSC} = 4 MHz			–	–	4	mA
	f_{OSC} = 1 MHz			–	–	1	
	f_{OSC} = 32 kHz			–	60	100	μA

* The time-base frequency to be used needs to be chosen in Register A.

- **AC CHARACTERISTICS (V_{CC} = 5.0V ± 10%, V_{SS} = 0V, Ta = 0 ~ +70°C, unless otherwise noted.)**

BUS TIMING

Item	Symbol	min	typ	max	Unit
Cycle Time	t_{cyc}	953	–	–	ns
Pulse Width, AS/ALE "High"	PW_{ASH}	100	–	–	ns
AS Rise Time	t_{ASr}	–	–	30	ns
AS Fall Time	t_{ASf}	–	–	30	ns
Delay Time DS/E to AS/ALE Rise	t_{ASD}	40	–	–	ns
DS Rise Time	t_{DSr}	–	–	30	ns
DS Fall Time	t_{DSf}	–	–	30	ns
Pulse Width, DS/E Low or $\overline{RD}/\overline{WR}$ "High"	PW_{DSH}	325	–	–	ns
Pulse Width, DS/E High or $\overline{RD}/\overline{WR}$ "Low"	PW_{DSL}	300	–	–	ns
Delay Time, AS/ALE to DS/E Rise	t_{ASDS}	90	–	–	ns
Address Setup Time (R/$\overline{W}$)	t_{AS1}	15	–	–	ns
Address Setup Time ($\overline{CE}$)	t_{AS2}	55	–	–	ns
Address Hold Time (R/$\overline{W}$, $\overline{CE}$)	t_{AH}	10	–	–	ns
Muxed Address Valid Time to AS/ALE Fall	t_{ASL}	50	–	–	ns
Muxed Address Hold Time	t_{AHL}	20	–	–	ns
Peripheral Data Setup Time	t_{DSW}	195	–	–	ns
Write Data Hold Time	t_{DHW}	0	–	–	ns
Peripheral Output Data Delay Time From DS/E or $\overline{RD}$	t_{DDR}	–	–	220	ns
Read Data Hold Time	t_{DHR}	10	–	–	ns

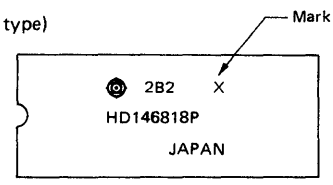
CONTROL SIGNAL TIMING

Item		Symbol	min	typ	max	Unit
Oscillator Startup	1 MHz, 4 MHz	t_{RC}	–	–	100	ms
	32 kHz		–	–	1000	
Reset Pulse Width		t_{RWL}	5.0	–	–	μs
Reset Delay Time		t_{RLH}	5.0	–	–	μs
Power Sense Pulse Width		t_{PWL}	5.0	–	–	μs
Power Sense Delay Time		t_{PLH}	5.0	–	–	μs
$\overline{IRQ}$ Release from DS		t_{IRDS}	–	–	2.0	μs
$\overline{IRQ}$ Release from $\overline{RES}$		t_{IRR}	–	–	2.0	μs
VRT Bit Delay		t_{VRTD}	–	–	2.0	μs

RESTRICTION ON HD146818 USAGE

The daylight saving function can not be performed on the HD146818P (X type). So do not use this function for the system design.

< Type number > HD146818P (X type)



< Restriction on usage >

Please set "0" to DSE bit (Daylight Saving Enable bit) on initializing the control register B.
DSE = "1" is prohibited.

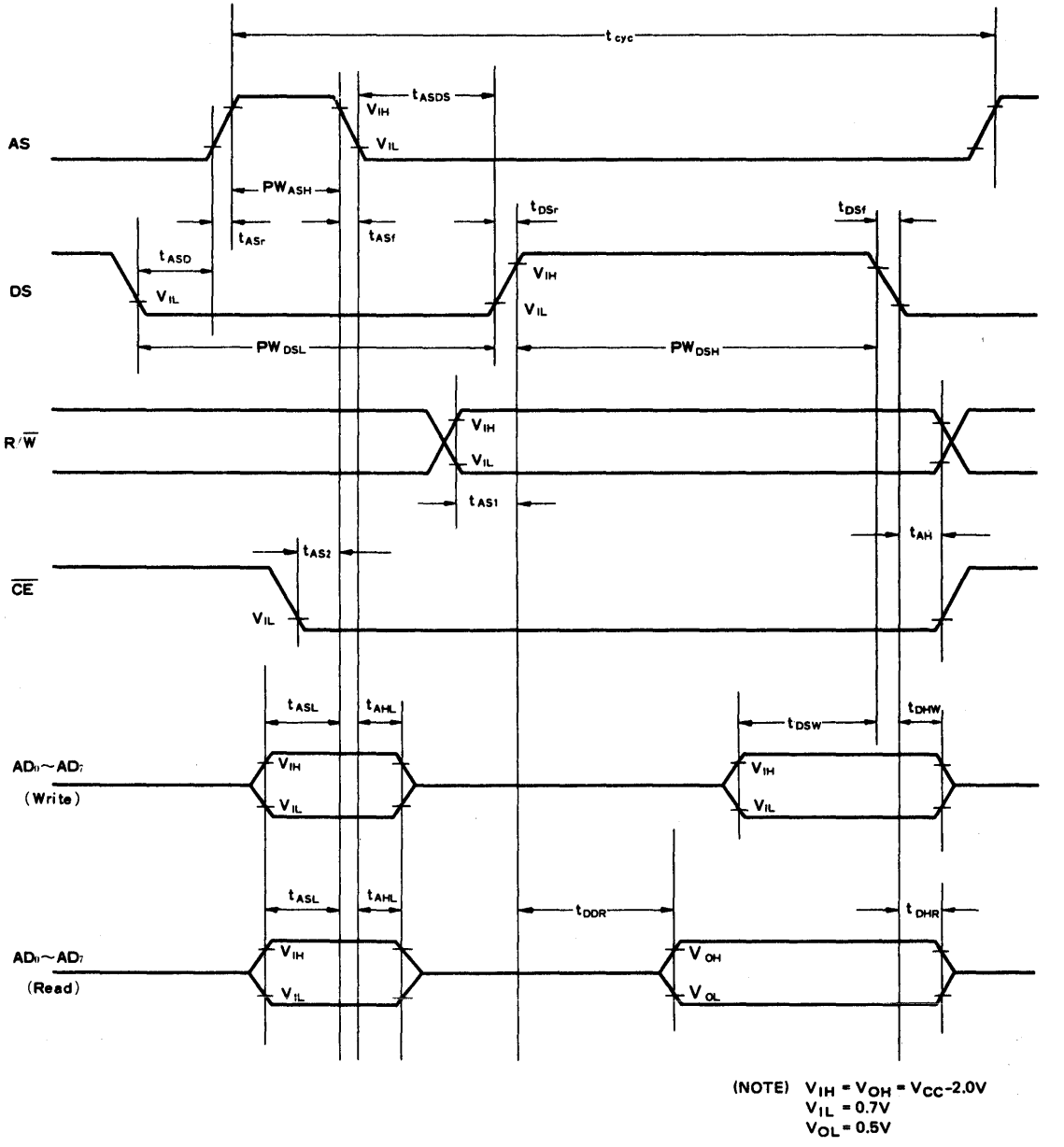
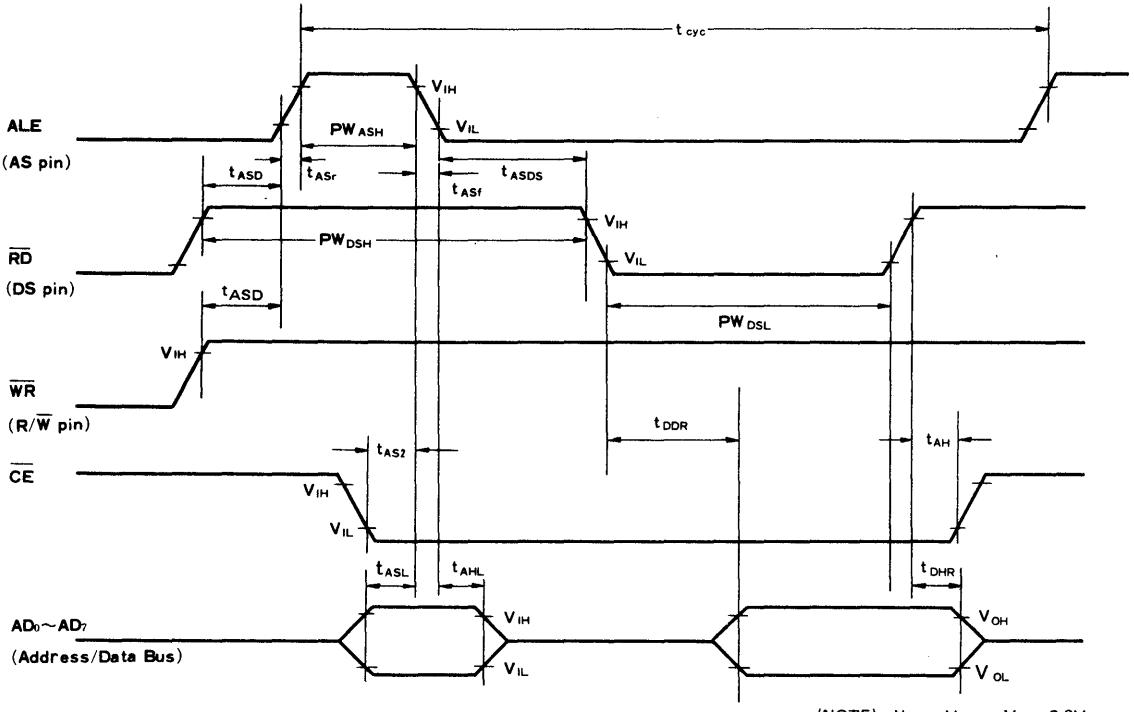
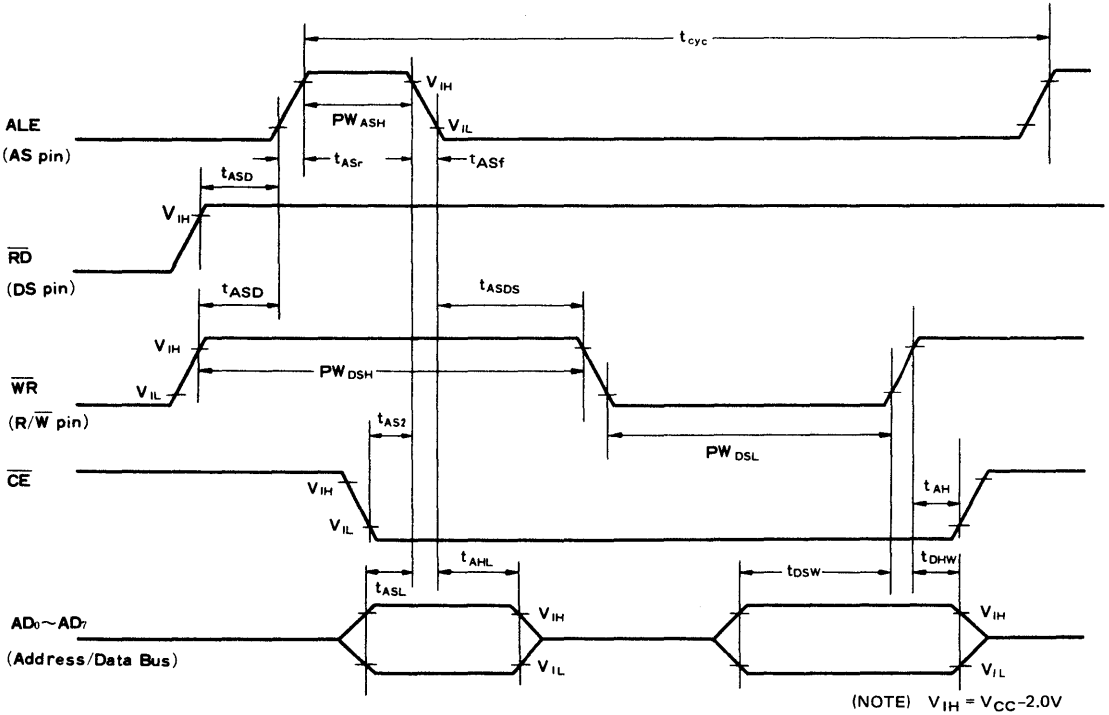


Figure 1 Bus Read, Write Timing (6801 Family)



(NOTE) $V_{IH} = V_{OH} = V_{CC}-2.0V$
$V_{IL} = 0.7V$, $V_{OL} = 0.5V$

Figure 2 Read Timing (8085 Family)



(NOTE) $V_{IH} = V_{CC}-2.0V$
$V_{IL} = 0.7V$

Figure 3 Write Timing (8085 Family)

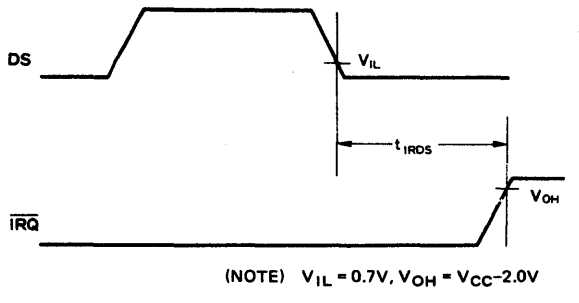


Figure 4 IRQ Release Delay (from DS)

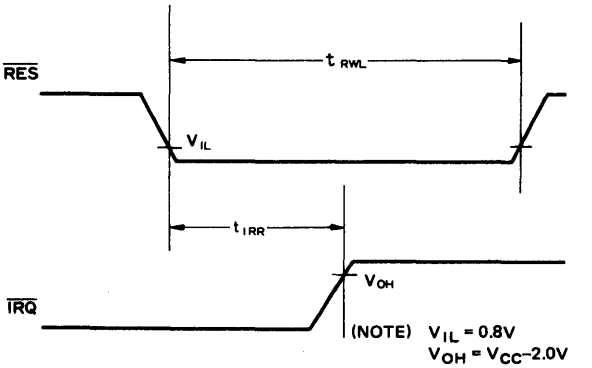
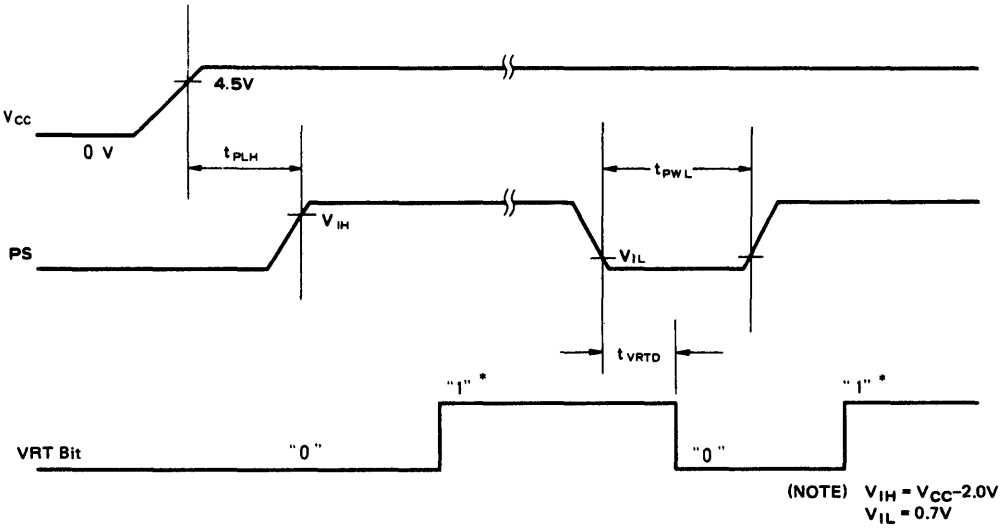


Figure 5 IRQ Release Delay (from RES)



* The VRT bit is set to a "1" by reading control register #D. There is no additional way to clear the VRT bit.

Figure 6 VRT Bit Clear Timing

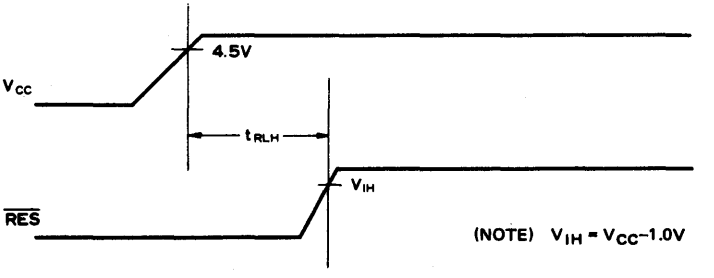


Figure 7 RES Release Delay

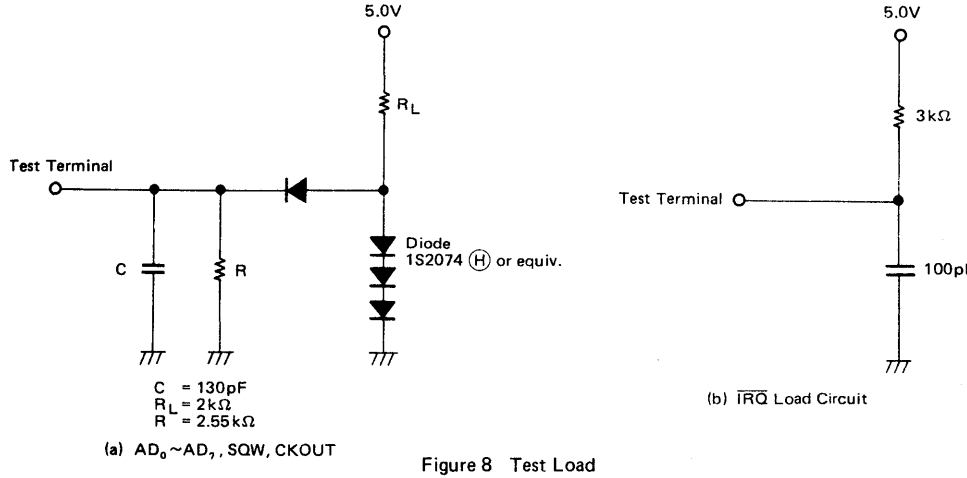


Figure 8 Test Load

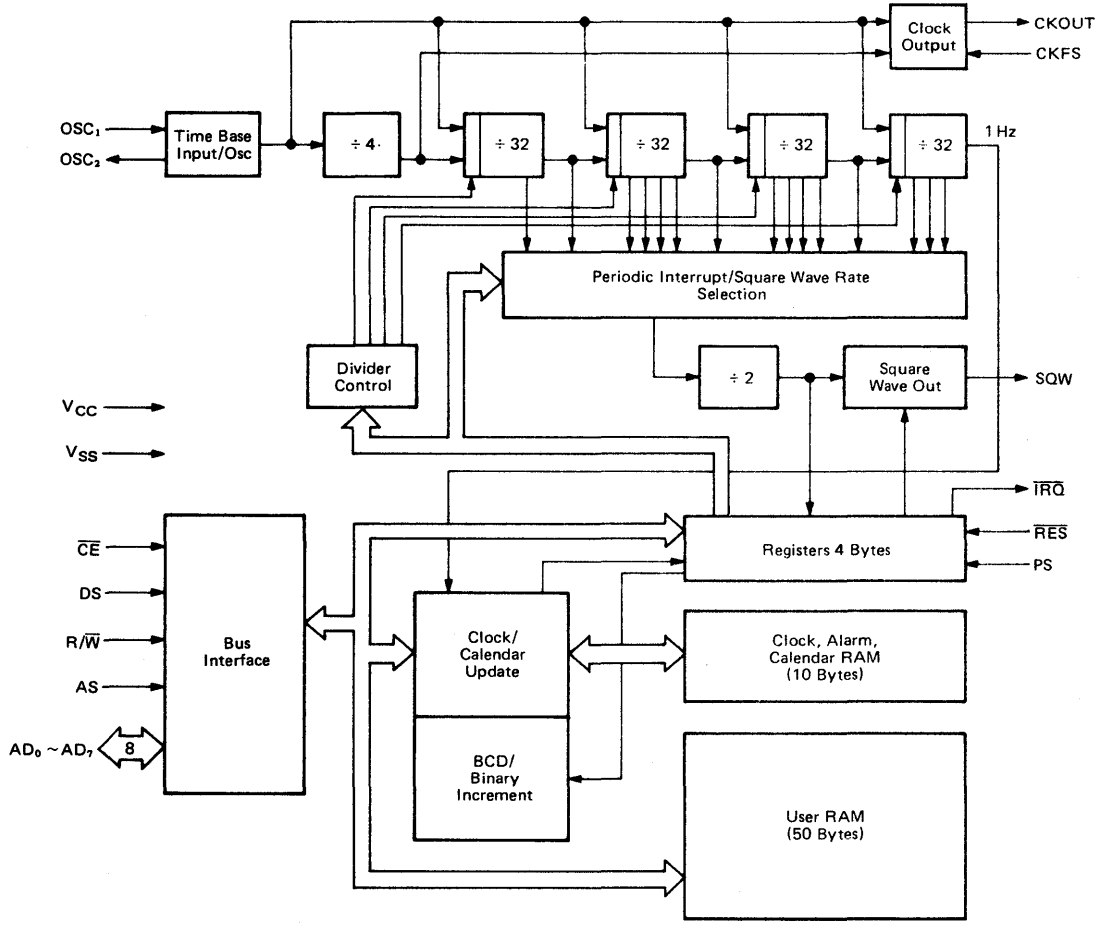


Figure 9 Block Diagram

■ CRYSTAL OSCILLATION CIRCUIT

The on-chip oscillator is designed for a parallel resonant crystal at 4.194304 MHz or 1.048576 MHz or 32.768 kHz frequencies. The crystal connections are shown in Figure 10.

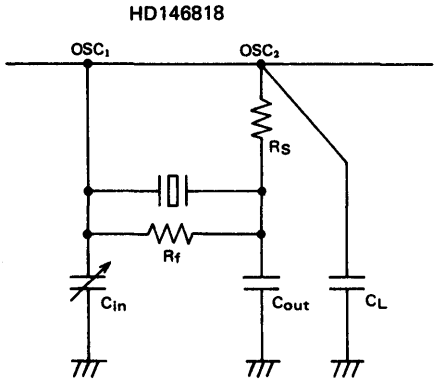


Figure 10 Crystal Oscillator Connection

Table 1 Oscillator Circuit Parameters

Parameter \ fosc	4.194304 MHz	1.048576 MHz	32.768 kHz
R_S	–	–	150 kΩ
R_f	150 kΩ	150 kΩ	5.6 MΩ
C_{in}	22 pF	33 pF	15 pF
C_{out}	22 pF	33 pF	33 pF
C_L	–	–	33 pF
CI	80 Ω (max)	700 Ω (max)	40 kΩ (max)

(NOTE) 1. R_S, C_L are used for 32.768 kHz only.
2. Capacitance (C_{in}) should be adjusted to accurate frequency. Parameters listed above are applied to the supply current measurement (See table of DC CHARACTERISTICS).
3. CI: Crystal Impedance

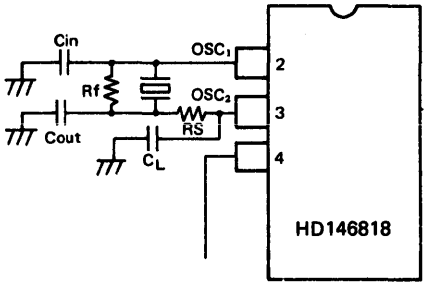
■ NOTE FOR BOARD DESIGN OF THE OSCILLATION CIRCUIT

In designing the board, the following notes should be taken when the crystal oscillator is used.

(1) Crystal oscillator, load capacity C_{in}, C_{out}, C_L and R_f, R_S must be placed near the LSI as much as possible.
[Normal oscillation may be disturbed when external noise is induced to pin 2 and 3.]

(2) Pin 3 signal line should be wired apart from pin 4 signal line as much as possible. Don't wire them in parallel, or normal oscillation may be disturbed when this signal is feedbacked to OSC_1.

(3) A signal line or a power source line must not cross or go near the oscillation circuit line as shown in the right figure to prevent the induction from these lines and perform the correct oscillation. The resistance among OSC_1, OSC_2 and other pins should be over 10MΩ.



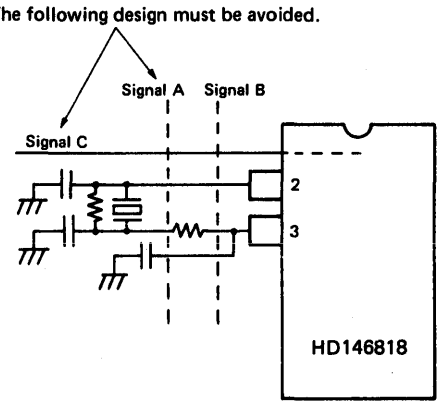


Figure 11 Note for Board Design of the Oscillation Circuit

■ INTERFACE CIRCUIT FOR HD6801, HD6301 AND 8085 PROCESSOR

HD146818 has a new interface circuit which permits the HD146818 to be directly interfaced with many type of multiplexed bus microprocessor such as HD6801, HD6301 and 8085 etc.

Figure 12 shows the bus control circuit. This circuit automatically selects the processor type by using AS/ASE to latch the state of DS/$\overline{RD}$ pin. Since DS is always "Low" and $\overline{RD}$ is always "High during AS/ALE, the latch automatically indicates which processor type is connected.

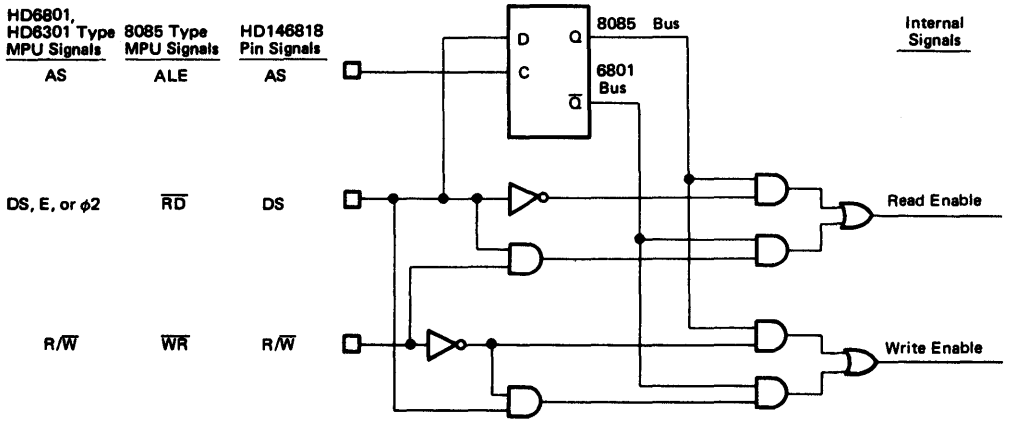


Figure 12 Functional Diagram of the Bus Control Circuit

■ ADDRESS MAP

Figure 13 shows the address map of the HD146818. The memory consists of 50 general purpose RAM bytes, 10 RAM bytes which normally contain the time, calendar, and alarm data, and four control and status bytes. All 64 bytes are directly readable and writable by the processor program except Registers C and D which are read only. Bit 7 of Register A and the seconds byte are also read only. Bit 7, of the second byte, always reads "0". The contents of the four control and status registers are described in the Register section.

• Time, Calendar, and Alarm Locations

The processor program obtains time and calendar information by reading the appropriate locations. The program may initialize the time, calendar, and alarm by writing to these RAM locations. The contents of the 10 time, calendar, and alarm byte may be either binary or binary-coded decimal (BCD).

Before initializing the internal registers, the SET bit in Register B should be set to a "1" to prevent time/calendar updates from occurring. The program initializes the 10 locations in the selected format (binary or BCD), then indicates the format in the data mode (DM) bit of Register B. All 10 time, calendar, and alarm bytes must use the same data mode, either binary or BCD. The SET bit may now be cleared to allow updates. Once initialized the real-time clock makes all updates in the selected data mode. The data mode cannot be changed without reinitializing the 10 data bytes.

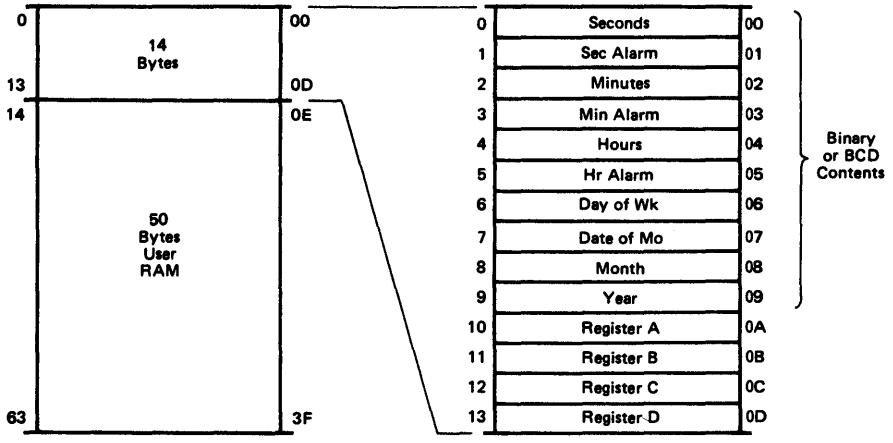


Figure 13 Address Map

Table 2 shows the binary and BCD formats of the 10 time, calendar, and alarm locations. The 24/12 bit in Register B establishes whether the hour locations represent 1-to-12 or 0-to-23. The 24/12 bit cannot be changed without reinitializing the hour locations. When the 12-hour format is selected the high-order bit of the hours byte represents PM when it is a "1".

The time, calendar, and alarm bytes are not always accessable by the processor program. Once-per-second the 10 bytes are switched to the update logic to be advanced by one second and to check for an alarm condition. If any of the 10 bytes are read at this time, the data outputs are undefined. The update lockout time is 248 μs at the 4.194304 MHz and 1.048567 MHz time bases and 1948 μs for the 32.768 kHz time base. The Update Cycle section shows how to accommodate the update cycle in the processor program.

The three alarm bytes may be used in two ways. When the program inserts an alarm time in the appropriate hours, minutes, and seconds alarm locations, the alarm interrupt is initiated at the specified time each day if the alarm enable bit is "1". The alternate usage is to insert a "don't care" state in one or more of three alarm bytes. The "don't care" code is any hexadecimal byte from C0 to FF. That is, the two most-significant bits of each byte, when set to "1", create a "don't care" situation. An alarm interrupt each hour is created with a "don't care" code in the hours alarm location. Similarly, an alarm is generated every minute with "don't care" codes in the hours and minutes alarm bytes. The "don't care" codes in all three alarm bytes create an interrupt every second.

Table 2 Time, Calendar, and Alarm Data Modes

Address Location	Function	Decimal Range	Range		Example*	
			Binary Data Mode	BCD Data Mode	Binary Data Mode	BCD Data Mode
0	Seconds	0~59	$00~$3B	$00~$59	15	21
1	Seconds Alarm	0~59	$00~$3B	$00~$59	15	21
2	Minutes	0~59	$00~$3B	$00~$59	3A	58
3	Minutes Alarm	0~59	$00~$3B	$00~$59	3A	58
4	Hours (12 Hour Mode)	1~12	$01~$0C (AM) and $81~$8C (PM)	$01~$12 (AM) and $81~$92 (PM)	05	05
	Hours (24 Hour Mode)	0~23	$00~$17	$00~$23	05	05
5	Hours Alarm (12 Hour Mode)	1~12	$01~$0C (AM) and $81~$8C (PM)	$01~$12 (AM) and $81~$92 (PM)	05	05
	Hours Alarm (24 Hour Mode)	0~23	$00~$17	$00~$23	05	05
6	Day of the Week Sunday = 1	1~7	$01~$07	$01~$07	05	05
7	Day of the Month	1~31	$01~$1F	$01~$31	0F	15
8	Month	1~12	$01~$0C	$01~$12	02	02
9	Year	0~99	$00~$63	$00~$99	4F	79

* Example: 5:58:21 Thursday 15th February 1979

• Static CMOS RAM

The 50 general purpose RAM bytes are not dedicated within the HD146818. They can be used by the processor program, and are fully available during the update cycle.

When time and calendar information must use battery back-up, very frequently there is other non-volatile data that must be retained when main power is removed. The 50 user RAM bytes serve the need for low-power CMOS battery-backed storage, and extend the RAM available to the program.

When further CMOS RAM is needed, additional HD146818s may be included in the system. The time/calendar functions may be disabled by holding the dividers, in Register A, in the reset state by setting the SET bit in Register B or by removing the oscillator. Holding the dividers in reset prevents interrupts or SQW output from operating while setting the SET bit allows these functions to occur. With the dividers clear, the available user RAM is extended to 59 bytes. Bit 7 of Register A, Registers C and D, and the high-order Bit of the seconds byte cannot effectively be used as general purpose RAM.

■ INTERRUPTS

The RTC plus RAM includes three separate fully automatic sources of interrupts to the processor. The alarm interrupt may be programmed to occur at rates from once-per-second to one-a-day. The periodic interrupt may be selected for rates from half-a-second to 30.517 μs. The update-ended interrupt may be used to indicate to the program that an up-date cycle is completed. Each of these independent interrupt conditions are described in greater detail in other sections.

The processor program selects which interrupts, if any, it wishes to receive. Three bits in Register B enable the three interrupts. Writing a "1" to a interrupt-enable bit permits that interrupt to be initiated when the event occurs. A "0" in the interrupt-enable bit prohibits the $\overline{IRQ}$ pin from being asserted due to the interrupt cause.

If an interrupt flag is already set when the interrupt becomes enabled, the $\overline{IRQ}$ pin is immediately activated, though the interrupt initiating the event may have occurred much earlier. Thus, there are cases where the program should clear such

earlier initiated interrupts before first enabling new interrupts.

When an interrupt event occurs a flag bit is set to a "1" in Register C. Each of the three interrupt sources have separate flag bits in Register C, which are set independent of the state of the corresponding enable bits in Register B. The flag bit may be used with or without enabling the corresponding enable bits.

In the software scanned case, the program does not enable the interrupt. The "interrupt" flag bit becomes a status bit, which the software interrogates, when it wishes. When the software detects that the flag is set, it is an indication to software that the "interrupt" event occurred since the bit was last read.

However, there is one precaution. The flag bits in Register C are cleared (record of the interrupt event is erased) when Register C is read. Double latching is included with Register C so the bits which are set are stable throughout the read cycle. All bits which are high when read by the program are cleared, and new interrupts (on any bits) are held until after the read cycle. One, two, or three flag bits may be found to be set when Register C is read. The program should inspect all utilized flag bits every time Register C is read to insure that no interrupts are lost.

The second flag bit usage method is with fully enabled interrupts. When an interrupt-flag bit is set and the corresponding interrupt-enable bit is also set, the $\overline{IRQ}$ pin is asserted "Low". $\overline{IRQ}$ is asserted as long as at least one of the three interrupt sources has its flag and enable bits both set. The IRQF bit in Register C is a "1" whenever the $\overline{IRQ}$ pin is being driven "Low"

The processor program can determine that the RTC initiated the interrupt by reading Register C. A "1" in bit 7 (IRQF bit) indicates that one of more interrupts have been initiated by the part. The act of reading Register C clears all the then-active flag bits, plus the IRQF bit. When the program finds IRQF set, it should look at each of the individual flag bits in the same byte which have the corresponding interrupt-mask bits set and service each interrupt which is set. Again, more than one interrupt-flag bit may be set.

■ DIVIDER STAGES

The HD146818 has 22 binary-divider stages following the time base as shown in Figure 9. The output of the dividers is a 1 Hz signal to the update-cycle logic. The dividers are controller by three divider bus (DV2, DV1, and DV0) in Register A.

• Divider Control

The divider-control bits have three uses, as shown in Table 3. Three usable operating time bases may be selected (4.194304 MHz, 1.048576 MHz, or 32.768 kHz). The divider chain may be held reset, which allows precision setting of the time. When the divider is changed from reset to an operating time base, the first update cycle is one second later. The divider-control bits are also used to facilitate testing the HD146818.

Table 3 Divider Configurations

Time-Base Frequency	Divider Bits Register A			Operation Mode	Divider Reset	Bypass First N-Divider Bits
	DV2	DV1	DV0			
4.194304 MHz	0	0	0	Yes	–	N = 0
1.048576 MHz	0	0	1	Yes	–	N = 2
32.768 kHz	0	1	0	Yes	–	N = 7
Any	1	1	0	No	Yes	–
Any	1	1	1	No	Yes	–

(NOTE) Other combinations of divider bits are used for test purposes only.

• Square-Wave Output Selection

Fifteen of the 22 divider taps are made available to a 1-of-15 selector as shown in Figure 9. The first purpose of selecting a divider tap is to generate a square-wave output signal in the SQW pin. Four bits in Register A establish the square-wave frequency as listed in Table 4. The SQW frequency selection shares the 1-of-15 selector with periodic interrupts.

Once the frequency is selected, the output of the SQW pin may be turned on and off under program control with the square-wave enable (SQWE) bit in Register B. Altering the divider, square-wave output selection bits, or the SQW output-enable bit may generate an asymetrical waveform at the time of execution. The square-wave output pin has a number of potential uses. For example, it can serve as a frequency standard for external use, a frequency synthesizer, or could be used to generate one or more audio tones under program control.

• Periodic Interrupt Selection

The periodic interrupt allows the $\overline{IRQ}$ pin to be triggered from once every 500 ms to once every 30.517 μs. The periodic interrupt is separate from the alarm interrupt which may be output from once-per-second to once-per-day.

Table 4 shows that the periodic interrupt rate is selected with the same Register A bits which select the square-wave frequency. Changing one also changes the other. But each function may be separately enabled so that a program could switch between the two features or use both. The SQW pin is enabled by the SQWE bit. Similarly the periodic interrupt is enabled by the PIE bit in Register B.

Periodic interrupt is usable by practically all real-time systems. It can be used to scan for all forms of input from contact closures to serial receive bits or tyes. It can be used in multiplexing displays or with software counters to measure inputs, create output intervals, or await the next needed software function.

Table 4 Periodic Interrupt Rate and Square Wave Output Frequency

Rate Select Control Register 1				4.194304 or 1.048576 MHz Time Base		32.768 kHz Time Base	
RS3	RS2	RS1	RS0	Periodic Interrupt Rate t_{PI}	SQW Output Frequency	Periodic Interrupt Rate t_{PI}	SQW Output Frequency
0	0	0	0	None	None	None	None
0	0	0	1	30.517 μs	32.768 kHz	3.90625 ms	256 Hz
0	0	1	0	61.035 μs	16.384 kHz	7.8125 ms	128 Hz
0	0	1	1	122.070 μs	8.192 kHz	122.070 μs	8.192 kHz
0	1	0	0	244.141 μs	4.096 kHz	244.141 μs	4.096 kHz
0	1	0	1	488.281 μs	2.048 kHz	488.281 μs	2.048 kHz
0	1	1	0	976.562 μs	1.024 kHz	976.562 μs	1.024 kHz
0	1	1	1	1.953125 ms	512 Hz	1.953125 ms	512 Hz
1	0	0	0	3.90625 ms	256 Hz	3.90625 ms	256 Hz
1	0	0	1	7.8125 ms	128 Hz	7.8125 ms	128 Hz
1	0	1	0	15.625 ms	64 Hz	15.625 ms	64 Hz
1	0	1	1	31.25 ms	32 Hz	31.25 ms	32 Hz
1	1	0	0	62.5 ms	16 Hz	62.5 ms	16 Hz
1	1	0	1	125 ms	8 Hz	125 ms	8 Hz
1	1	1	0	250 ms	4 Hz	250 ms	4 Hz
1	1	1	1	500 ms	2 Hz	500 ms	2 Hz

■ UPDATE CYCLE

The HD146818 executes an update cycle once-per-second, assuming one of the proper time bases is in place, the divider is not clear, and the SET bit in Register B is clear. The SET bit in the "1" state permits the program to initialize the time and calendar bytes by stopping an existing update and preventing a new one from occurring.

The primary function of the update cycle is to increment the seconds byte, check for overflow, increment the minutes byte when appropriate and so forth through to the year of the century byte. The update cycle also compares each alarm byte with the corresponding time byte and issues an alarm if a match or if a "don't care" code (11XXXXXX) is present in all three positions.

With a 4.194304 MHz or 1.048576 MHz time base the update cycle takes 248 μs while a 32.768 kHz time base update cycle takes 1984 μs. During the update cycle, the time, calendar, and alarm bytes are not accessable by the processor program. The HD146818 protects the program from reading transitional data. This protection is provided by switching the time, calendar, and alarm portion of the RAM off the microprocessor bus during the entire update cycle. If the processor reads these RAM locations before the update is complete the output will be undefined. The update in progress (UIP) status bit is set during the interval.

A program which randomly accesses the time and date information finds data unavailable statistically once every 4032 attempts. Three methods of accommodating nonavailability during update are usable by the program. In discussing the three methods it is assumed that at random points user programs are able to call a subroutine to obtain the time of day.

The first method of avoiding the update cycle uses the update-ended interrupt. If enabled, an interrupt occurs after every update cycle which indicates that over 999 ms are available to read valid time and date information. During this time a display could be updated or the information could be transfered to continuously available RAM. Before leaving the interrupt service routine, the IRQF bit in Register C should be cleared.

The second method uses the update-in-progress bit (UIP) in Register A to determine if the update cycle is in progress or not. The UIP bit will pulse once-per-second. Statistically, the UIP bit will indicate that time and date information is unavailable once every 2032 attempts. After the UIP bit goes "1", the update cycle begins 244 μs later. Therefore, if a "0" is read on the UIP bit, the user has at least 244 μs before the time/calendar data will be changed. If a "1" is read in the UIP bit, the time/calendar data may not be valid. The user should avoid interrupt service routines that would cause the time needed to read valid time/calendar data to exceed 244 μs.

The third method uses a periodic interrupt to determine if an update cycle is in progress. The UIP bit in Register A is set "1" between the setting of the PF bit in Register C (see Figure 14) Periodic interrupts that occur at a rate of greater than $t_{BUC} + t_{UC}$ allow valid time and date information to be read at each occurrence of the periodic interrupt. The reads should be completed within $(t_{PI} \div 2) + t_{BUC}$ to insure that data is not read during the update cycle.

■ POWER-DOWN CONSIDERATIONS

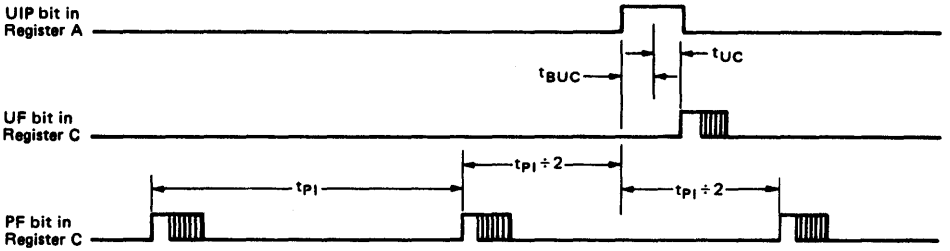
In most systems, the HD146818 must continue to keep time when system power is removed. In such systems, a conversion from system power to an alternate power supply, usually a battery, must be made. During the transition from system to battery power, the designer of a battery backed-up RTC system must protect data integrity, minimize power

consumption, and ensure hardware reliability.

The chip enable ($\overline{CE}$) pin controls all bus inputs (R/$\overline{W}$, DS, AS, $AD_0 \sim AD_7$). $\overline{CE}$, when negated, disallows any unintended modification of the RTC data by the bus. $\overline{CE}$ also reduces power consumption by reducing the number of transitions seen internally.

Power consumption may be further reduced by removing resistive and capacitive loads from the clock out (CKOUT) pin and the squarewave (SQW) pin.

During and after the power source conversion, the V_{IN} maximum specification must never be exceeded. Failure to meet the V_{IN} maximum specification can cause a virtual SCR to appear which may result in excessive current drain and destruction of the part.



t_{PI} = Periodic Interrupt Time Interval (500 ms, 250 ms, 125 ms, 62.5 ms, etc.)
t_{UC} = Update Cycle Time (248 μs or 1984 μs)
t_{BUC} = Delay Time Before Update Cycle (244 μs)

Figure 14 Update-Ended and Periodic Interrupt Relationship

■ SIGNAL DESCRIPTIONS

The block diagram in Figure 9, shows the pin connection with the major internal functions of the HD146818 Real-Time Clock plus RAM. The following paragraphs describe the function of each pin.

• V_{CC}, V_{SS}

DC power is provided to the part on these two pins, V_{CC} being the most positive voltage. The minimum and maximum voltages are listed in the Electrical Characteristics tables.

• OSC_1, OSC_2 – Time Base (Inputs)

The time base for the time functions may be an external signal or the crystal oscillator. External square waves at 4.194304 MHz, 1.048576 MHz, or 32.768 kHz may be connected to OSC_1 as shown in Figure 15. The time-base frequency to be used is chosen in Register A.

The on-chip oscillator is designed for a parallel resonant crystal at 4.194304 MHz or 1.048576 MHz or 32.768 kHz frequencies. The crystal connections are shown in Figure 10.

• CKOUT – Clock Out (Output)

The CKOUT pin is an output at the time-base frequency divided by 1 or 4. A major use for CKOUT is as the input clock to the microprocessor; thereby saving the cost of a second crystal. The frequency of CKOUT depends upon the time-base frequency and the state of the CKFS pin as shown in Table 5.

• CKFS – Clock Out Frequency Select (Input)

The CKOUT pin is an output at the time-base frequency divided by 1 or 4. CKFS tied to V_{CC} causes CKOUT to be the same frequency as the time base at the OSC_1 pin. When CKFS is at V_{SS}, CKOUT is the OSC_1 time-base frequency divided by four. Table 5 summarizes the effect of CKFS.

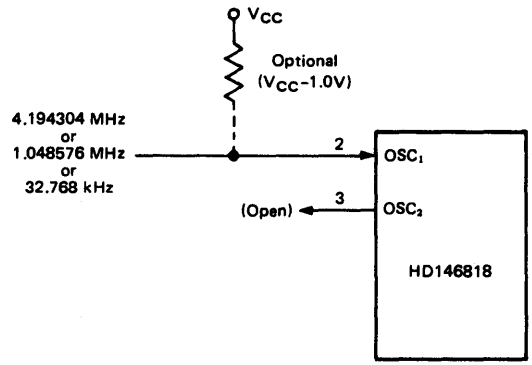


Figure 15 External Time-Base Connection

Table 5 Clock Output Frequencies

Time Base (OSC_1) Frequency	Clock Frequency Select Pin (CKFS)	Clock Frequency Output Pin (CKOUT)
4.194304 MHz	"High"	4.194304 MHz
4.194304 MHz	"Low"	1.048576 MHz
1.048576 MHz	"High"	1.048576 MHz
1.048576 MHz	"Low"	262.144 kHz
32.768 kHz	"High"	32.768 kHz
32.768 kHz	"Low"	8.192 kHz

- **SQW – Square Wave (Output)**

The SQW pin can output a signal one of 15 of the 22 internal-divider stages. The frequency and output enable of the SQW may be altered by programming Register A, as shown in Table 4. The SQW signal may be turned on and off using a bit in Register B.

- **$AD_0 \sim AD_7$ – Multiplexed Bidirectional Address/Data Bus**

Multiplexed bus processors save pins by presenting the address during the first portion of the bus cycle and using the same pins during the second portion for data. Address-then-data multiplexing does not slow the access time of the HD146818 since the bus reversal from address to data is occurring during the internal RAM access time.

The address must be valid just prior to the fall of AS/ALE at which time the HD146818 latches the address from AD_0 to AD_5. Valid write data must be presented and held stable during the latter portion of the DS or $\overline{WR}$ pulses. In a read cycle, the HD146818 outputs 8 bits of data during the latter portion of the DS or $\overline{RD}$ pulses, then ceases driving the bus (returns the output drivers to three-state) when DS falls in the HD6801, HD6301 case or $\overline{RD}$ rises in the other case.

- **AS – Multiplexed Address Strobe (Input)**

A positive going multiplexed address strobe pulse serves to demultiplex the bus. The falling edge of AS or ALE causes the address to be latched within the HD146818. The bus control circuit in the HD146818 also latches the state of the DS pin with the falling edge of AS or ALE.

- **DS – Data Strobe or Read (Input)**

The DS pin has two interpretations via the bus control circuit. When emanating from 6801 family type processor, DS is a positive pulse during the latter portion of the bus cycle, and is variously called DS (data strobe), E (enable), and ϕ_2 (ϕ_2 clock). During read cycles, DS signifies the time that the RTC is to drive the bidirectional bus. In write cycles, the trailing edge of DS causes the Real-Time Clock plus RAM to latch the written data.

The second interpretation of DS is that of $\overline{RD}$, $\overline{MEMR}$, or $\overline{I/OR}$ emanating from the 8085 type processor. In this case, DS identifies the time period when the real-time clock plus RAM drives the bus with read data. This interpretation of DS is also the same as an output-enable signal on a typical memory.

The bus control circuit, within the HD146818, latches the state of the DS pin on the falling edge of AS/ALE. When 6801 mode, DS must be "Low" during AS/ALE, which is the case with 6801 family multiplexed bus processors. To insure the 8085 mode of this circuit the DS pin must remain "High" during the time AS/ALE is "High".

- **R/$\overline{W}$ – Read/Write (Input)**

The bus control circuit treats the R/$\overline{W}$ pin in one of two ways. When 6801 family type processor is connected, R/$\overline{W}$ is a level which indicates whether the current cycle is a read or write. A read cycle is indicated with a "High" level on R/$\overline{W}$ while DS is "High", whereas a write cycle is a "Low" on R/$\overline{W}$ during DS

The second interpretation of R/$\overline{W}$ is as a negative write pulse, $\overline{WR}$, $\overline{MEMW}$, and $\overline{I/OW}$ from 8085 type processors. This circuit in this mode gives R/$\overline{W}$ pin the same meaning as the write ($\overline{W}$) pulse on many generic RAMs.

- **$\overline{CE}$ – Chip Enable (Input)**

The chip-enable ($\overline{CE}$) signal must be asserted ("Low") for a bus cycle in which the HD146818 is to be accessed. $\overline{CE}$ is not latched and must be stable during DS and AS (in the 6801 case) and during $\overline{RD}$ and $\overline{WR}$ (in the 8085 case). Bus cycles which take place without asserting $\overline{CE}$ cause no actions to take place within the HD146818. When $\overline{CE}$ is "High", the multiplexed bus output is in a high-impedance state.

When $\overline{CE}$ is "High", all address, data, DS, and R/$\overline{W}$ inputs from the processor are disconnected within the HD146818. This permits the HD146818 to be isolated from a powered-down processor. When $\overline{CE}$ is held "High", an unpowered device cannot receive power through the input pins from the real-time clock power source. Battery power consumption can thus be reduced by using a pullup resistor or active clamp on $\overline{CE}$ when the main power is off.

- **$\overline{IRQ}$ – Interrupt Request (Output)**

The $\overline{IRQ}$ pin is an active "Low" output of the HD146818 that may be used as an interrupt input to a processor. The $\overline{IRQ}$ output remains "Low" as long as the status bit causing the interrupt is present and the corresponding interrupt-enable bit is set. To clear the $\overline{IRQ}$ pin, the processor program normally reads Register C. The $\overline{RES}$ pin also clears pending interrupts.

When no interrupt conditions are present, the $\overline{IRQ}$ level is in the high-impedance state. Multiple interrupting devices may thus be connected to an $\overline{IRQ}$ bus with one pullup at the processor.

- **$\overline{RES}$ – Reset (Input)**

The $\overline{RES}$ pin does not affect the clock, calendar, or RAM functions. On powerup, the $\overline{RES}$ pin must be held "Low" for the specified time, t_{RLH}, in order to allow the power supply to stabilize, Figure 16 shows a typical representation of the $\overline{RES}$ pin circuit.

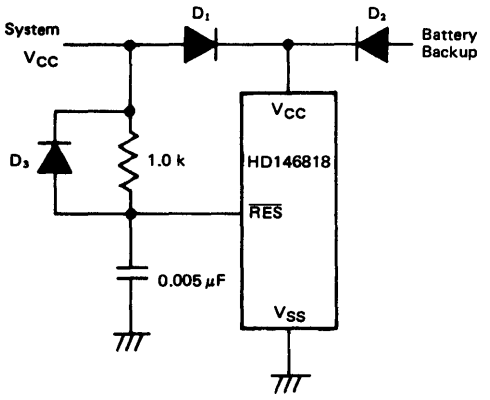
When $\overline{RES}$ is "Low" the following occurs:

a) Periodic Interrupt Enable (PIE) bit is cleared to "0".
b) Alarm Interrupt Enable (AIE) bit is cleared to "0".
c) Update ended interrupt Enable (UIE) bit is cleared to "0".
d) Update ended Interrupt Flag (UF) bit is cleared to "0".
e) Interrupt Request status Flag (IRQF) bit is cleared to "0".
f) Periodic Interrupt Flag (PF) bit is cleared to "0".
g) Alarm Interrupt Flag (AF) bit is cleared to "0".
h) $\overline{IRQ}$ pin is in high-impedance state, and
i) Square Wave output Enable (SQWE) bit is cleared to "0".

- **PS – Power Sense (Input)**

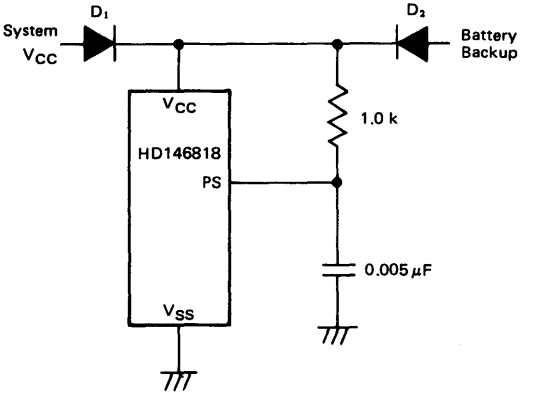
The power-sense pin is used in the control of the valid RAM and time (VRT) bit in Register C. When the PS pin is "Low" the VRT bit is cleared to "0".

During powerup, the PS pin must be externally held "Low" for the specified time, t_{PLH}. As power is applied the VRT bit remain "Low" indicating that the contents of the RAM, time registers, and calendar are not guaranteed. When normal operation commences PS should be permitted to go "High". Figure 17 shows a typical circuit connection for the power-sense pin.



(NOTE) If the RTC is isolated from the MPU or MCU power by a diode drop, care must be taken to meet V_{in} requirements.

Figure 16 Typical Powerup Delay Circuit for $\overline{RES}$



(NOTE) If the RTC is isolated from the MPU or MCU power by a diode drop, care must be taken to meet V_{in} requirements.

Figure 17 Typical Powerup Delay Circuit for Power Sense

■ REGISTERS

The HD146818 has four registers which are accessible to the processor program. The four registers are also fully accessible during the update cycle.

• Register A ($0A)

MSB							LSB	
b7	b6	b5	b4	b3	b2	b1	b0	Read/Write Register except UIP
UIP	DV2	DV1	DV0	RS3	RS2	RS1	RS0	

UIP – The update in progress (UIP) bit is a status flag that may be monitored by the program. When UIP is a "1" the update cycle is in progress or will soon begin. When UIP is a "0" the update cycle is not in progress and will not be for at least 244 μs (for all time bases). This is detailed in Table 6. The time, calendar, and alarm information in RAM is fully available to the program when the UIP bit is zero – it is not in transition. The UIP bit is a read-only bit, and is not affected by Reset. Writing the SET bit in Register B to a "1" inhibit any update cycle and then clear the UIP status bit.

Table 6 Update Cycle Times

UIP Bit	Time Base (OSC_1)	Update Cycle Time (t_{UC})	Minimum Time Before Update Cycle (t_{BUC})
1	4.194304 MHz	248 μs	–
1	1.048576 MHz	248 μs	–
1	32.768 kHz	1984 μs	–
0	4.194304 MHz	–	244 μs
0	1.048576 MHz	–	244 μs
0	32.768 kHz	–	244 μs

DV2, DV1, DV0 – Three bits are used to permit the program to select various conditions of the 22-stage divider chain. The divider selection bits identify which of the three time-base frequencies is in use. Table 3 shows that time bases of 4.194304 MHz, 1.048576 MHz, and 32.768 kHz may be used. The divider selection bits are also used to reset the divider chain. When the time/calendar is first initialized, the program may start the divider at the precise time stored in the RAM. When the divider reset is removed the first update cycle begins one second later. These three read/write bits are never modified by the RTC and are not affected by $\overline{RES}$.

RS3, RS2, RS1, RS0 – The four rate selection bits select one of 15 taps on the 22-stage divider, or disable the divider output. The tap selected may be used to generate an output square wave (SQW pin) and/or a periodic interrupt. The program may do one of the following: 1) enable the interrupt with the PIE bit, 2) enable the SQW output pin with the SQWE bit, 3) enable both at the same time at the same rate, or 4) enable neither. Table 4 lists the periodic interrupt rates and the square-wave frequencies that may be chosen with the RS bits. These four bits are read/write bits which are not affected by $\overline{RES}$ and are never changed by the RTC.

• Register B ($0B)

MSB							LSB	
b7	b6	b5	b4	b3	b2	b1	b0	Read/Write Register
SET	PIE	AIE	UIE	SQWE	DM	24/12	DSE	

SET – When the SET bit is a "0", the update cycle functions normally by advancing the counts once-per-second. When the SET bit is written to a "1", any update cycle in progress is aborted and the program may initialize the time and calendar

bytes without an update occurring in the midst of initializing. SET is a read/write bit which is not modified by $\overline{RES}$ or internal functions of the HD146818.

PIE – The periodic interrupt enable (PIE) bit is a read/write bit which allows the periodic-interrupt flag (PF) bit to cause the $\overline{IRQ}$ pin to be driven "Low". A program writes a "1" to the PIE bit in order to receive periodic interrupts at the rate specified by the RS3, RS2, RS1, and RS0 bits in Control Register A. A "0" in PIE blocks $\overline{IRQ}$ from being initiated by a periodic interrupt, but the periodic flag (PF) bit is still at the periodic rate. PIE is not modified by any internal HD146818 functions, but is cleared to "0" by a $\overline{RES}$.

AIE – The alarm interrupt enable (AIE) bit is a read/write bit which when set to a "1" permits the alarm flag (AF) to assert $\overline{IRQ}$. An alarm interrupt occurs for each second that the three time bytes equal the three alarm bytes (including a "don't care" alarm code of binary 11XXXXXX). When the AIE bit is a "0", the AF bit does not initiate an $\overline{IRQ}$ signal. The $\overline{RES}$ pin clears AIE to "0". The internal functions do not affect the AIE bit.

UIE – The UIE (update-ended interrupt enable) bit is a read/write bit which enables the update-end flage (UF) bit to assert $\overline{IRQ}$. The $\overline{RES}$ pin going "Low" or the SET bit going "1" clears the UIE bit.

SQWE – When the square-wave enable (SQWE) bit is set to a "1" by the program, a square-wave signal at the frequency specified in the rate selection bits (RS3 to RS0) appears on the SQW pin. When the SQWE bit is set to a "0" the SQW pin is held "Low". The state of SQWE is cleared by the $\overline{RES}$ pin. SQWE is a read/write bit.

DM – The data mode (DM) bit indicates whether time and calendar updates are to use binary or BCD formats. The DM bit is written by the processor program and may be read by the program, but is not modified by any internal functions or $\overline{RES}$. A "1" in DM signifies binary data, while a "0" in DM specified binary-coded-decimal (BCD) data.

24/12 – The 24/12 control bit establishes the format of the hours bytes as either the 24-hour mode (a "1") or the 12-hour mode (a "0"). This is a read/write bit, which is affected only by the software.

DSE – The daylight savings enable (DSE) bit is a read/write bit which allows the program to enable two special updates (when DSE is a "1"). On the last Sunday in April the time increments from 1:59:59 AM to 3:00:00 AM. On the last Sunday in October when the time first reaches 1:59:59 AM it changes to 1:00:00 AM. These special updates do not occur when the DSE bit is a "0". DSE is not changed by any internal operations or reset.

- **Register C ($0C)**

MSB LSB

b7	b6	b5	b4	b3	b2	b1	b0
IRQF	PF	AF	UF	0	0	0	0

Read-Only Register

IRQF – The interrupt request flag (IRQF) is set to a "1" when one or more of the following are true:

PF = PIE = "1"
AF = AIE = "1"
UF = UIE = "1"

i.e., IRQF = PF · PIE + AF · AIE + UF · UIE

Any time the IRQF bit is a "1", the $\overline{IRQ}$ pin is driven "Low". All flag bits are cleared after Register C is read by the program or when the $\overline{RES}$ pin is low. A program write to Register C does not modify any of the flag bits.

PF – The periodic interrupt flag (PF) is a read-only bit which is set to a "1" when a particular edge is detected on the selected tap of the divider chain. The RS3 to RS0 bits establish the periodic rate. PF is set to a "1" independent of the state of the PIE bit. PF being a "1" initiates an $\overline{IRQ}$ signal and the IRQF bit when PIE is also a "1". The PF bit is cleared by a $\overline{RES}$ or a software read of Register C.

AF – A "1" in the AF (alarm interrupt flag) bit indicates that the current time has matched the alarm time. A "1" in the AF causes the $\overline{IRQ}$ pin to go "Low", and a "1" to appear in the IRQF bit, when the AIE bit also is a "1". A $\overline{RES}$ or a read of Register C clears AF.

UF – The update-ended interrupt flag (UF) bit is set after each update cycle. When the UIE bit is a "1", the "1" in UF causes the IRQF bit to be a "1", asserting $\overline{IRQ}$. UF is cleared by a Register C read or a $\overline{RES}$.

b3 to b0 – The unused bits of Status Register C are read as "0's". They can not be written.

- **Register D ($0D)**

MSB LSB

b7	b6	b5	b4	b3	b2	b1	b0
VRT	0	0	0	0	0	0	0

Read Only Register

VRT – The valid RAM and time (VRT) bit indicates the condition of the contents of the RAM, provided the power sense (PS) pin is satisfactorily connected. A "0" appears in the VRT bit when the power-sense pin is "Low". The processor program can set the VRT bit when the time and calendar are initialized to indicate that the RAM and time are valid. The VRT is a read/only bit which is not modified by the $\overline{RES}$ pin. The VRT bit can only be set by reading the Register D.

b6 to b0 – The remaining bits of Register D are unused. They cannot be written, but are always read as "0's".

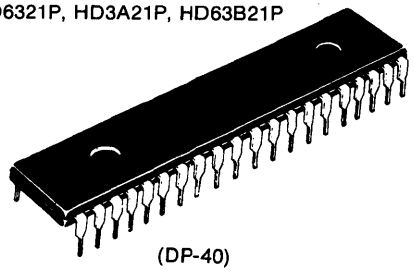
HD6321, HD63A21, HD63B21
CMOS PIA (Peripheral Interface Adapter)

ADVANCED INFORMATION

The HD6321 Peripheral Interface Adapter provides the universal means of interfacing peripheral equipment to the HD6800 Microprocessing Unit (MPU). This device is capable of interfacing the MPU to peripherals through two 8-bit bi-directional peripheral data buses and four control lines. No external logic is required for interfacing to most peripheral devices.

The functional configuration of the PIA is programmed by the MPU during system initialization. Each of the peripheral data lines can be programmed to act as an input or output, and each of the four control/interrupt lines may be programmed for one of several control modes. This allows a high degree of flexibility in the overall operation of the interface.

HD6321P, HD3A21P, HD63B21P

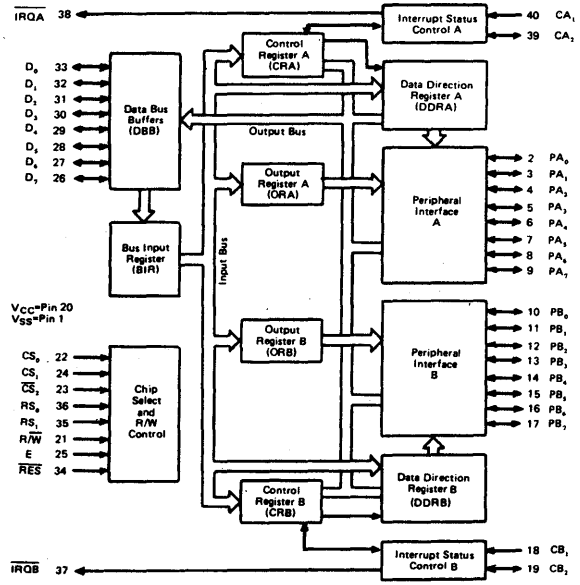


(DP-40)

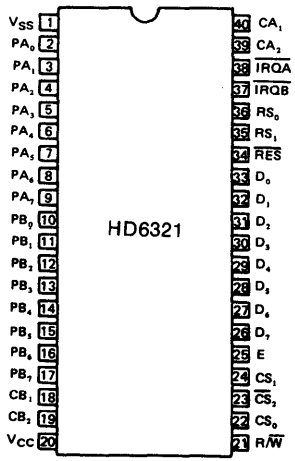
■ FEATURES

- High performance, low power CMOS process technology
- Two Bi-directional 8-Bit Peripheral Data Bus for interface to Peripheral devices
- Two Programmable Control Registers
- Two Programmable Direction Registers
- Four Individually-Controlled Interrupt Input Lines: Two Usable as Peripheral Control Outputs
- Handshake Control Logic for Input and Output Peripheral Operation
- High-Impedance 3-State and Direct Transistor Drive Peripheral Lines
- Program Controlled Interrupt and Interrupt Disable Capability
- CMOS Drive Capability on Side A Peripheral Lines
- Two TTL Drive Capability on All A and B Side Buffers
- Compatible with MC6821, MC68A21 and MC68B21

■ BLOCK DIAGRAM



■ PIN ARRANGEMENT



(Top View)

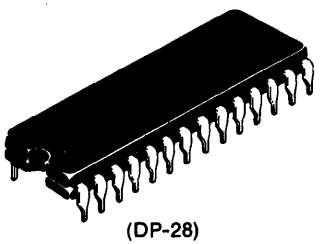
HD6340, HD63A40, HD63B40
CMOS PTM (Programmable Timer Module)

ADVANCED INFORMATION

The HD6340 is a programmable subsystem component of the HMCS6800 family designed to provide variable system time intervals.

The HD6340 has three 16-bit binary counters, three corresponding control registers and a status register. These counters are under software control and may be used to cause system interrupts and/or generate output signals. The HD6340 may be utilized for such tasks as frequency measurements, event counting, interval measuring and similar tasks. The device may be used for square wave generation, gated delay signals, single pulses of controlled duration, and pulse width modulation as well as system interrupts.

HD6340P, HD3A40P, HD63B40P

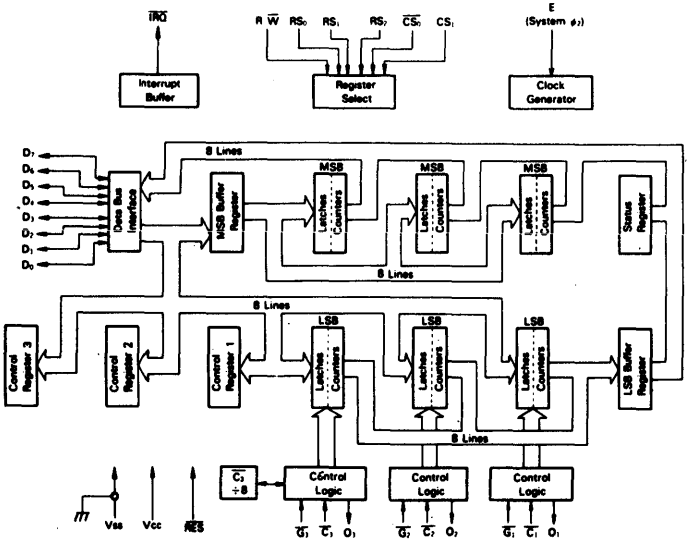


(DP-28)

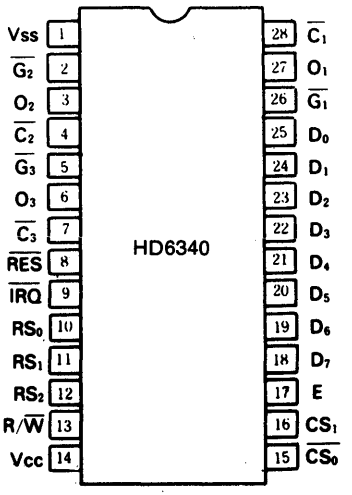
■ FEATURES

- High Performance, Low Power CMOS Process Technology
- Operates from a Single 5 Volt Power Supply
- Single System Clock Required (E)
- Selectable Prescaler on Timer 3 Capable of 4 MHz for the HD6840, 6 MHz for the HD68A40 and 8 MHz for the HD68B40
- Programmable Interrupts (IRQ) Output to MPU
- Readable Down Counter Indicates Counts to Go until Time-Out
- Selectable Gating for Frequency or Pulse-Width Comparison
- RES Input
- Three Asynchronous External Clock and Gate/Trigger Inputs Internally Synchronized
- Three Maskable Outputs
- Compatible with MC6840, MC68A40 and MC68B40

■ BLOCK DIAGRAM



■ PIN ARRANGEMENT



(Top View)

HD6350, HD63A50, HD63B50 CMOS ACIA (Asynchronous Communications Interface Adapter)

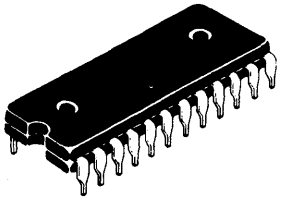
ADVANCED INFORMATION

The HD6350 Asynchronous Communications Interface Adapter provides the data formatting and control to interface serial asynchronous data communications information to bus organized systems such as the HMCS6800 Microprocessing Unit.

The bus interface of the HD6350 includes select, enable, read/write, interrupt and bus interface logic to allow data transfer over an 8-bit bi-directional data bus. The parallel data of the bus system is serially transmitted and received by the asynchronous data interface, with proper formatting and error checking.

The functional configuration of the ACIA is programmed via the data bus during system initialization. A programmable Control Register provides variable word lengths, clock division ratios, transmit control, receive control, and interrupt control. For peripheral or modem operation three control lines are provided.

HD6350P, HD63A50P, HD63B50P

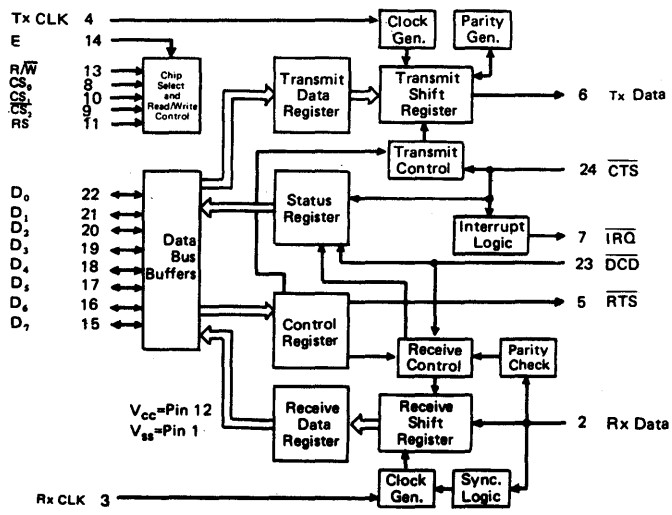


(DP-24)

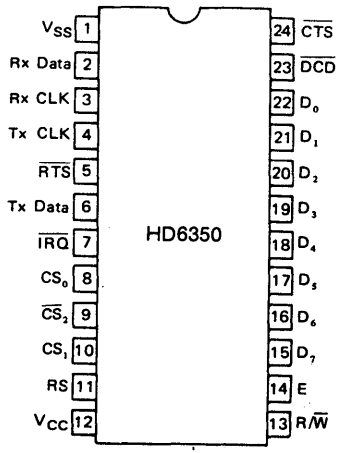
■ FEATURES

- High Performance, Low Power CMOS Process Technology
- Serial/Parallel Conversion of Data
- Eight and Nine-bit Transmission
- Insertion and Deleting of Start and Stop Bit
- Optional Even and Odd Parity
- Parity, Overrun and Framing Error Checking
- Peripheral/Modem Control Functions (Clear to Send CTS, Request to Send RTS, Data Carrier Detect (DCD)
- Optional ÷ 1, ÷ 16, and ÷ 64 Clock Modes
- Up to 500kbps Transmission
- Programmable Control Register
- Compatible with MC6850, MC68A50 and MC68B50

■ BLOCK DIAGRAM



■ PIN ARRANGEMENT



(Top View)

16-BIT MICROCOMPUTER
HMCS68000 MULTI-CHIP
SERIES

HD68000-4, HD68000-6, HD68000-8, HD68000-10
MPU (Micro Processing Unit)

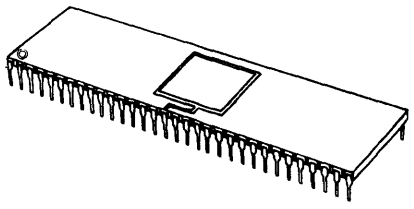
Advances in semiconductor technology have provided the capability to place on a single silicon chip a microprocessor at least an order of magnitude higher in performance and circuit complexity than has been previously available. The HD68000 is one of such VLSI microprocessors. It combines rate-of-the-art technology and advanced circuit design techniques with computer sciences to achieve an architecturally advanced 16-bit microprocessor.

The resources available to the HD68000 user consist of the following.

As shown in the programming model, the HD68000 offers seventeen 32-bit registers in addition to the 32-bit program counter and a 16-bit status register. The first eight registers (D0-D7) are used as data registers for byte (8-bit), word (16-bit), and long word (32-bit) data operations. The second set of seven registers (A0-A6) and the system stack pointer may be used as software stack pointers and base address registers. In addition, these registers may be used for word and long word address operations. All 17 registers may be used as index registers.

The specification for HD68000-10 is preliminary.

HD68000-4, HD68000-6,
HD68000-8, HD68000-10

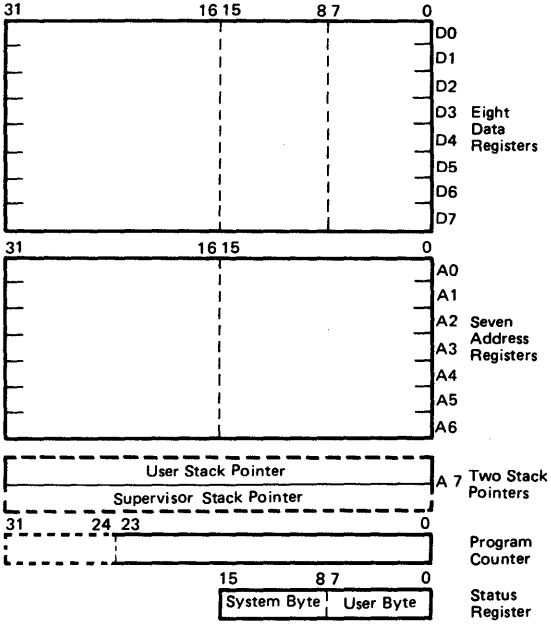


(DC-64)

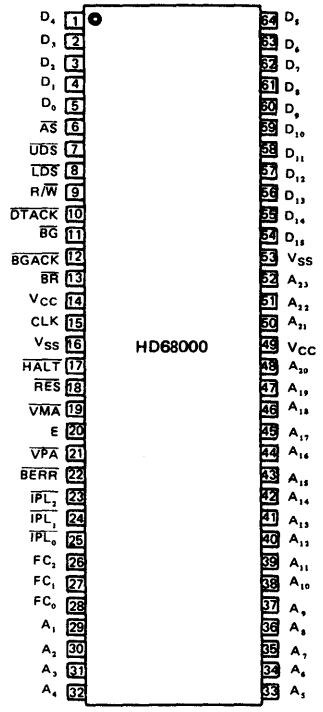
■ FEATURES

- 32-Bit Data and Address Registers
- 16 Megabyte Direct Addressing Range
- 56 Powerful Instruction Types
- Operations of Five Main Data Types
- Memory Mapped I/O
- 14 Addressing Modes
- Compatible with MC68000L4, MC68000L6, MC68000L8, and MC68000L10

■ PROGRAMMING MODEL



■ PIN ARRANGEMENT



(Top View)

This specification is applied for the mask version 68000R.
You can distinguish the mask version 68000R from the mask version 68000 by a letter R in mark pattern.
If there is a letter R in mark pattern, then the mask version is 68000R, else the mask version is 68000.

● 2A4 R
HD68000-10
JAPAN

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC}*	-0.3 ~ +7.0	V
Input Voltage	V_{in}*	-0.3 ~ +7.0	V
Operating Temperature Range	T_{opr}	0 ~ +70	°C
Storage Temperature	T_{stg}	-55 ~ +150	°C

* With respect to V_{SS} (SYSTEM GND)

(NOTE) Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ RECOMMENDED OPERATING CONDITIONS

Item	Symbol	min	typ	max	Unit
Supply Voltage	V_{CC}*	4.75	5.0	5.25	V
Input Voltage	V_{IH}*	2.0	–	V_{CC}	V
	V_{IL}*	-0.3	–	0.8	V
Operating Temperature	T_{opr}	0	25	70	°C

* With respect to V_{SS} (SYSTEM GND)

■ ELECTRICAL CHARACTERISTICS

• DC CHARACTERISTICS (V_{CC} = 5V ±5%, V_{SS} = 0V, Ta = 0 ~ +70°C, Fig. 1, 2, 3, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Input "High" Voltage		V_{IH}		2.0	–	V_{CC}	V
Input "Low" Voltage		V_{IL}		V_{SS}-0.3	–	0.8	V
Input Leakage Current	$\overline{BERR}$, $\overline{BGACK}$, $\overline{BR}$, $\overline{DTACK}$, $\overline{IPL_0}$~$\overline{IPL_2}$, VPA, CLK	I_{in}	@5.25V	–	–	2.5	μA
	$\overline{HALT}$, $\overline{RES}$			–	–	20	
Three-State (Off State) Input Current	$\overline{AS}$, A_1~A_{23}, D_0~D_{15}, $\overline{FC_0}$~FC_2, $\overline{LDS}$, R/$\overline{W}$, $\overline{UDS}$, $\overline{VMA}$	I_{TSI}	@2.4V/0.4V	–	–	20	μA
Output "High" Voltage	$\overline{AS}$, A_1~A_{23}, $\overline{BG}$, D_0~D_{15}, $\overline{FC_0}$~FC_2, $\overline{LDS}$, R/$\overline{W}$, $\overline{UDS}$, $\overline{VMA}$	V_{OH}	I_{OH} = -400μA	2.4	–	–	V
	E*			V_{CC}-0.75	–	–	
Output "Low" Voltage	$\overline{HALT}$	V_{OL}	I_{OL} = 1.6mA	–	–	0.5	V
	A_1~A_{23}, $\overline{BG}$, FC_0~FC_2		I_{OL} = 3.2mA	–	–	0.5	
	$\overline{RES}$		I_{OL} = 5.3mA	–	–	0.5	
	$\overline{AS}$, D_0~D_{15}, $\overline{LDS}$, R/$\overline{W}$, E, $\overline{UDS}$, $\overline{VMA}$		I_{OL} = 5.3mA	–	–	0.5	
Power Dissipation		P_D	f = 8MHz	–	–	1.5	W
Capacitance (Package Type Dependent)		C_{in}	V_{in} = 0V, Ta = 25°C, f = 1MHz	–	10.0	20.0	pF

* With external pull up register of 470Ω

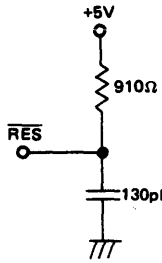


Figure 1 $\overline{RES}$ Test Load

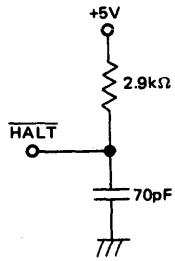
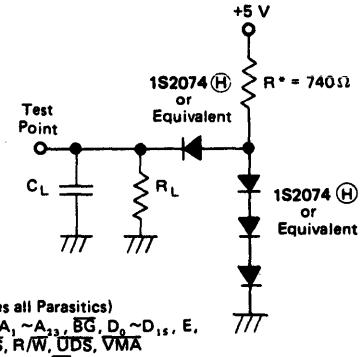


Figure 2 $\overline{HALT}$ Test Load



C_L = 130 pF (Includes all Parasitics)
R_L = 6.0 kΩ for $\overline{AS}$, A_1~A_{23}, $\overline{BG}$, D_0~D_{15}, E, FC_0~FC_2, $\overline{LDS}$, R/$\overline{W}$, $\overline{UDS}$, $\overline{VMA}$
*R = 1.22 kΩ for A_1~A_{23}, $\overline{BG}$, E, FC_0~FC_2

Figure 3 Test Loads

• AC CHARACTERISTICS (V_{CC} = 5.0V ±5%, V_{SS} = 0V, Ta = 0 ~ +70°C, unless otherwise noted.)

Number	Item	Symbol	Test Condition	4 MHz HD68000-4		6 MHz HD68000-6		8 MHz HD68000-8		10 MHz HD68000-10		Unit
				min	max	min	max	min	max	min	max	
	Frequency of Operation	f		2.0	4.0	2.0	6.0	2.0	8.0	2.0	10.0	MHz
①	Clock Period	t_{cyc}		250	500	167	500	125	500	100	500	ns
②	Clock Width Low	t_{CL}		115	250	75	250	55	250	45	250	ns
③	Clock Width High	t_{CH}		115	250	75	250	55	250	45	250	ns
④	Clock Fall Time	t_{Cf}		–	10	–	10	–	10	–	10	ns
⑤	Clock Rise Time	t_{Cr}		–	10	–	10	–	10	–	10	ns
⑥	Clock Low to Address	t_{CLAV}		–	90	–	80	–	70	–	55	ns
⑥A	Clock High to FC Valid	t_{CHFCV}		–	90	–	80	–	70	–	60	ns
⑦	Clock High to Address Data High Impedance (Maximum)	t_{CHAZx}		–	120	–	100	–	80	–	70	ns
⑧	Clock High to Address/FC Invalid (Minimum)	t_{CHAZn}		0	–	0	–	0	–	0	–	ns
⑨[1]	Clock High to $\overline{AS}$, $\overline{DS}$ Low (Maximum)	t_{CHSLx}		–	80	–	70	–	60	–	55	ns
⑩	Clock High to $\overline{AS}$, $\overline{DS}$ Low (Minimum)	t_{CHSLn}		0	–	0	–	0	–	0	–	ns
⑪[2]	Address to $\overline{AS}$, $\overline{DS}$ (Read) Low/$\overline{AS}$ Write	t_{AVSL}		55	–	35	–	30	–	20	–	ns
⑪A[2]	FC Valid to $\overline{AS}$, $\overline{DS}$, (Read) Low/$\overline{AS}$ Write	t_{FCVSL}		80	–	70	–	60	–	50	–	ns
⑫[1]	Clock Low to $\overline{AS}$, $\overline{DS}$ High	t_{CLSH}		–	90	–	80	–	70	–	55	ns
⑬[2]	$\overline{AS}$, $\overline{DS}$ High to Address/FC Invalid	t_{SHAZ}		60	–	40	–	30	–	20	–	ns
⑭[2,5]	$\overline{AS}$, $\overline{DS}$ Width Low (Read)/$\overline{AS}$ Write	t_{SL}		535	–	337	–	240	–	195	–	ns
⑭A[2]	$\overline{DS}$ Width Low (Write)	–		285	–	170	–	115	–	95	–	ns
⑮[2]	$\overline{AS}$, $\overline{DS}$ Width High	t_{SH}		285	–	180	–	150	–	105	–	ns
⑯	Clock High to $\overline{AS}$, $\overline{DS}$ High Impedance	t_{CHSZ}		–	120	–	100	–	80	–	70	ns
⑰[2]	$\overline{AS}$, $\overline{DS}$ High to R/$\overline{W}$ High	t_{SHRH}		60	–	50	–	40	–	20	–	ns
⑱[1]	Clock High to R/$\overline{W}$ High (Maximum)	t_{CHRHx}		–	90	–	80	–	70	–	60	ns
⑲	Clock High to R/$\overline{W}$ High (Minimum)	t_{CHRHn}		0	–	0	–	0	–	0	–	ns
⑳[1]	Clock High to R/$\overline{W}$ Low	t_{CHRL}	Fig. 4 ~ 7	–	90	–	80	–	70	–	60	ns
㉑[2]	Address Valid to R/$\overline{W}$ Low	t_{AVRL}		45	–	25	–	20	–	0	–	ns
㉑A[2]	FC Valid to R/$\overline{W}$ Low	t_{FCVRL}		80	–	70	–	60	–	50	–	ns
㉒[2]	R/$\overline{W}$ Low to $\overline{DS}$ Low (Write)	t_{RLSL}		200	–	140	–	80	–	50	–	ns
㉓	Clock Low to Data Out Valid	t_{CLDO}		–	90	–	80	–	70	–	55	ns
㉕[2]	$\overline{DS}$ High to Data Out Invalid	t_{SHDO}		60	–	40	–	30	–	20	–	ns
㉖[2]	Data Out Valid to $\overline{DS}$ Low (Write)	t_{DOSL}		55	–	35	–	30	–	20	–	ns
㉗[6]	Data In to Clock Low (Setup Time)	t_{DICL}		30	–	25	–	15	–	15	–	ns
㉘[2]	$\overline{AS}$, $\overline{DS}$ High to $\overline{DTACK}$ High	t_{SHDAH}		0	240	0	160	0	120	0	90	ns
㉙	$\overline{DS}$ High to Data Invalid (Hold Time)	t_{SHDI}		0	–	0	–	0	–	0	–	ns
㉚	$\overline{AS}$, $\overline{DS}$ High to $\overline{BERR}$ High	t_{SHBEH}		0	–	0	–	0	–	0	–	ns
㉛[3,6]	$\overline{DTACK}$ Low to Data In (Setup Time)	t_{DALDI}		–	180	–	120	–	90	–	65	ns
㉜	$\overline{HALT}$ and $\overline{RES}$ Input Transition Time	t_{RHrf}		0	200	0	200	0	200	0	200	ns
㉝	Clock High to $\overline{BG}$ Low	t_{CHGL}		–	90	–	80	–	70	–	60	ns
㉞	Clock High to $\overline{BG}$ High	t_{CHGH}		–	90	–	80	–	70	–	60	ns
㉟	$\overline{BR}$ Low to $\overline{BG}$ Low	t_{BRLGL}		1.5	3.0	1.5	3.0	1.5	3.0	1.5	3.0	Clk. Per.
㊱	$\overline{BR}$ High to $\overline{BG}$ High	t_{BRHGH}		1.5	3.0	1.5	3.0	1.5	3.0	1.5	3.0	Clk. Per.
㊲	$\overline{BGACK}$ Low to $\overline{BG}$ High	t_{GALGH}		1.5	3.0	1.5	3.0	1.5	3.0	1.5	3.0	Clk. Per.
㊳	$\overline{BG}$ Low to Bus High Impedance (With $\overline{AS}$ High)	t_{GLZ}		–	120	–	100	–	80	–	70	ns
㊴	$\overline{BG}$ Width High	t_{GH}		1.5	–	1.5	–	1.5	–	1.5	–	Clk. Per.
㊻	$\overline{BGACK}$ Width	t_{BGL}		1.5	–	1.5	–	1.5	–	1.5	–	Clk. Per.
㊼[6]	Asynchronous Input Setup Time	t_{ASI}		30	–	25	–	20	–	20	–	ns
㊽	$\overline{BERR}$ Low to $\overline{DTACK}$ Low (Note 3)	t_{BELDAL}		50	–	50	–	50	–	50	–	ns
㊸	Data Hold from Clock High	t_{CHDO}		0	–	0	–	0	–	0	–	ns
㊺	R/$\overline{W}$ to Data Bus Impedance Change	t_{RLDO}		55	–	35	–	30	–	20	–	ns
㊻	Halt/$\overline{RES}$ Pulse Width (Note 4)	t_{HRPW}		10	–	10	–	10	–	10	–	Clk. Per.

(to be continued)

Number	Item	Symbol	Test Condition	4 MHz HD68000-4		6 MHz HD68000-6		8 MHz HD68000-8		10 MHz HD68000-10		Unit
				min	max	min	max	min	max	min	max	
㉔	Clock High to R/$\overline{W}$, $\overline{VMA}$ High Impedance	t_{CHRZ}	Fig. 45, 46	–	120	–	100	–	80	–	70	ns
㊵	Clock Low to $\overline{VMA}$ Low	t_{CLVML}		–	90	–	80	–	70	–	70	ns
㊶	Clock Low to E Transition	t_{CLE}		–	100	–	85	–	70	–	55	ns
㊷	E Output Rise and Fall Time	t_{Erf}		–	25	–	25	–	25	–	25	ns
㊸	$\overline{VMA}$ Low to E High	t_{VMLEH}		325	–	240	–	200	–	150	–	ns
㊹	$\overline{AS}$, $\overline{DS}$ High to $\overline{VPA}$ High	t_{SHVPH}		0	240	0	160	0	120	0	90	ns
㊺	E Low to Address/$\overline{VMA}$/FC Invalid	t_{ELAI}		55	–	35	–	30	–	10	–	ns
㊾	E Low to $\overline{AS}$, $\overline{DS}$ Invalid	t_{ELSI}		-80	–	-80	–	-80	–	-80	–	ns
㊿	E Width High	t_{EH}		900	–	600	–	450	–	350	–	ns
(51)	E Width Low	t_{EL}		1400	–	900	–	700	–	550	–	ns
(52)	E Extended Rise Time	t_{CIEHX}		80	–	80	–	80	–	80	–	ns
(54)	Data Hold from E Low (Write)	t_{ELDOZ}		60	–	40	–	30	–	20	–	ns

(NOTES) 1. For a loading capacitance of less than or equal to 50 picofarads, subtract 5 nanoseconds from the values given in these columns.
2. Actual value depends on clock period.
3. If #47 is satisfied for both $\overline{DTACK}$ and $\overline{BERR}$, #48 may be 0 ns.
4. After V_{CC} has been applied for 100 ms.
5. For the mask version 68000 #14 and #14A are one clock period less than the given number.
6. If the asynchronous setup time (#47) requirements are satisfied, the $\overline{DTACK}$ low-to-data setup time (#31) requirement can be ignored. The data must only satisfy the data-in to clock-low setup time (#27) for the following cycle.

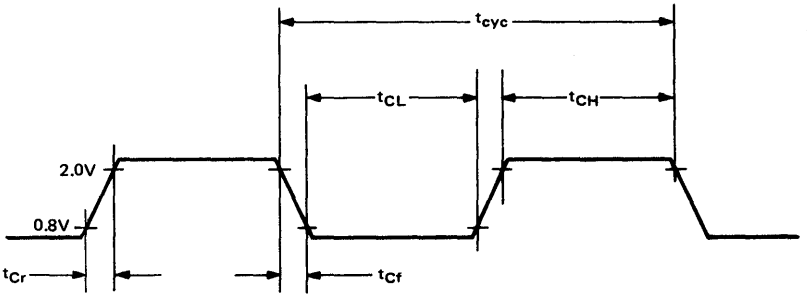
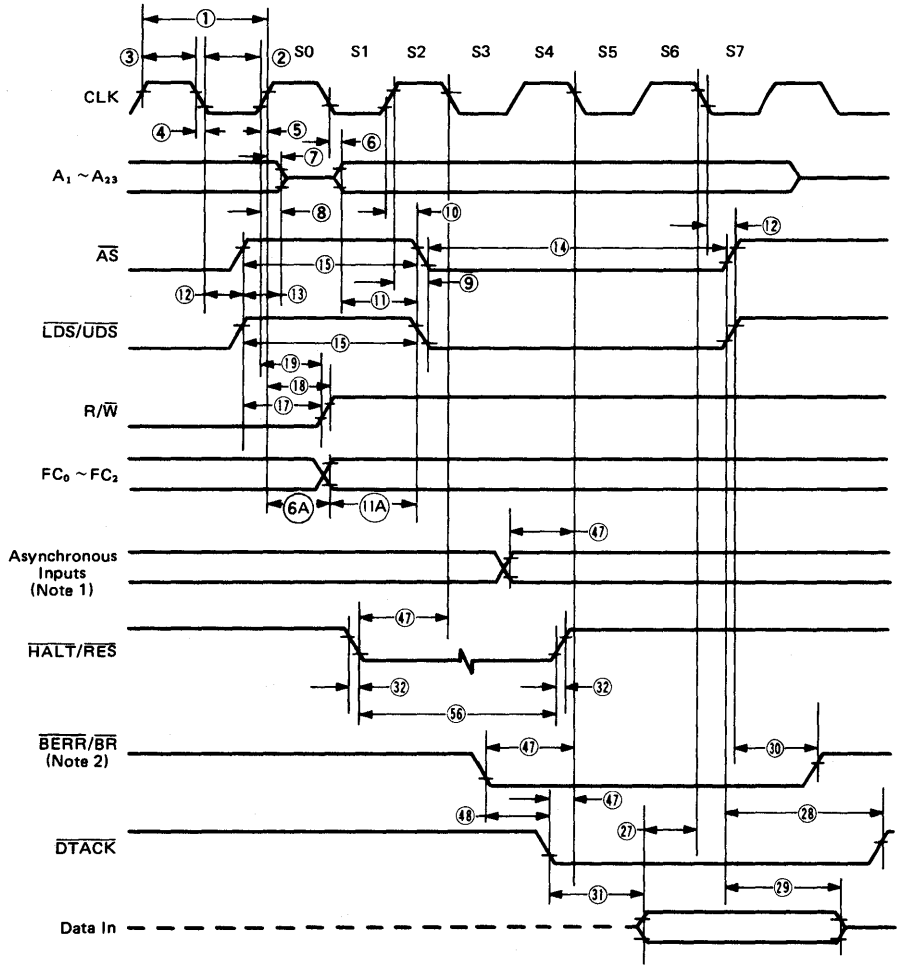
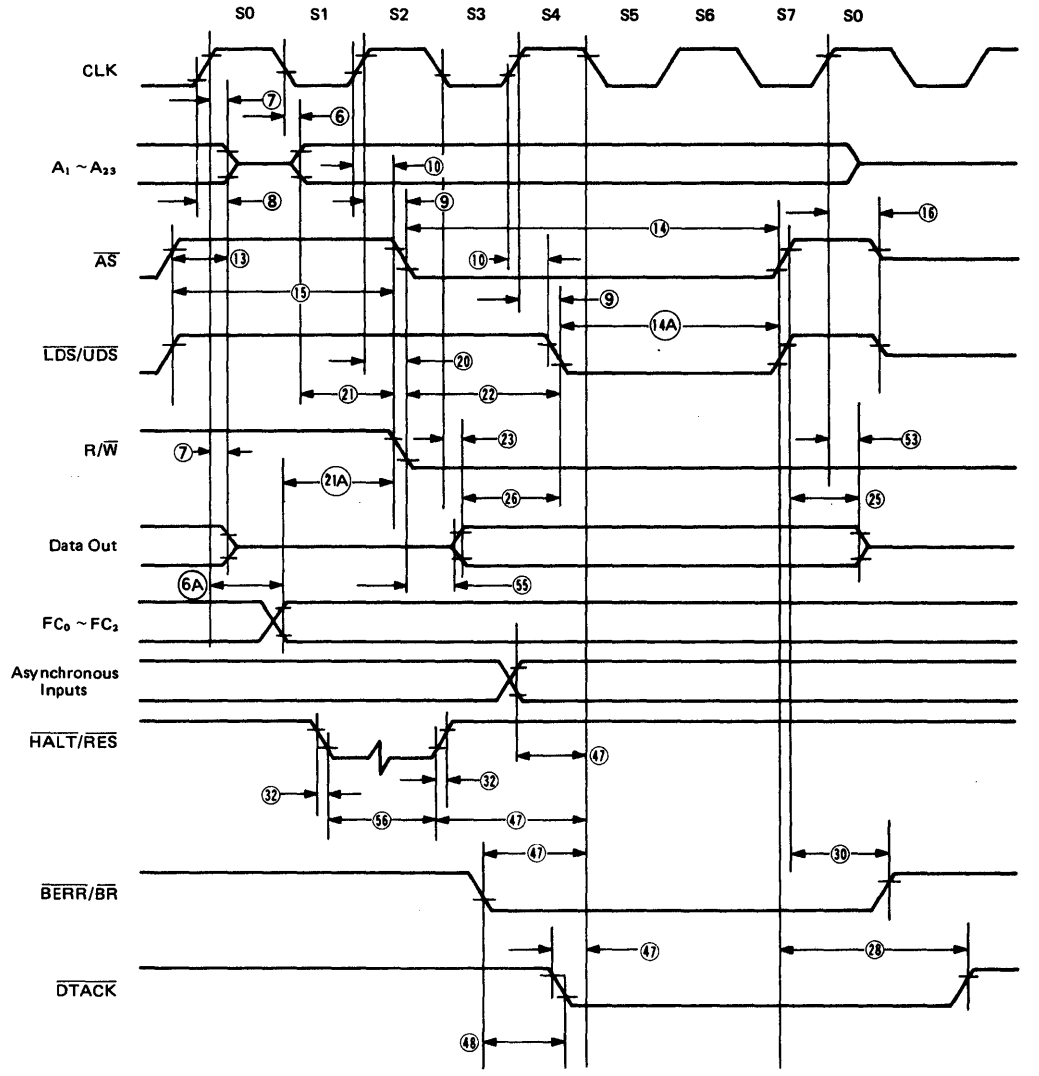


Figure 4 Input Clock Waveform



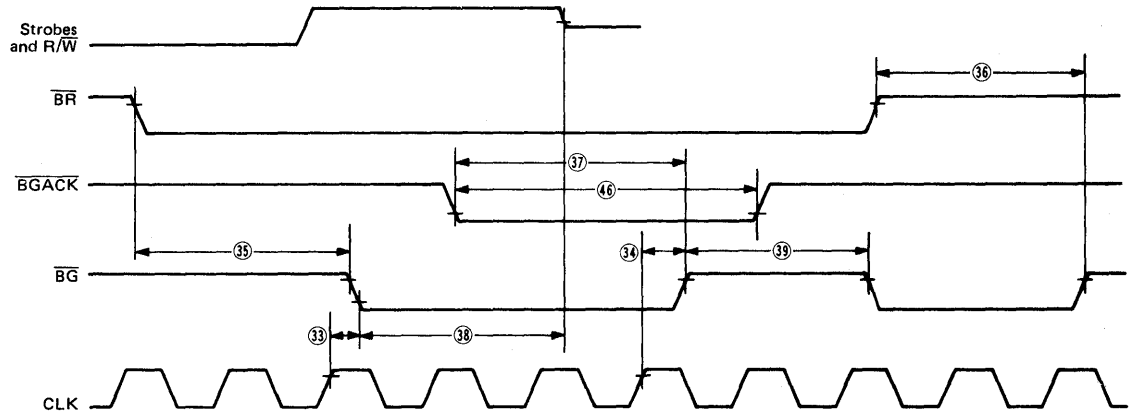
(NOTES) 1. Setup time for the asynchronous inputs $\overline{BGACK}$, $\overline{IPL_0} \sim \overline{IPL_2}$ and $\overline{VPA}$ guarantees their recognition at the next falling edge of the clock.
2. $\overline{BR}$ need fall at this time only in order to insure being recognized at the end of this bus cycle.
3. Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 2.0 volts, unless otherwise noted.

Figure 5 Read Cycle Timing



(NOTE) Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 2.0 volts, unless otherwise noted.

Figure 6 Write Cycle Timing



(NOTES) 1. Setup time for the asynchronous inputs $\overline{BERR}$, $\overline{BGACK}$, $\overline{BR}$, $\overline{DTACK}$, $\overline{IPL_0} \sim \overline{IPL_2}$, and $\overline{VPA}$ guarantees their recognition at the next falling edge of the clock.
2. Waveform measurements for all inputs and outputs are specified at: logic high = 2.0 volts, logic low = 0.8 volts.
3. These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input an output signals. Refer to other functional descriptions and their related diagrams for device operation.

Figure 7 AC Electrical Waveforms — Bus Arbitration

■ SIGNAL DESCRIPTION

The following paragraphs contain a brief description of the input and output signals. A discussion of bus operation during the various machine cycles and operations is also given.

• SIGNAL DESCRIPTION

The input and output signals can be functionally organized into the groups shown in Figure 8. The following paragraphs provide a brief description of the signals and also a reference (if applicable) to other paragraphs that contain more detail about the function being performed.

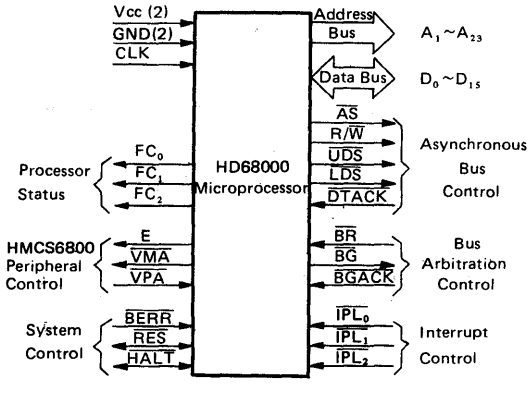


Figure 8 Input and Output Signals

ADDRESS BUS (A_1 through A_{23})

This 23-bit, unidirectional, three-state bus is capable of addressing 8 megawords of data. It provides the address for bus operation during all cycles except interrupt cycles. During interrupt cycles, address lines A_1, A_2, and A_3 Provide information about what level interrupt is being serviced while address lines A_4 through A_{23} are all set to a logic high.

DATA BUS (D_0 through D_{15})

This 16-bit, bidirectional, three-state bus is the general purpose data path. It can transfer and accept data in either word or byte length. During an interrupt acknowledge cycle, an external device supplies the vector number on data lines $D_0 \sim D_7$.

ASYNCHRONOUS BUS CONTROL

Asynchronous data transfer are handled using the following control signals: address strobe, read/write, upper and lower data strobes, and data transfer acknowledge. These signals are explained in the following paragraphs.

Address Strobe ($\overline{AS}$)

This signal indicates that there is a valid address on the address bus.

Read/Write (R/$\overline{W}$)

This signal defines the data bus transfer as a read or write cycle. The R/$\overline{W}$ signal also works in conjunction with the upper and lower data strobes as explained in the following paragraph.

Upper and Lower Data Strobes ($\overline{UDS}$, $\overline{LDS}$)

These signals control the data on the data bus, as shown in Table 1. When the R/$\overline{W}$ line is high, the processor will read from the data bus as indicated. When the R/$\overline{W}$ line is low, the processor will write to the data bus as shown.

Table 1 Data Strobe Control of Data Bus

$\overline{UDS}$	$\overline{LDS}$	R/$\overline{W}$	$D_8 \sim D_{15}$	$D_0 \sim D_7$
High	High	–	No valid data	No valid data
Low	Low	High	Valid data bits $8 \sim 15$	Valid data bits $0 \sim 7$
High	Low	High	No valid data	Valid data bits $0 \sim 7$
Low	High	High	Valid data bits $8 \sim 15$	No valid data
Low	Low	Low	Valid data bits $8 \sim 15$	Valid data bits $0 \sim 7$
High	Low	Low	Valid data bits $0 \sim 7$*	Valid data bits $0 \sim 7$
Low	High	Low	Valid data bits $8 \sim 15$	Valid data bits $8 \sim 15$*

* These conditions are a result of current implementation and may not appear on future devices.

Data Transfer Acknowledge ($\overline{DTACK}$)

This input indicates that the data transfer is completed. When the processor recognizes $\overline{DTACK}$ during a read cycle, data is latched and the bus cycle terminated. When $\overline{DTACK}$ is recognized during a write cycle, the bus cycle is terminated.

An active transition of data transfer acknowledge, $\overline{DTACK}$, indicates the termination of a data transfer on the bus.

If the system must run at a maximum rate determined by RAM access times, the relationship between the times at which $\overline{DTACK}$ and DATA are sampled are important.

All control and data lines are sampled during the HD68000's clock high time. The clock is internally buffered, which results in some slight differences in the sampling and recognition of various signals. HD68000 allow $\overline{BERR}$ or $\overline{DTACK}$ to be recognized in S4, S6, etc., which terminates the cycle*. The $\overline{DTACK}$ signal, like other control signals, is internally synchronized to allow for valid operation in an asynchronous system. If the required setup time (#47) is met during S4, $\overline{DTACK}$ will be recognized during S5 and S6, and data will be captured during S6. The data must meet the required setup time (#27).

If an asynchronous control signal does not meet the required setup time, it is possible that it may not be recognized during that cycle. Because of this, asynchronous systems must not allow $\overline{DTACK}$ to precede data by more than parameter #31.

Asserting $\overline{DTACK}$ (or $\overline{BERR}$) on the rising edge of a clock (such as S4) after the assertion of address strobe will allow a HD68000 system to run at its maximum bus rate. If setup times #27 and #47 are guaranteed, #31 may be ingnored.

* The mask version 68000 allowed $\overline{DTACK}$ to be recognized as early as S2 (bus state 2).

BUS ARBITRATION CONTROL

These three signals form a bus arbitration circuit to determine which device will be the bus master device.

Bus Request ($\overline{BR}$)

This input is wire ORed with all other devices that could be bus masters. This input indicates to the processor that some other device desires to become the bus master.

Bus Grant ($\overline{BG}$)

This output indicates to all other potential bus master devices that the processor will release bus control at the end of the current bus cycle.

Bus Grand Acknowledge ($\overline{BGACK}$)

This input indicates that some other device has become the bus master. This signal cannot be asserted until the following four conditions are met:

(1) A Bus Grant has been received
(2) Address Strobe is inactive which indicates that the microprocessor is not using the bus
(3) Data Transfer Acknowledge is inactive which indicates that neither memory nor peripherals are using the bus
(4) Bus Grant Acknowledge is inactive which indicates that no other device is still claiming bus mastership.

INTERRUPT CONTROL ($\overline{IPL_0}$, $\overline{IPL_1}$, $\overline{IPL_2}$)

These input pins indicate the encoded priority level of the device requesting an interrupt. Level seven is the highest priority while level zero indicates that no interrupts are requested. The least significant bit is given in $\overline{IPL_0}$ and the most significant bit is contained in $\overline{IPL_2}$.

SYSTEM CONTROL

The system control inputs are used to either reset or halt the processor and to indicate to the processor that bus errors have occurred. The three system control inputs are explained in the following paragraphs.

Bus Error ($\overline{BERR}$)

This input informs the processor that there is a problem with the cycle currently being executed. Problems may be a result of:

(1) Nonresponding devices
(2) Interrupt vector number acquisition failure
(3) Illegal access request as determined by a memory management unit
(4) Other application dependent errors.

The bus error signal interacts with the halt signal to determine if exception processing should be performed or the current bus cycle should be retried.

Refer to **BUS ERROR AND HALT OPERATION** paragraph for additional information about the interaction of the bus error and halt signals.

Reset ($\overline{RES}$)

This bidirectional signal line acts to reset (initiate a system initialization sequence) the processor in response to an external reset signal. An internally generated reset (result of a RESET instruction) causes all external devices to be reset and the internal state of the processor is not affected. A total system reset (processor and external devices) is the result of external HALT and RESET signals applied at the same time. Refer to **RESET OPERATION** paragraph for additional information about reset operation.

Halt ($\overline{HALT}$)

When this bidirectional line is driven by an external device,

it will cause the processor to stop at the completion of the current bus cycle. When the processor has been halted using this input, all control signals are inactive and all three-state lines are put in their high-impedance state. Refer to **BUS ERROR AND HALT OPERATION** paragraph for additional information about the interaction between the halt and bus error signals.

When the processor has stopped executing instructions, such as in a double bus fault condition, the halt line is driven by the processor to indicate to external devices that the processor has stopped.

HMCS6800 PERIPHERAL CONTROL

These control signals are used to allow the interfacing of synchronous HMCS6800 peripheral devices with the asynchronous HD68000. These signals are explained in the following paragraphs.

Enable (E)

This signal is the standard enable signal common to all HMCS6800 type peripheral devices. The period for this output is ten HD68000 clock periods (six clocks low; four clocks high).

Valid Peripheral Address ($\overline{VPA}$)

This input indicates that the device or region addressed is a HMCS6800 family device and that data transfer should be synchronized with the enable (E) signal. This input also indicates that the processor should use automatic vectoring for an interrupt. Refer to **INTERFACE WITH HMCS6800 PERIPHERALS.**

Valid Memory Address ($\overline{VMA}$)

This output is used to indicate to HMCS6800 peripheral devices that there is a valid address on the address bus and the processor is synchronized to enable. This signal only responds to a valid peripheral address ($\overline{VPA}$) input which indicates that the peripheral is a HMCS6800 family device.

PROCESSOR STATUS (FC_0, FC_1, FC_2)

These function code outputs indicate the state (user or supervisor) and the cycle type currently being executed, as shown in Table 2. The information indicated by the function code outputs is valid whenever address strobe ($\overline{AS}$) is active.

Table 2 Function Code Outputs

FC_2	FC_1	FC_0	Cycle Type
Low	Low	Low	(Undefined, Reserved)
Low	Low	High	User Data
Low	High	Low	User Program
Low	High	High	(Undefined, Reserved)
High	Low	Low	(Undefined, Reserved)
High	Low	High	Superviser Data
High	High	Low	Supervisor Program
High	High	High	Interrupt Acknowledge

CLOCK (CLK)

The clock input is a TTL-compatible signal that is internally buffered for development of the internal clocks needed by the processor. The clock input shall be a constant frequency.

SIGNAL SUMMARY

Table 3 is a summary of all the signals discussed in the previous paragraphs.

Table 3 Signal Summary

Signal Name	Mnemonic	Input/Output	Active State	Three State
Address Bus	$A_1 \sim A_{23}$	output	high	yes
Data Bus	$D_0 \sim D_{15}$	input/output	high	yes
Address Strobe	$\overline{AS}$	output	low	yes
Read/Write	R/$\overline{W}$	output	read-high write-low	yes
Upper and Lower Data Strobes	$\overline{UDS}$, $\overline{LDS}$	output	low	yes
Data Transfer Acknowledge	$\overline{DTACK}$	input	low	no
Bus Request	$\overline{BR}$	input	low	no
Bus Grant	$\overline{BG}$	output	low	no
Bus Grant Acknowledge	$\overline{BGACK}$	input	low	no
Interrupt Priority Level	$\overline{IPL_0}$, $\overline{IPL_1}$, $\overline{IPL_2}$	input	low	no
Bus Error	$\overline{BERR}$	input	low	no
Reset	$\overline{RES}$	input/output	low	no*
Halt	$\overline{HALT}$	input/output	low	no*
Enable	E	output	high	no
Valid Memory Address	$\overline{VMA}$	output	low	yes
Valid Peripheral Address	$\overline{VPA}$	input	low	no
Function Code Output	FC_0, FC_1, FC_2	output	high	yes
Clock	CLK	input	high	no
Power Input	V_{CC}	input	–	–
Ground	V_{SS}	input	–	–

* Open drain

■ REGISTER DESCRIPTION AND DATA ORGANIZATION

The following paragraphs describe the registers and data organization of the HD68000.

• OPERAND SIZE

Operand sizes are defined as follows: a byte equals 8 bits, a word equals 16 bits, and a long word equals 32 bits. The operand size for each instruction is either explicitly encoded in the instruction or implicitly defined by the instruction operation. All explicit instructions support byte, word or long word operands. Implicit instructions support some subset of all three sizes.

• DATA ORGANIZATION IN REGISTERS

The eight data registers support data operands of 1, 8, 16, or 32 bits. The seven address registers together with the active stack pointer support address operands of 32 bits.

DATA REGISTERS

Each data register is 32 bits wide. Byte operands occupy the low order 8 bits, word operands the low order 16 bits, and long word operands the entire 32 bits. The least significant bit is addressed as bit zero; the most significant bit is addressed as bit 31.

When a data register is used as either a source or destination operand, only the appropriate low-order portion is changed; the remaining high-order portion is neither used nor changed.

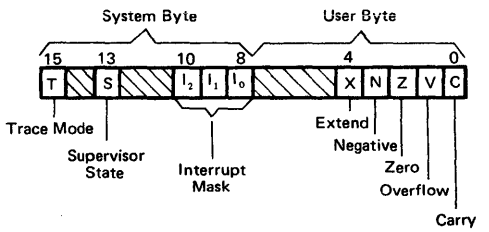
ADDRESS REGISTERS

Each address register and the stack pointer is 32 bits wide and holds a full 32 bit address. Address registers do not support byte sized operands. Therefore, when an address register is used as a source operand, either the low order word or the entire long word operand is used depending upon the operation size. When an address register is used as the destination operand, the entire register is affected regardless of the operation size. If the operation size is word, any other operands are sign extended to 32 bits before the operation is performed.

• STATUS REGISTER

The status register contains the interrupt mask (eitht levels available) as well as the condition codes; extend (X), negative (N), zero (Z), overflow (V), and carry (C). Additional status bits indicate that the processor is in a trace (T) mode and/or in a supervisor (S) state.

Status Register



• DATA ORGANIZATION IN MEMORY

Bytes are individually addressable with the high order byte having an even address the same as the word, as shown in Figure 9. The low order byte has an odd address that is one count higher than the word address. Instructions and multibyte data are accessed only on word (even byte) boundaries. If a long word datum is located at address n (n even), then the second word of that datum is located at address n + 2.

The data types supported by the HD68000 are: bit data, integer data of 8, 16, or 32 bits, 32-bit addresses and binary coded decimal data. Each of these data types is put in memory, as shown in Figure 10.

■ BUS OPERATION

The following paragraphs explain control signal and bus operation during data transfer operations, bus arbitration, bus error and halt conditions, and reset operation.

• DATA TRANSFER OPERATIONS

Transfer of data between devices involve the following leads:

(1) Address Bus A_1 through A_{23}
(2) Data Bus D_0 through D_{15}
(3) Control Signals

The address and data buses are separate parallel buses used to transfer data using an asynchronous bus structure. In all cycles, the bus master assumes responsibility for deskewing all signals it issues at both the start and end of a cycle. In addition, the bus master is responsible for deskewing the acknowledge and data signals from the slave device.

The following paragraphs explain the read, write, and read-modify-write cycles. The indivisible read-modify-write cycle is the method used by the HD68000 for interlocked multiprocessor communications.

(NOTE) The terms **assertion** and **negation** will be used extensively. This is done to avoid confusion when dealing with a mixture of "active-low" and "active-high" signals. The term assert or assertion is used to indicate that a signal is active or true independent of whether that voltage is low or high. The term negate or negation is used to indicate that a signal is inactive or false.

Read Cycle

During a read cycle, the processor receives data from memory or a peripheral device. The processor reads bytes of data in all cases. If the instruction specifies a word (or double word) operation, the processor reads both bytes. When the instruction specifies byte operation, the processor uses an internal A_0 bit to determine which byte to read and then issues the data strobe required for that byte. For bytes operations, when the A_0 bit equals zero, the upper data strobe is issued. When the A_0 bit equals one, the lower data strobe is issued. When the data is received, the processor correctly positions it internally.

A word read cycle flow chart is given in Figure 11. A byte read cycle flow chart is given in Figure 12. Read cycle timing is given in Figure 13. Figure 14 details word and byte read cycle operations. Refer to these illustrations during the following detailed.

At state zero (S0) in the read cycle, the address bus (A_1 through A_{23}) is in the high impedance state. A function code is asserted on the function code output line (FC_0 through FC_2). The read/write (R/W) signal is switched high to indicate a read cycle. One half clock cycle later, at state 1, the address bus is released from the high impedance state. The function code outputs indicate which address space that this cycle will operate on.

In state 2, the address strobe ($\overline{AS}$) is asserted to indicate that there is a valid address on the address bus and the upper and lower data strobe ($\overline{UDS}$, $\overline{LDS}$) is asserted as required. The memory or peripheral device uses the address bus and the address strobe to determine if it has been selected. The selected device uses the read/write signal and the data strobe to place its information on the data bus. Concurrent with placing data on the data bus, the selected device asserts data transfer acknowledge ($\overline{DTACK}$). No new control signals are issued during states 3 and 4.

Data transfer acknowledge must be present at the processor at the start of state 5 or the processor will substitute wait states for states 5 and 6. State 5 starts the synchronization of the returning data transfer acknowledge. The bus interface circuitry issues request for subsequent internal cycles during state 6. At the end of state 6 (beginning of state 7) incoming data is latched into an internal data bus holding register.

During state 7, address strobe and the upper and/or lower data strobes are negated. The address bus is held valid through state 7 to allow for static memory operation and signal skew. The read/write signal and the function code outputs also remain valid through state 7 to ensure a correct transfer operation. The slave device keeps its data asserted until it detects the negation of either the address strobe or the upper and/or lower data strobe. The slave device must remove its data and data transfer acknowledge within one clock period of recognizing the negation of the address or data strobes. Note that the data bus might not become free and data transfer acknowledge might not be removed until state 0 or 1.

When address strobe is negated, the slave device is released. Note that a slave device must remain selected as long as address strobe is asserted to ensure the correct functioning of the read-modify-write cycle.

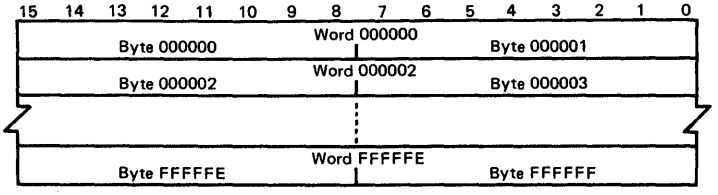


Figure 9 Word Organization in Memory

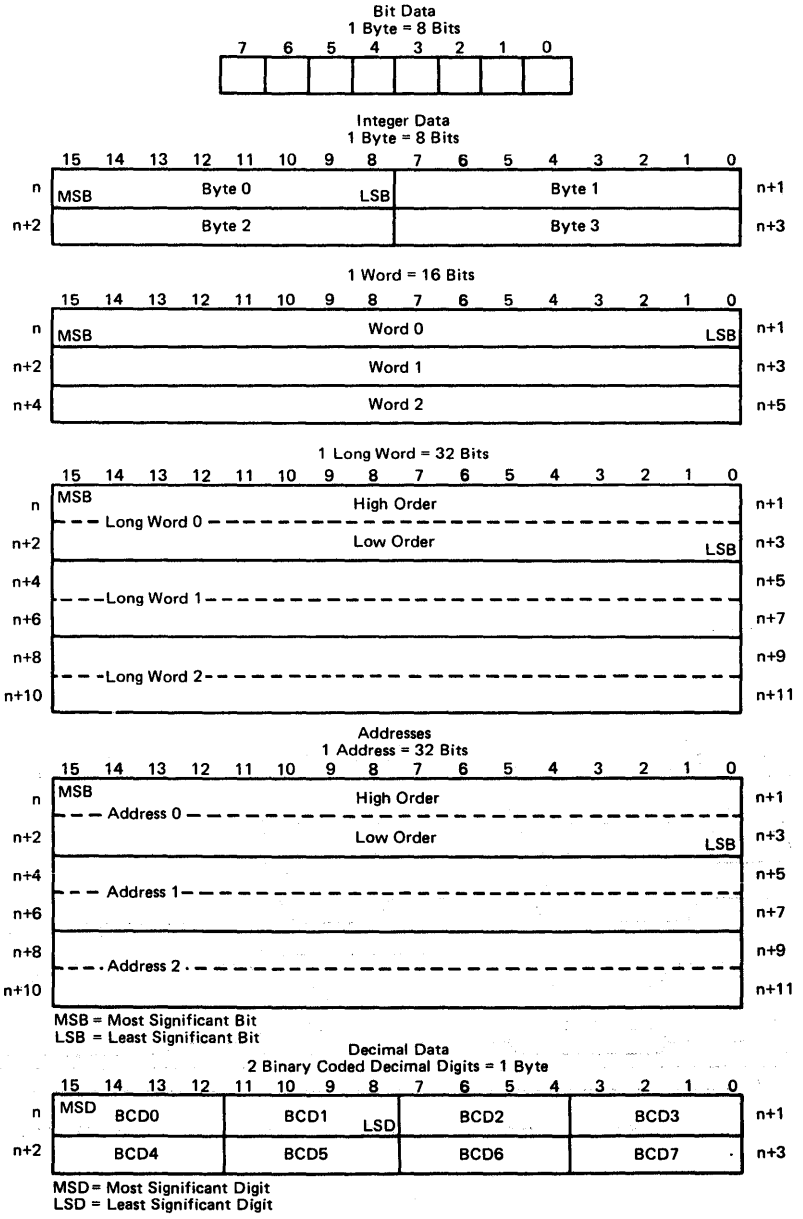


Figure 10 Data Organization in Memory

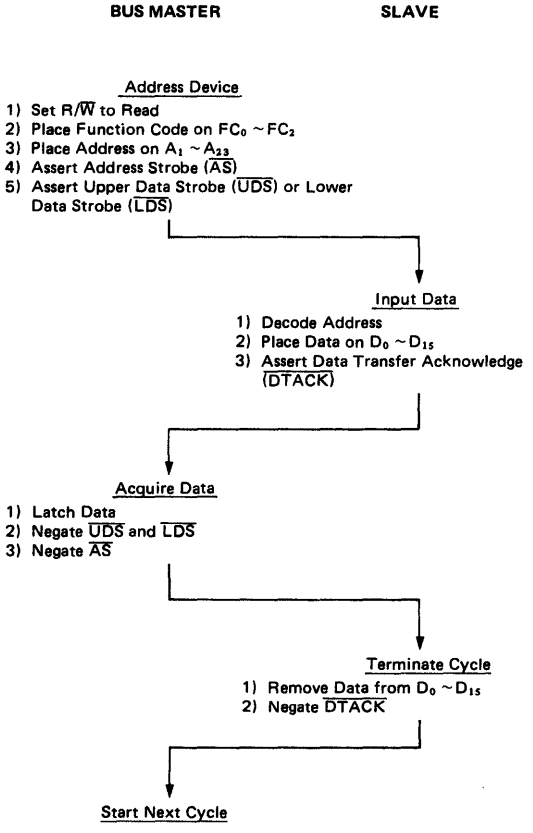


Figure 11 Word Read Cycle Flow Chart

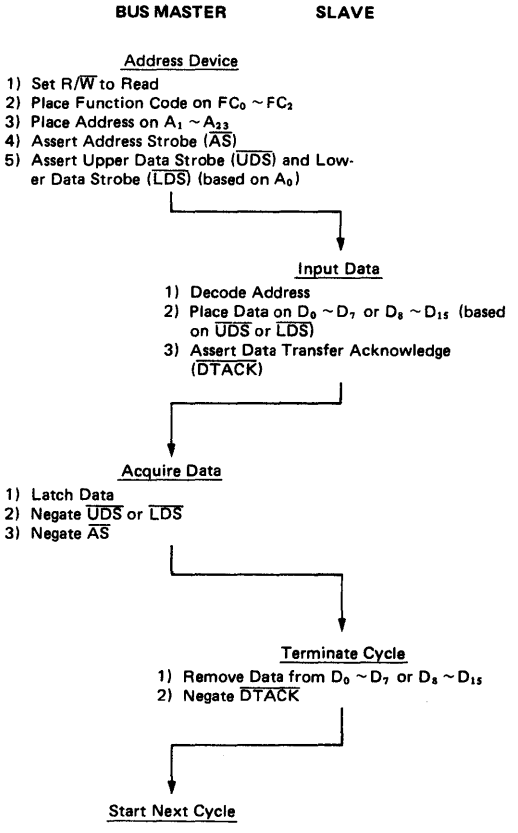


Figure 12 Byte Read Cycle Flow Chart

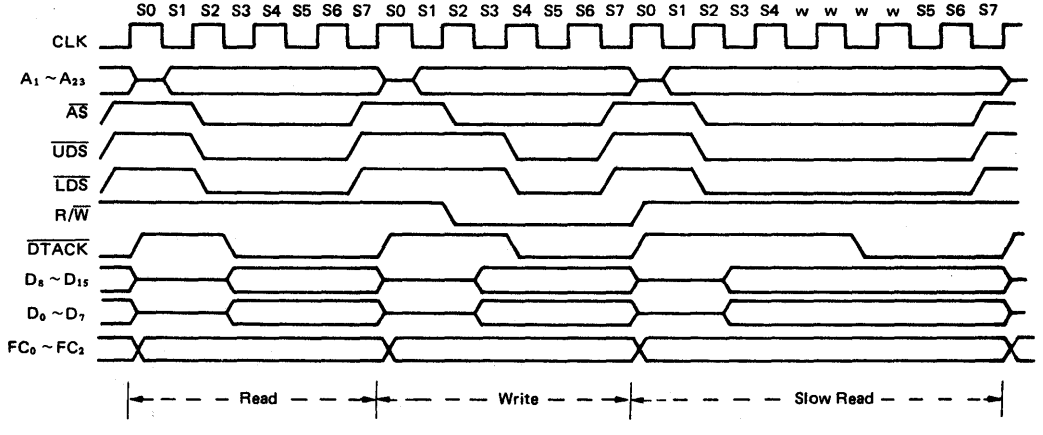


Figure 13 Read and Write Cycle Timing Diagram

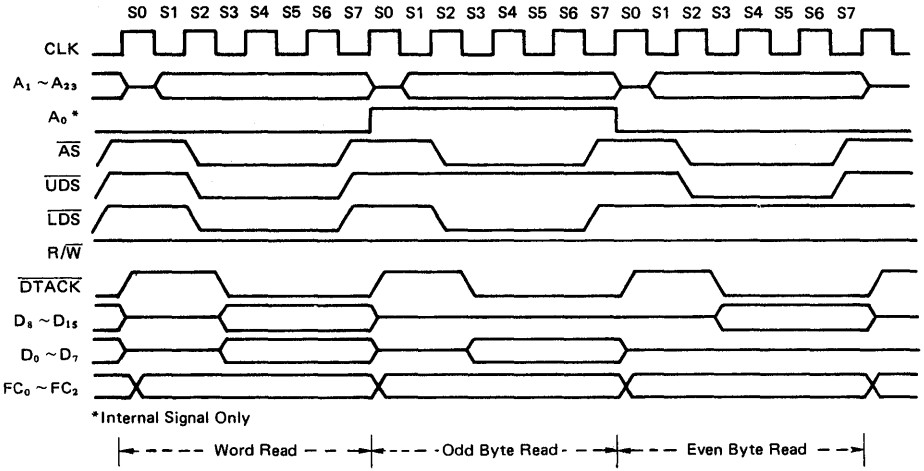


Figure 14 Word and Byte Read Cycle Timing Diagram

Write Cycle

During a write cycle, the processor sends data to memory or a peripheral device. The processor writes bytes of data in all cases. If the instruction specifies a word operation, the processor writes both bytes. When the instruction specifies a byte operation, the processor uses an internal A_0 bit to determine which byte to write and then issues the data strobe required for that byte. For byte operations, when the A_0 bit equals zero, the upper data strobe is issued. When the A_0 bit equals one, the lower data strobe is issued. A word write cycle flow chart is given in Figure 15. A byte write cycle flow chart is given in Figure 16. Write cycle timing is given in Figure 13. Figure 17 details word and byte write cycle operation. Refer to these illustrations during the following detailed discussion.

At state zero (S0) in the write cycle, the address bus (A_1 through A_{23}) is in the high impedance state. A function code is asserted on the function code output line (FC_0 through FC_2).

(NOTE) The read/write (R/$\overline{W}$) signal remains high until state 2 to prevent bus conflicts with preceding read cycles. The data bus is not driven until state 3.

One half clock later, at state 1, the address bus is released from the high impedance state. The function code outputs indicate which address space that this cycle will operate on.

In state 2, the address strobe ($\overline{AS}$) is asserted to indicate that there is a valid address on the address bus. The memory or peripheral device uses the address bus and the address strobe to determine if it has been selected. During state 2, the read/write signal is switched low to indicate a write cycle. When external processor data bus buffers are required, the read/write line provides sufficient directional control. Data is not asserted during this state to allow sufficient turn around time for external data buffers (if used). Data is asserted onto the data bus during state 3.

In state 4, the data strobes are asserted as required to indicate that the data bus is stable. The selected device uses the read/write signal and the data strobes to take its information from the data bus. The selected device asserts data transfer acknowledge ($\overline{DTACK}$) when it has successfully stored the data.

Data transfer acknowledge must be present at the processor at the start of state 5 or the processor will substitute wait states for states 5 and 6. State 5 starts the synchronization of the returning data transfer acknowledge. The bus interface circuitry issues requests for subsequent internal cycles during state 6.

During state 7, address strobe and the upper and/or lower data strobes are negated. The address and data buses are held valid through state 7 to allow for static memory operation and signal skew. The read/write signal and the function code outputs also remain valid through state 7 to ensure a correct transfer operation. The slave device keeps its data transfer acknowledge asserted until it detects the negation of either the address strobe or the upper and/or lower data strobe. The slave device must remove its data transfer acknowledge within one clock period after recognizing the negation of the address or data strobes. Note that the processor releases the data bus at the end of state 7 but that data transfer acknowledge might not be removed until state 0 or 1. When address strobe is negated, the slave device is released.

Read-Modify-Write Cycle

The read-modify-write cycle performs a read, modifies the data in the arithmetic-logic unit, and writes the data back to the same address. In the HD68000 this cycle is indivisible in that the address strobe is asserted throughout the entire cycle. The test and set (TAS) instruction uses this cycle to provide meaningful communication between processors in a multiple processor environment. This instruction is the only instruction that uses the read-modify-write cycle and since the test and set instruction only operates on bytes, all read-modify-write cycles are byte operations. A read-modify-write cycle flow chart is given in Figure 18 and a timing diagram is given in Figure 19. Refer to these illustrations during the following detailed discus-

sions.

At state zero (S0) in the read-modify-write cycle, the address bus (A_1 through A_{23}) is in the high impedance state. A function code is asserted on the function code output line (FC_0 through FC_2). The read/write (R/$\overline{W}$) signal is switched high to indicate a read cycle. One half clock cycle later, at state 1, the address bus is released from the high impedance state. The function code outputs indicate which address space that this cycle will operate on.

In state 2, the address strobe (AS) is asserted to indicate that there is a valid address on the address bus and the upper or lower data strobe ($\overline{UDS}$, $\overline{LDS}$) is asserted as required. The memory or peripheral device uses the address bus and the address strobe to determine if it has been selected. The selected device uses the read/write signal and the data strobe to place its information on the data bus. Concurrent with placing data on the data bus, the selected device asserts data transfer acknowledge ($\overline{DTACK}$). No new control signals are issued during states 3 and 4.

Data transfer acknowledge must be present at the processor at the start of state 5 or the processor will substitute wait states for states 5 and 6. State 5 starts the synchronization of the returning data transfer acknowledge. The bus interface circuitry issues requests for subsequent internal cycles during state 6. At the end of state 6 (beginning of state 7) incoming data is latched into an internal data bus holding register.

During state 7, the upper or lower data strobe is negated. The address bus, address strobe, read/write signal, and function code outputs remain as they were in preparation for the write portion of the cycle. The slave device keeps its data asserted until it detects the negation of the upper or lower data strobe. The slave device must remove its data and data transfer acknowledge within one clock period of recognizing the negation of the data strobes. No new control signals are issued during state 8. Internal modification of data may occur at this time.

(NOTE) The read/write signal remains high until state 14 to prevent bus conflicts with the preceding read portion of the cycle and the data bus is not asserted by the processor until state 15.

No new control signals are issued during state 9, state 10, state 11, state 12 and state 13.

In state 14, the read/write signal is switched low to indicate a write cycle. When external processor data bus buffers are required, the read/write line provides sufficient directional control. Data is not asserted during this state to allow sufficient turn around time for external data buffers (if used). Data is asserted onto the data bus during state 15.

In state 16, the data strobe is asserted as required to indicate that the data bus is stable. The selected device uses the read/write signal and the data strobe to take its information from the data bus. The selected device asserts data transfer acknowledge ($\overline{DTACK}$) when it has successfully stored its data. No new control signals are issued during states 17 and 18.

Data transfer acknowledge must be present at the processor at the start of state 17 or the processor will substitute wait states for states 17 and 18. State 17 starts the synchronization of the returning data transfer acknowledge for the write portion of the cycle. The bus interface circuitry issues requests for subsequent internal cycles during state 18.

During state 19, address strobe and the upper or lower data strobe is negated. The address and data buses are held valid through state 19 to allow for static memory operation and signal skew. The read/write signal and the function code outputs also remain valid through state 19 to ensure a correct transfer operation. The slave device keeps its data transfer acknowledge asserted until it detects the negation of either the address strobe or the upper or lower data strobe. The slave device must remove its data transfer acknowledge within once clock period after recognizing the negation of the address or data strobes. Note that the processor releases the data bus at the end of state 19 but that data transfer acknowledge might not be removed until state 0 or 1. When address strobe is negated the slave device is released.

• BUS ARBITRATION

Bus arbitration is a technique used by master-type devices to request, be granted, and acknowledge bus mastership. In its simples form, it consists of:

(1) Asserting a bus mastership request.
(2) Receiving a grant that the bus is available at the end of the current cycle.
(3) Acknowledging that mastership has been assumed.

Figure 20 is a flow chart showing the detail involved in a request from a single device. Figure 21 is a timing diagram for the same operations. This technique allows processing of bus requests during data transfer cycles.

The timing diagram shows that the bus request is negated at the time that an acknowledge is asserted. This type of operation would be true for a system consisting of the processor and one device capable of bus mastership. In systems having a number of devices capable of bus mastership, the bus request line from each device is wire ORed to the processor. In this system, it is easy to see that there could be more that one bus request being made. The timing diagram shows that the bus grant signal is negated a few clock cycles after the transition of the acknowledge ($\overline{BGACK}$) signal.

However, if the bus requests are still pending, the processor will assert another bus grant within a few clock cycles after it was negated. This additional assertion of bus grant allows external arbitration circuitry to select the next bus master before the current bus master has completed its requirements. The following paragraphs provide additional information about the three steps in the arbitration process.

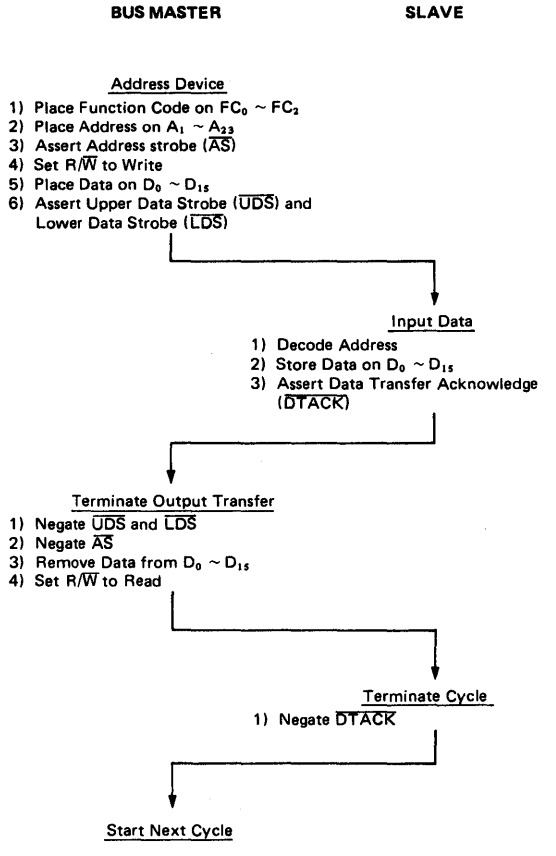


Figure 15 Word Write Cycle Flow Chart

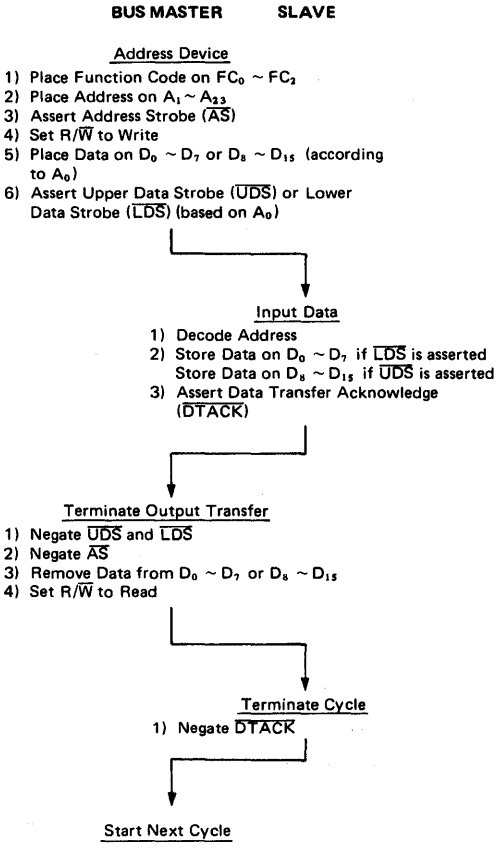


Figure 16 Byte Write Cycle Flow Chart

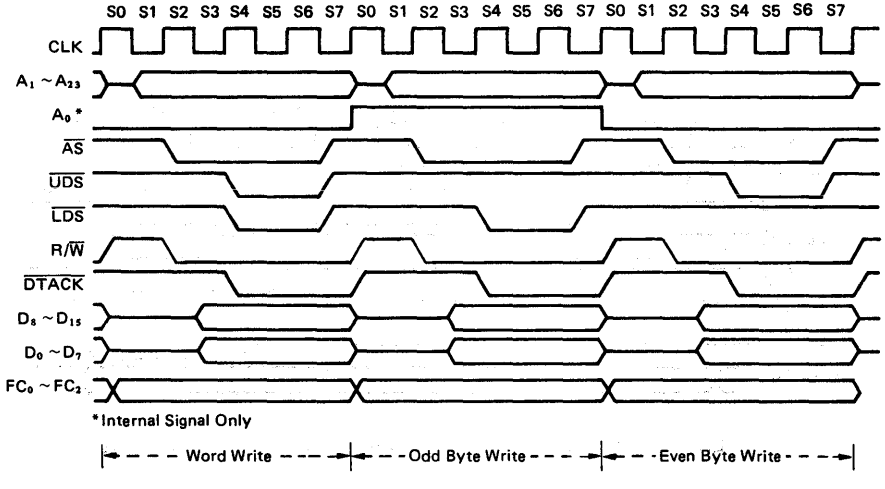


Figure 17 Word and Byte Write Cycle Timing Diagram

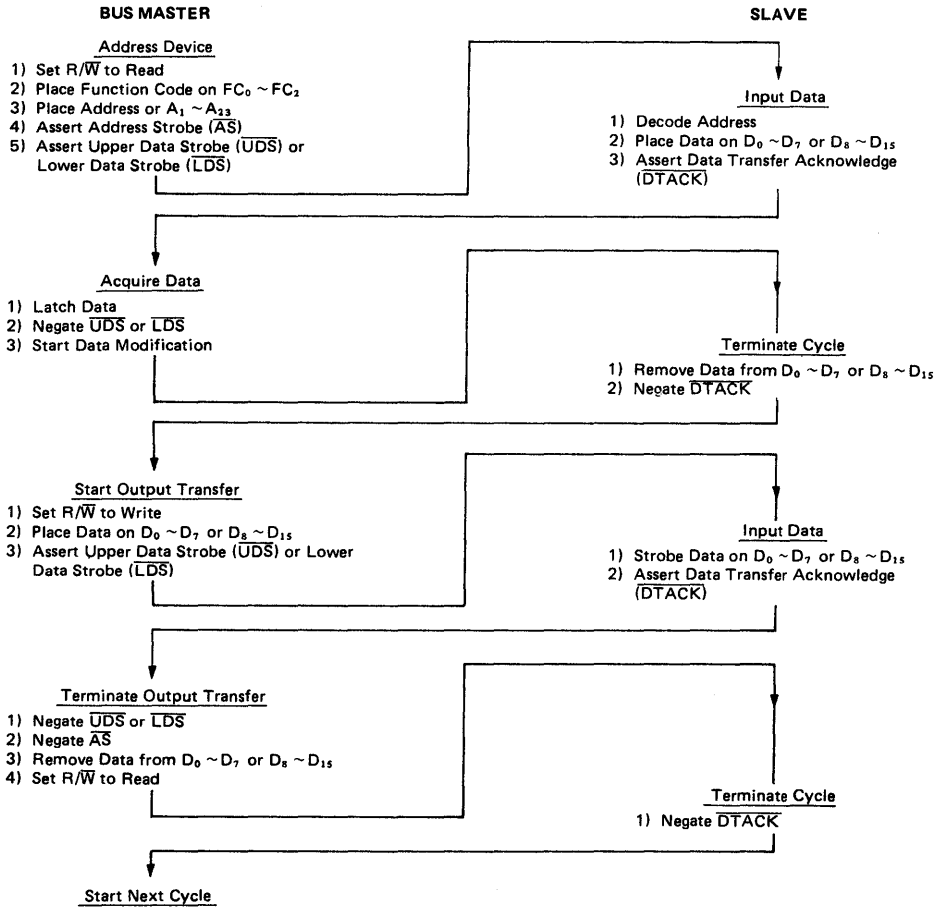


Figure 18 Read-Modify-Write Cycle Flow Chart

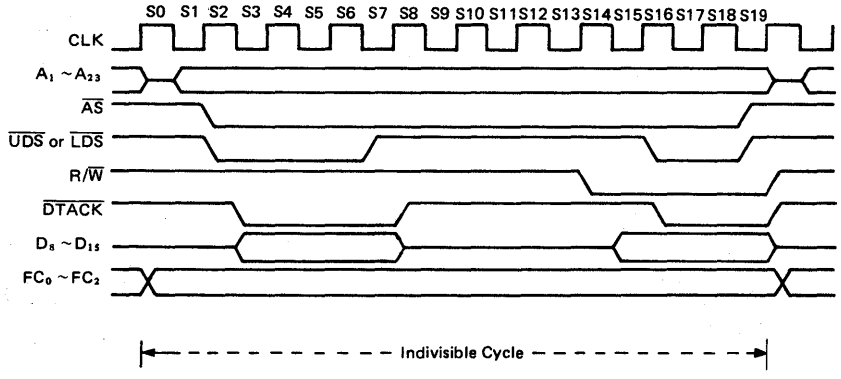


Figure 19 Read-Modify-Write Cycle Timing Diagram

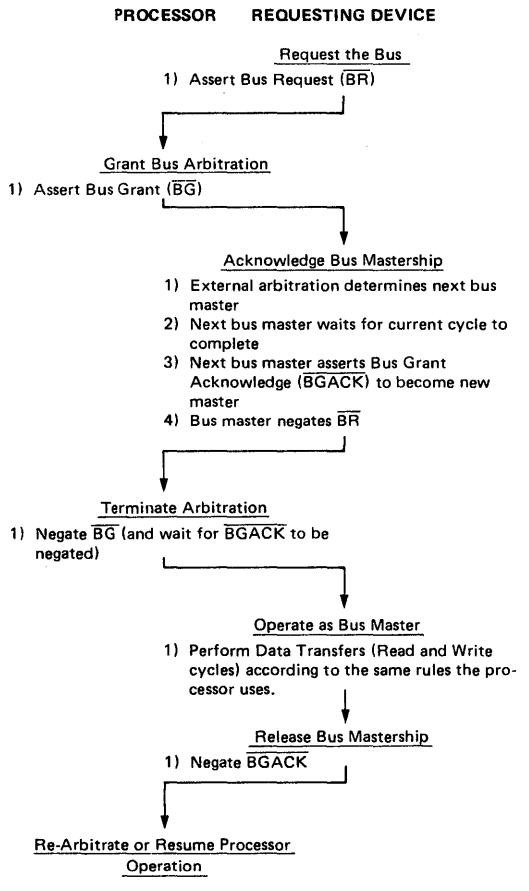


Figure 20 Bus Arbitration Cycle Flow Chart

Requesting the Bus

External devices capable of becoming bus masters request the bus by asserting the bus request ($\overline{BR}$) signal. This is a wire ORed signal (although it need not be constructed from open collector devices) that indicates to the processor that some external device requires control of the external bus. The processor is effectively at a lower bus priority level that the external device and will relinquish the bus after it has completed the last bus cycle it has started.

When no acknowledge is received before the bus request signal goes inactive, the processor will continue processing when it detects that the bus request is inactive. This allows ordinary processing to continue if the arbitration circuitry responded to noise inadvertently.

Receiving the Bus Grant

The processor asserts bus grant ($\overline{BG}$) as soon as possible. Normally this is immediately after internal synchronization. The only exception to this occurs when the processor has made an internal decision to execute the next bus cycle but has not progressed far enough into the cycle to have asserted the address strobe ($\overline{AS}$) signal. In this case, bus grant will not be asserted until one clock after address strobe is asserted to indicate to external devices that a bus cycle is being executed.

The bus grant signal may be routed through a daisy-chained network or through a specific priority-encoded network. The processor is not affected by the external method of arbitration as long as the protocol is obeyed.

Acknowledgement of Mastership

Upon receiving a bus grant, the requesting device waits until address strobe, data transfer acknowledge, and bus grant acknowledge are negated before issuing its own $\overline{BGACK}$. The negation of the address strobe indicates that the previous master has completed its cycle, the negation of bus grant acknowledge indicates that the previous master has released the bus. (While address strobe is asserted no device is allowed to "break into" a cycle.) The negation of data transfer acknowledge indicates the previous slave has terminated its connection to the previous master. Note that in some applications data

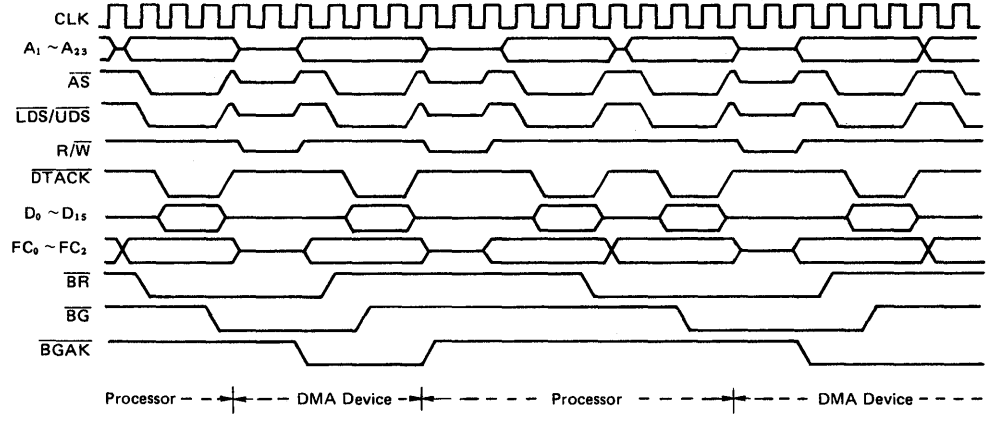


Figure 21 Bus Arbitration Cycle Timing Diagram

transfer acknowledge might not enter into this function. General purpose devices would then be connected such that they were only dependent on address strobe. When bus grant acknowledge is issued the device is bus master until it negates bus grant acknowledge. Bus grant acknowledge should not be negated until after the bus cycle(s) is (are) completed. Bus mastership is terminated at the negation of bus grant acknowledge.

The bus request from the granted device should be dropped after bus grant acknowledge is asserted. If a bus request is still pending, another bus grant will be asserted within a few clocks of the negation of bus grant. Refer to Bus Arbitration Control section. Note that the processor does not perform any external bus cycles before it re-asserts bus grant.

• BUS ARBITRATION CONTROL

The bus arbitration control unit in the HD68000 is implemented with a finite state machine. A state diagram of this machine is shown in Figure 22. All asynchronous signals to the HD68000 are synchronized before being used internally. This synchronization is accomplished in a maximum of one cycle of the system clock, assuming that the asynchronous input setup time (#47) has been met (see Figure 23). The input signal is sampled on the falling edge of the clock and is valid internally after the next falling edge.

As shown in Figure 22, input signals labeled R and A are internally synchronized on the bus request and bus grant acknowledge pins respectively. The bus grant output is lebeled G and the internal three-state control signal T. If T is true, the address, data, and control buses are placed in a high-impedance state when $\overline{AS}$ is negated. All signals are shown in positive logic (active high) regardless of their true active voltage level.

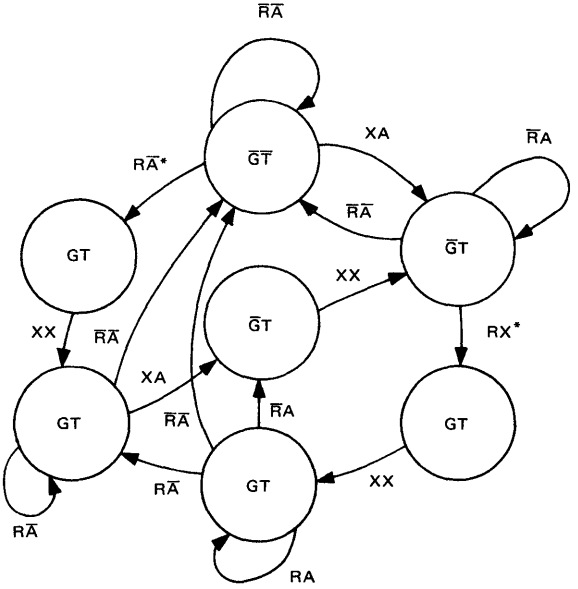
State changes (valid outputs) occur on the next rising edge after the internal signal is valid.

A timing diagram of the bus arbitration sequence during a processor bus cycle is shown in Figure 24. The bus arbitration sequence while the bus is inactive (i.e., executing internal operations such as a multiply instruction) is shown in Figure 25.

If a bus request is made at a time when the MPU has already begun a bus cycle but $\overline{AS}$ has not been asserted (bus state S0), $\overline{BG}$ will not be asserted on the next rising edge. Instead, $\overline{BG}$ will be delayed until the second rising edge following it's internal assertion. This sequence is shown in Figure 26.

• BUS ERROR AND HALT OPERATION

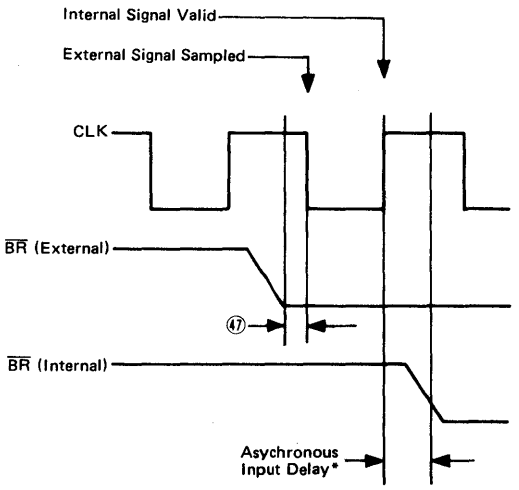
In a bus architecture that requires a handshake from an external device, the possibility exists that the handshake might not occur. Since different systems will require a different maximum response time, a bus error input is provided. External circuitry must be used to determine the duration between address strobe and data transfer acknowledge before issuing a bus error signal. When a bus error signal is received, the processor has two options initiate a bus error exception sequence or try running the bus cycle again.



R = Bus Request Internal
A = Bus Grant Acknowledge Internal
G = Bus Grant
T = Three-State Control to Bus Control Logic
X = Don't Care

* State machine will not change state if bus is in S0. Refer to BUS ARBITRATION CONTROL for additional information.

Figure 22 State Diagram of HD68000 Bus Arbitration Unit



* This delay time is equal to parameter #33, t_{CHGL}.

Figure 23 Timing Relationship of External Asynchronous Inputs to Internal Signals

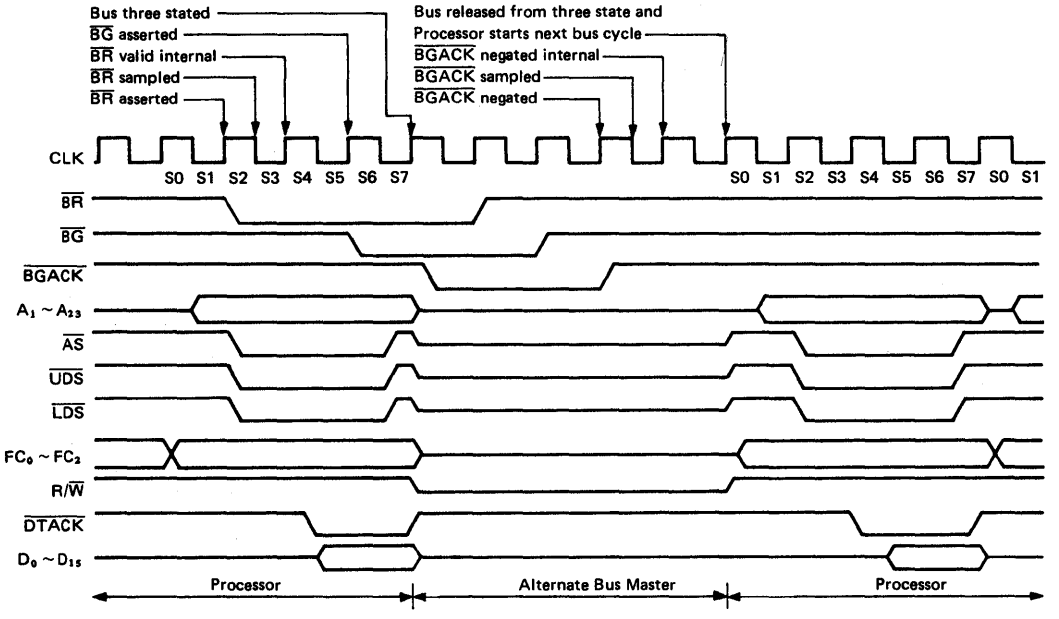


Figure 24 Bus Arbitration During Processor Bus Cycle

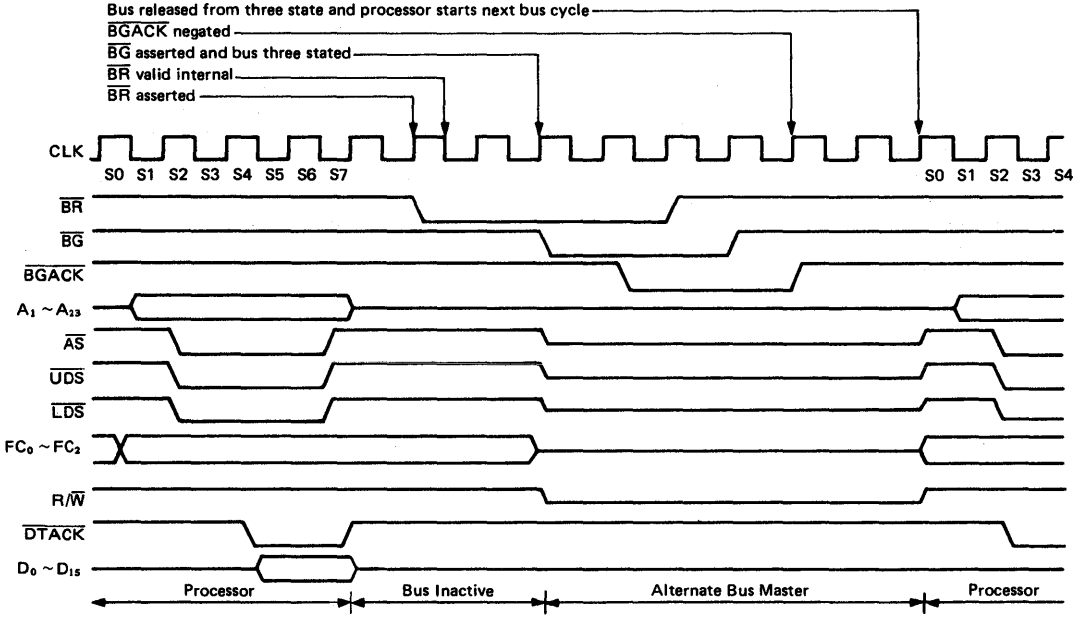


Figure 25 Bus Arbitration with Bus Inactive

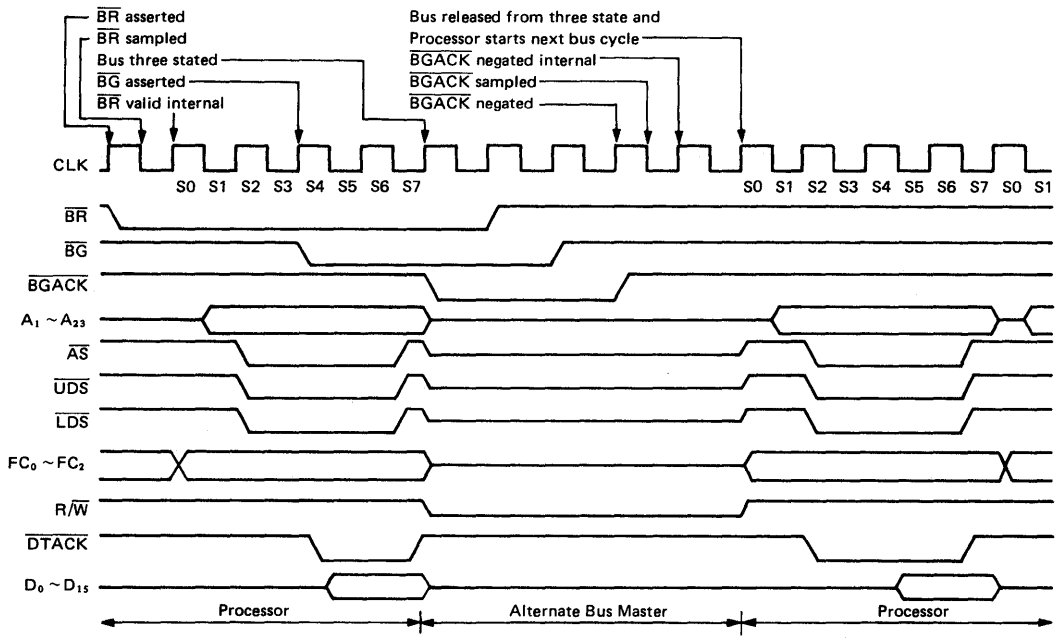


Figure 26 Bus Arbitration During Processor Bus Cycle Special Case

Exception Sequence

When the bus error signal is asserted, the current bus cycle is terminated. If $\overline{BERR}$ is asserted before the falling edge of S4, $\overline{AS}$ will be negated in S9 in either a read or write cycle. As long as $\overline{BERR}$ remains asserted, the data and address buses will be in the high-impedance state. When $\overline{BERR}$ is negated, the processor will begin stacking for exception processing. Figure 27 is a timing diagram for the exception sequence. The sequence is composed of the following elements.

(1) Stacking the program counter and status register
(2) Stacking the error information
(3) Reading the bus error vector table entry
(4) Executing the bus error handler routine

The stacking of the program counter and the status register is the same as if an interrupt had occurred. Several additional items are stacked when a bus error occurs. These items are used to determine the nature of the error and correct it, if possible. The bus error vector is vector number two located at address $000008. The processor loads the new program counter from this location. A software bus error handler routine is then executed by the processor. Refer to **EXCEPTION PROCESSING** for additional information.

Re-Running the Bus Cycle

When, during a bus cycle, the processor receives a bus error signal and the halt pin is being driven by an external device, the processor enters the re-run sequence. Figure 28 is a timing diagram for re-running the bus cycle.

The processor terminates the bus cycle, then puts the address and data output lines in the high-impedance state. The processor remains "halted," and will not run another bus cycle until the halt signal is removed by external logic. Then the processor will re-run the previous bus cycle using the same address, the same function codes, the same data (for a write operation), and the same controls. The bus error signal should be removed at least one clock cycle before the halt signal is removed.

(NOTE) The processor will not re-run a read-modify-write cycle. This restriction is made to guarantee that the entire cycle runs correctly and that the write operation of a Test-and-Set operation is performed without ever releasing $\overline{AS}$. If $\overline{BERR}$ and $\overline{HALT}$ are asserted during a read-modify-write bus cycle, a bus error operation results.

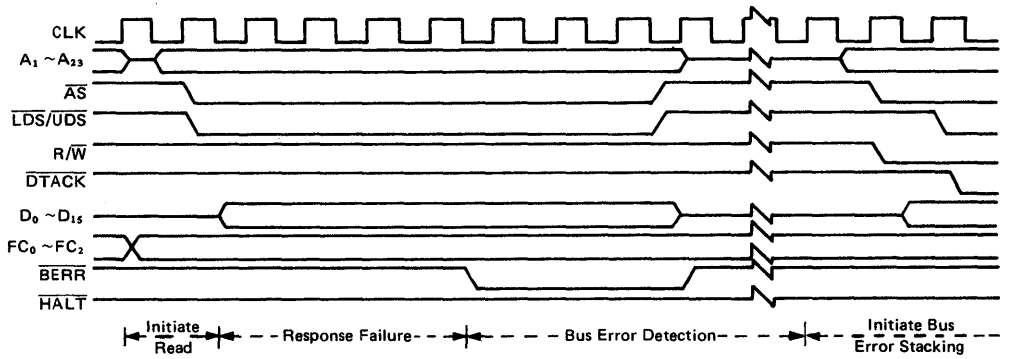


Figure 27 Bus Error Timing Diagram

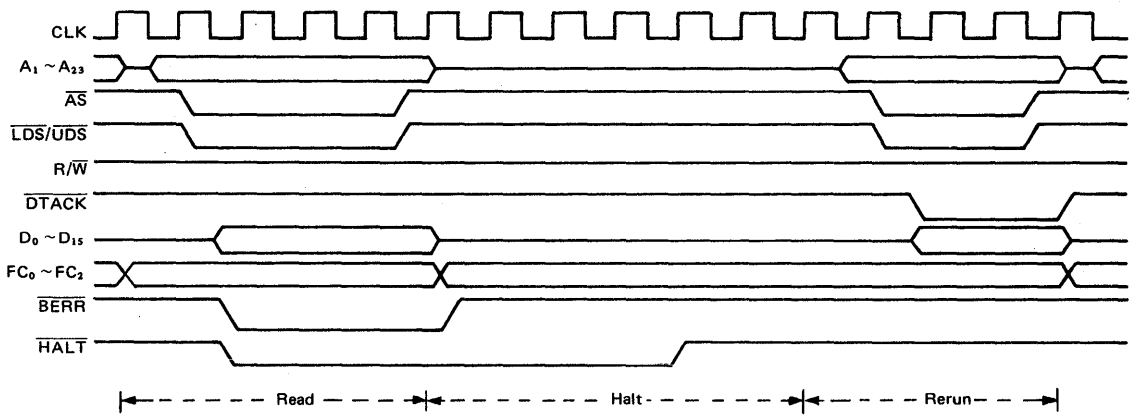


Figure 28 Re-Run Bus Cycle Timing Information

The processor terminates the bus cycle, then puts the address, data and function code output lines in the high-impedance state. The processor remains "halted," and will not run another bus cycle until the halt signal is removed by external logic. Then the processor will re-run the previous bus cycle using the same address, the same function codes, the same data (for a write operation), and the same controls. The bus error signal should be removed before the halt signal is removed.

Halt Operation with No Bus Error

The halt input signal to the HD68000 perform a Halt/Run/Single-Step function in a similar fashion to the HMCS6800 halt function. The halt and run modes are somewhat self explanatory in that when the halt signal is constantly active the processor "halts" (does nothing) and when the halt signal is constantly inactive the processor "runs" (does something).

The single-step mode is derived from correctly timed transitions on the halt signal input. It forces the processor to execute a single bus cycle by entering the "run" mode until the processor starts a bus cycle then changing to the "halt" mode. Thus, the single-step mode allows the user to proceed through (and therefore debug) processor operations one bus cycle at a time.

Figure 29 details the timing required for correct single-step operations. Some care must be exercised to avoid harmful interactions between the bus error signal and the halt pin when using the single cycle mode as a debugging tool. This is also true of interactions between the halt and reset lines since these can reset the machine.

When the processor completes a bus cycle after recognizing that the halt signal is active, most three-state signals are put in the high-impedance state. These include:

(1) Address lines
(2) Data lines

This is required for correct performance of the re-run bus cycle operation.

While the processor is honoring the halt request, bus arbitration performs as usual. That is, halting has no effect on bus arbitration. It is the bus arbitration function that removes the control signals from the bus.

The halt function and the hardware trace capability allow the hardware debugger to trace single bus cycles or single instructions at a time. These processor capabilities, along with a software debugging package, give total debugging flexibility.

Double Bus Faults

When a bus error exception occurs, the processor will attempt to stack several words containing information about the state of the machine. If a bus error exception occurs during the stacking operation, there have been two bus errors in a row. This is commonly referred to as a double bus fault. When a double bus fault occurs, the processor will halt. Once a bus error exception has occurred, any bus error exception occurring before the execution of the next instruction constitutes a double bus fault.

Note that a bus cycle which is re-run does not constitute a bus error exception, and does not contribute to a double bus fault. Note also that this means that as long as the external hardware requests it, the processor will continue to re-run the same bus cycle.

The bus error pin also has an effect on processor operation after the processor receives an external reset input. The processor reads the vector table after a reset to determine the address to start program execution. If a bus error occurs while reading the vector table (or at any time before the first instruction is executed), the processor reacts as if a double bus fault has occurred and it halts. Only an external reset will start a halted processor.

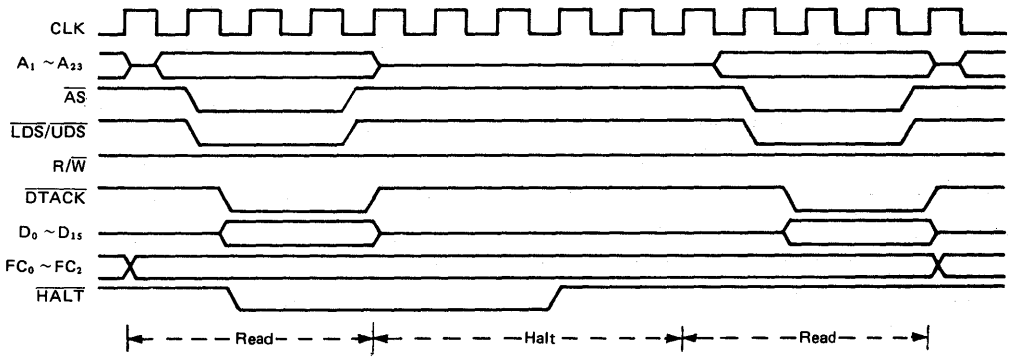


Figure 29 Halt Signal Timing Characteristics

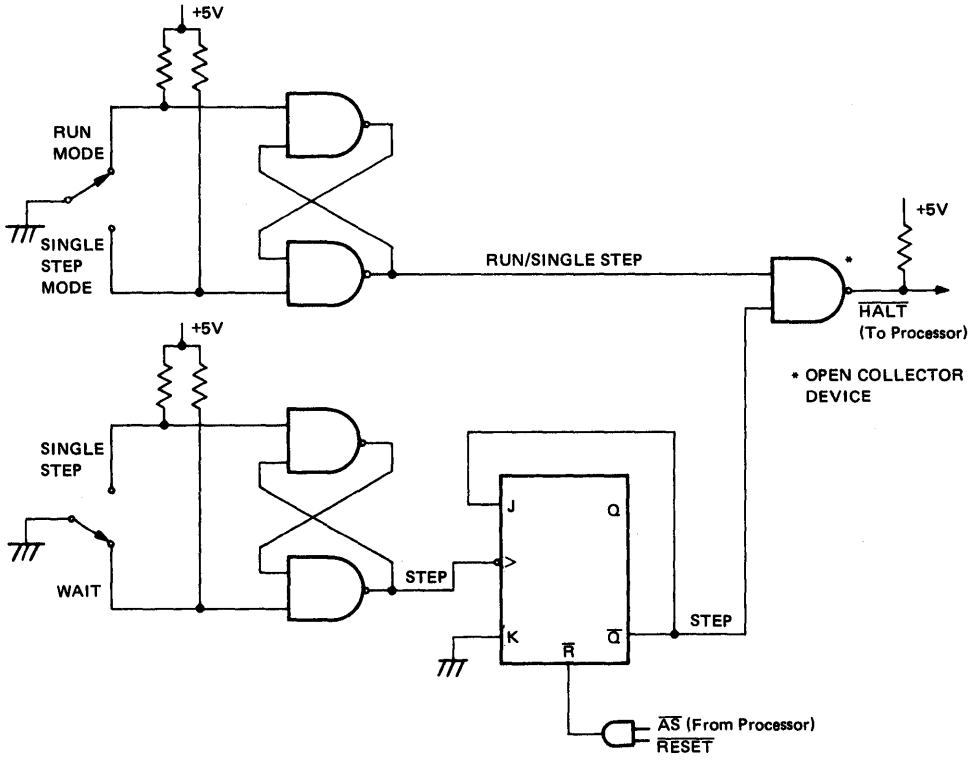


Figure 30 Simplified Single-Step Circuit

■ THE RELATIONSHIP OF DTACK, BERR, AND HALT

In order to properly control termination of a bus cycle for a re-run or a bus error condition, DTACK, BERR, and HALT should be asserted and negated on the rising edge of the HD68000 clock. This will assure that when two signals are asserted simultaneously, the required setup time (#47) for both of them will be met during the same bus state.

This, or some equivalent precaution, should be designed external to the HD68000. Parameter #48 is intended to ensure this operation in a totally asynchronous system, and may be ignored if the above conditions are met.

The preferred bus cycle terminations may be summarized as follows (case numbers refer to Table 4):

Normal Termination: DTACK occurs first (case 1).

Halt Termination: HALT is asserted at same time, or precedes DTACK (no BERR) cases 2 and 3.

Bus Error Termination: BERR is asserted in lieu of, at same time, or preceding DTACK (case 4); BERR negated at same time, or after DTACK.

Re-Run Termination: HALT and BERR asserted at the same time, or before DTACK (cases 6 and 7); HALT must be negated at least 1 cycle after BERR. (Case 5 indicates BERR may precede HALT which allows fully asynchronous assertion).*

Table 4 details the resulting bus cycle termination under various combinations of control signal sequences. The negation of these same control signals under several conditions is shown in Table 5 (DTACK is assumed to be negated normally in all cases; for best results, both DTACK and BERR should be negated when address strobe is negated.)

Example A: A system uses a watch-dog timer to terminate accesses to un-populated address space. The timer asserts DTACK and BERR simultaneously after time-out. (case 4)

Example B: A system uses error detection on RAM contents. Designer may (a) delay DTACK until data verified, and return BERR and HALT simultaneously to re-run error cycle (case 6), or if valid, return DTACK; (b) delay DTACK until data verified, and return BERR at same time as DTACK if data in error (case 4); (c) return DTACK prior to data verification, as described in previous section. If data invalid, BERR is asserted (case 1) in next cycle. Error-handling software must know how to recover error cycle.

* For the mask version 68000, HALT and BERR must be asserted at the same time.

Table 4 $\overline{DTACK}$, $\overline{BERR}$, $\overline{HALT}$ Assertion Results

Case No.	Control Signal	Asserted on Rising Edge of State		Result
		N	N + 2	
1	$\overline{DTACK}$	A	S	Normal cycle terminate and continue.
	$\overline{BERR}$	NA	X	
	$\overline{HALT}$	NA	X	
2	$\overline{DTACK}$	A	S	Normal cycle terminate and halt. Continue when HALT removed.
	$\overline{BERR}$	NA	X	
	$\overline{HALT}$	A	S	
3	$\overline{DTACK}$	NA	A	Normal cycle terminate and halt. Continue when HALT removed.
	$\overline{BERR}$	NA	NA	
	$\overline{HALT}$	A	S	
4	$\overline{DTACK}$	X	X	Terminate and take bus error trap.
	$\overline{BERR}$	A	S	
	$\overline{HALT}$	NA	NA	
5	$\overline{DTACK}$	NA	X	Terminate and re-run*.
	$\overline{BERR}$	A	S	
	$\overline{HALT}$	NA	A	
6	$\overline{DTACK}$	X	X	Terminate and re-run.
	$\overline{BERR}$	A	S	
	$\overline{HALT}$	A	S	
7	$\overline{DTACK}$	NA	X	Terminate and re-run when HALT removed.
	$\overline{BERR}$	NA	A	
	$\overline{HALT}$	A	S	

Legend:
N – The number of the current even bus state (e.g., S4, S6, etc.)
A – Signal is asserted in this bus state
NA – Signal is not asserted in this state
X – Don't care
S – Signal was asserted in previous state and remains asserted in this state

* For the mask version 68000, unpredictable results, no re-run, no error trap; usually traps to vector number 0.

Table 5 $\overline{BERR}$ and $\overline{HALT}$ Negation Results

Conditions of Termination in Table A	Control Signal	Negated on Rising Edge of State			Results – Next Cycle
		N		N + 2	
Bus Error	$\overline{BERR}$	●	or	●	Takes bus error trap.
	$\overline{HALT}$	●	or	●	
Re-run	$\overline{BERR}$	●	or	●	Illegal sequence; usually traps to vector number 0.
	$\overline{HALT}$	●			
Re-run	$\overline{BERR}$	●			Re-runs the bus cycle.
	$\overline{HALT}$			●	
Normal	$\overline{BERR}$	●			May lengthen next cycle.
	$\overline{HALT}$	●	or	●	
Normal	$\overline{BERR}$			●	If next cycle is started it will be terminated as a bus error.
	$\overline{HALT}$	●	or	none	

• RESET OPERATION

The reset signal is a bidirectional signal that allows either the processor or an external signal to reset the system. Figure 31 is a timing diagram for reset operations. Both the halt and reset lines must be applied to ensure total reset of the processor.

When the reset and halt lines are driven by an external device, it is recognized as an entire system reset, including the processor. The processor responds by reading the reset vector table entry (vector unumber zero, address $000000) and loads it into the supervisor stack pointer (SSP). Vector table entry number one at address $000004 is read next and loaded into the program counter. The processor initializes the status register to an interrupt level of seven. No other registers are affected by the reset sequence.

When a RESET sequence is executed, the processor drives the reset pin for 124 clock pulses. In this case, the processor is trying to reset the rest of the system. Therefore, there is no effect on the internal state of the processor. All of the processor's internal registers and the status register are unaffected by the execution of a RESET instruction. All external devices connected to the reset line should be reset at the completion of the RESET instruction.

Asserting the Reset and Halt pins for 10 clock cycles will cause a processor reset, except when V_{CC} is initially applied to the processor. In this case, an external reset must be applied for 100 milliseconds.

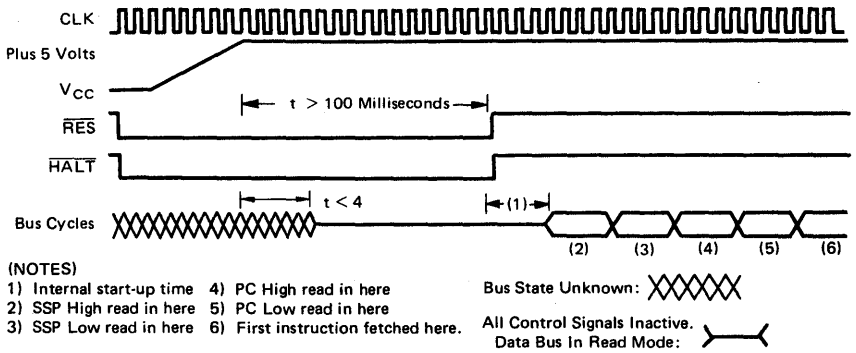


Figure 31 Reset Operation Timing Diagram

■ PROCESSING STATES

The HD68000 is always in one of three processing states: normal, exception, or halted. The normal processing state is that associated with instruction execution; the memory of the bits in the supervisor portion of the status register are covered: the supervisor/user bit, the trace enable bit, and the processor interrupt priority mask. Finally, the sequence of memory references and actions taken by the processor on exception conditions is detailed.

The HD68000 is always in one of three processing states: normal, exception, or halted. The normal processing state is that associated with instruction execution; the memory references are to fetch instructions and operands, and to store results. A special case of the normal state is the stopped state which the processor enters when a STOP instruction is executed. In this state, no further memory references are made.

The exception processing state is associated with interrupts, trap instructions, tracing and other exceptional conditions. The exception may be internally generated by an instruction or by an unusual condition arising during the execution of an instruction. Externally, exception processing can be forced by an interrupt, by a bus error, or by a reset. Exception processing is designed to provide an efficient context switch so that the processor may handle unusual conditions.

The halted processing state is an indication of catastrophic hardware failure. For example, if during the exception processing of a bus error another bus error occurs, the processor assumes that the system is unusable and halts. Only an external reset can restart a halted processor. Note that a processor in the stopped state is not in the halted state, nor vice versa.

PROCESSING STATES

NORMAL	INSTRUCTION EXECUTION (INCLUDING STOP)
EXCEPTION	INTERRUPTS TRAPS TRACING ETC.
HALTED	HARDWARE HALT DOUBLE BUS FAULT

● PRIVILEGE STATES

The processor operates in one of two states of privilege: the "user" state or the "supervisor" state. The privilege state determines which operations are legal, is used by the external memory management device to control and translate accesses, and is used to choose between the supervisor stack pointer and the user stack pointer in instruction references.

The privileges state is a mechanism for providing security in a computer system. Programs should access only their own code and data areas, and ought to be restricted from accessing information which they do not need and must not modify.

The privilege mechanism provides security by allowing most programs to execute in user state. In this state, the accesses are controlled, and the effects on other parts of the system are limited. The operating system executes in the supervisor state, has access to all resources, and performs the overhead tasks for the user state programs.

SUPERVISOR STATE

The supervisor state is the higher state of privilege. For instruction execution, the supervisor state is determined by the S-bit of the status register; if the S-bit is asserted (high), the processor is in the supervisor state. All instructions can be executed in the supervisor state. The bus cycles generated by instructions executed in the supervisor state are classified as supervisor references. While the processor is in the supervisor privilege state, those instructions which use either the system stack pointer implicitly or address register seven explicitly access the supervisor stack pointer.

All exception processing is done in the supervisor state, regardless of the setting of the S-bit. The bus cycles generated during exception processing are classified as supervisor references. All stacking operations during exception processing use the supervisor stack pointer.

USER STATE

The user state is the lower state of privilege. For instruction execution, the user state is determined by the S-bit of the status register; if the S-bit is negated (low), the processor is executing instructions in the user state.

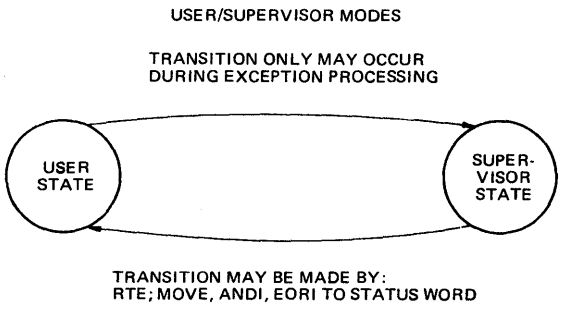
Most instructions execute the same in user state as in the supervisor state. However, some instructions which have important system effects are made privileged. User programs are not permitted to execute the STOP instruction, or the

RESET instruction. To ensure that a user program cannot enter the supervisor state except in a controlled manner, the instructions which modify the whole status register are privileged. To aid in debugging programs which are to be used as operating systems, the move to user stack pointer (MOVE USP) and move from user stack pointer (MOVE from USP) instructions are also privileged.

The bus cycles generated by an instruction executed in user state are classified as user state references. This allows an external memory management device to translate the address and to control access to protected portions of the address space. While the processor is in the user privilege state, those instructions which use either the system stack pointer implicitly, or address register seven explicitly, access the use stack pointer.

PRIVILEGE STATE CHANGES

Once the processor is in the user state and executing instructions, only exception processing can change the privilege state. During exception processing, the current setting of the S-bit of the status register is saved and the S-bit is asserted, putting the processing in the supervisor state. Therefore, when instruction execution resumes at the address specified to process the exception, the processor is in the supervisor privilege state.



REFERENCE CLASSIFICATION

When the processor makes a reference, it classifies the kind of reference being made, using the encoding on the three function code output lines. This allows external translation of addresses, control of access, and differentiation of special processor states, such as interrupt acknowledge. Table 6 lists the classification of references.

Table 6 Reference Classification

Function Code Output			Reference Class
FC_2	FC_1	FC_0	
0	0	0	(Unassigned)
0	0	1	User Data
0	1	0	User Program
0	1	1	(Unassigned)
1	0	0	(Unassigned)
1	0	1	Supervisor Data
1	1	0	Supervisor Program
1	1	1	Interrupt Acknowledge

• EXCEPTION PROCESSING

Before discussing the details of interrupts, traps, and tracing, a general description of exception processing is in order. The processing of an exception occurs in four steps, with variations for different exception causes. During the first step, a temporary copy of the status register is made, and the status register is set for exception processing. In the second step the exception vector is determined, and the third step is the saving of the current processor context. In the fourth step a new context is obtained, and the processor switches to instruction processing.

EXCEPTION VECTORS

Exception vectors are memory locations from which the processor fetches the address of a routine which will handle that exception. All exception vectors are two words in length (Figure 32), except for the reset vector, which is four words. All exception vectors lie in the supervisor data space, except for the reset vector which is in the supervisor program space. A vector number is an eight-bit number which, when multiplied by four, gives the address of an exception vector. Vector numbers are generated internally or externally depending on the cause of the exception. In the case of interrupts, during the interrupt acknowledge bus cycle, a peripheral provides an 8-bit vector number (Figure 33) to the processor on data bus lines D_0 through D_7. The processor translates the vector number into a full 24-bit address, as shown in Figure 34. The memory layout for exception vectors is given in Table 7.

As shown in Table 7, the memory layout is 512 words long (1024 bytes). It starts at address 0 and proceeds through address 1023. This provides 255 unique vectors; some of these are reserved for TRAPS and other system functions. Of the 255, there are 192 reserved for user interrupt vectors. However, there is no protection on the first 64 entries, so user interrupt vectors may overlap at the discretion of the systems designer.

KINDS OF EXCEPTIONS

Exceptions can be generated by either internal or external causes. The externally generated exceptions are the interrupts and the bus error and reset requests. The interrupts are requests from peripheral devices for processor action while the bus error and reset inputs are used for access control and processor restart. The internally generated exceptions come from instructions, or from address error or tracing. The trap (TRAP), trap on overflow (TRAPV), check register against bounds (CHK) and divide (DIV) instructions all can generate exceptions as part of their instruction execution. In addition, illegal instructions, word fetches from odd addresses and privilege violations cause exceptions. Tracing behaves like a very high priority, internally generated interrupt after each instruction execution.

EXCEPTION PROCESSING SEQUENCE

Exception processing occurs in four identifiable steps. In the first step, an internal copy is made of the status register. After the copy is made, the S-bit is asserted, putting the processor into the supervisor privilege state. Also, the T-bit is negated which will allow the exception handler to execute unhindered by tracing. For the reset and interrupt exceptions, the interrupt priority mask is also updated.

In the second step, the vector number of the exception is determined. For interrupts, the vector number is obtained by a processor fetch, classified as an interrupt acknowledge. For all other exceptions, internal logic provides the vector number. This vector number is then used to generate the address of the exception vector.

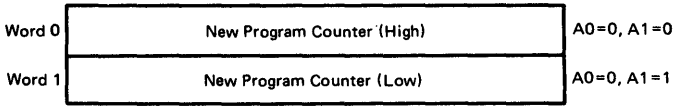
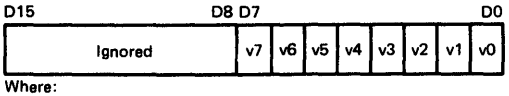


Figure 32 Exception Vector Format



Where:
v7 is the MSB of the Vector Number
v0 is the LSB of the Vector Number

Figure 33 Peripheral Vector Number Format

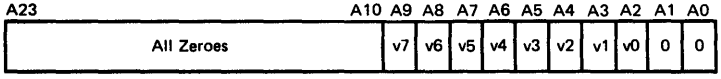


Figure 34 Address Translated From 8-Bit Vector Number

Table 7 Exception Vector Assignment

Vector Number(s)	Address			Assignment
	Dec	Hex	Space	
0	0	000	SP	Reset: Initial SSP
–	4	004	SP	Reset: Initial PC
2	8	008	SD	Bus Error
3	12	00C	SD	Address Error
4	16	010	SD	Illegal Instruction
5	20	014	SD	Zero Divide
6	24	018	SD	CHK Instruction
7	28	01C	SD	TRAPV Instruction
8	32	020	SD	Privilege Violation
9	36	024	SD	Trace
10	40	028	SD	Line 1010 Emulator
11	44	02C	SD	Line 1111 Emulator
12*	48	030	SD	(Unassigned, reserved)
13*	52	034	SD	(Unassigned, reserved)
14*	56	038	SD	(Unassigned, reserved)
15	60	03C	SD	Uninitialized Interrupt Vector
16 ~ 23*	64	04C	SD	(Unassigned, reserved)
	95	05F		
24	96	060	SD	Spurious Interrupt
25	100	064	SD	Level 1 Interrupt Autovector
26	104	068	SD	Level 2 Interrupt Autovector
27	108	06C	SD	Level 3 Interrupt Autovector
28	112	070	SD	Level 4 Interrupt Autovector
29	116	074	SD	Level 5 Interrupt Autovector
30	120	078	SD	Level 6 Interrupt Autovector
31	124	07C	SD	Level 7 Interrupt Autovector
32 ~ 47	128	080	SD	TRAP Instruction Vectors
	191	0BF		
48 ~ 63*	192	0C0	SD	(Unassigned, reserved)
	255	0FF		
64 ~ 255	256	100	SD	User Interrupt Vectors
	1023	3FF		

SP: Supervisor program, SD: Supervisor data

* Vector numbers 12, 13, 14, 16 through 23 and 48 through 63 are reserved for future enhancements by Hitachi. No user peripheral devices should be assigned these numbers.

The third step is to save the current processor status, except for the reset exception. The current program counter value and the saved copy of the status register are stacked using the supervisor stack pointer. The program counter value stacked usually points to the next unexecuted instruction, however for bus error and address error, the value stacked for the program counter is unpredictable, and may be incremented from the address of the instruction which caused the error. Additional information defining the current context is stacked for the bus error and address error exceptions.

The last step is the same for all exceptions. The new program counter value is fetched from the exception vector. The processor then resumes instruction execution. Then instruction at the address given in the exception vector is fetched, and normal instruction decoding and execution is started.

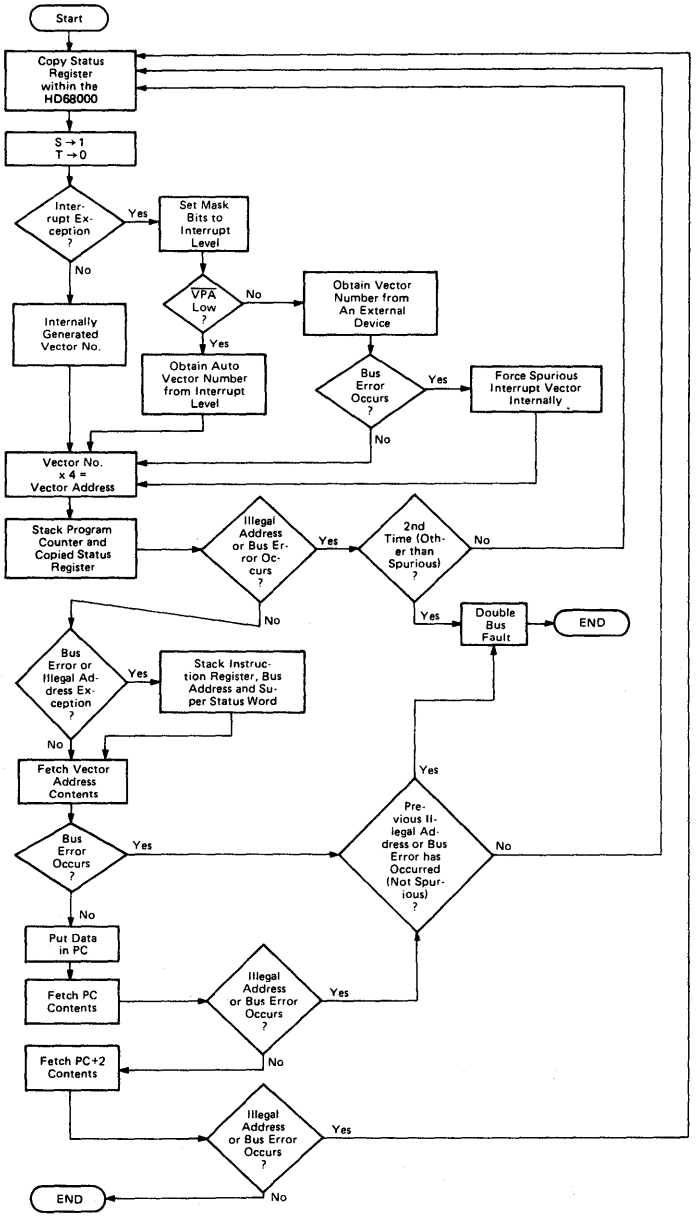


Figure 35 Exception Processing Sequence (Not Reset)

MULTIPLE EXCEPTIONS

These paragraphs describe the processing which occurs when multiple exceptions arise simultaneously. Exceptions can be grouped according to their occurrence and priority. The Group 0 exceptions are reset, but error, and address error. These exceptions cause the instruction currently being executed to be aborted, and the exeception processing to commence within two clock cycles. The Group 1 exceptions are trace and interrupt, as well as the privilege violations and illegal instructions. These exceptions allow the current instruction to execute to completion, but preempt the execution of the next instruction by forcing exception processing to occur (privilege violations and illegal instructions are detected when they are the next instruction to be executed). The Group 2 exceptions occur as part of the normal processing of instructions. The TRAP, TRAPV, CHK, and zero divide exceptions are in this group. For these exceptions, the normal execution of an instruction may lead to exception processing.

Group 0 exceptions have highest priority, while Group 2 exceptions have lowest priority. Within Group 0, reset has highest priority, followed by bus error and then address error. Within Group 1, trace has priority over external interrupts, which in turn takes priority over illegal instruction and privilege violation. Since only one instruction can be executed at a time, there is no priority relation within Group 2.

The priority relation between two exceptions determines which is taken, or taken first, if the conditions for both arise simultaneously. Therefore, if a bus error occurs during a TRAP instruction, the bus error takes precedence, and the TRAP instruction processing is aborted. In another example, if an interrupt request occurs during the execution of an instruction while the T-bit is asserted, the trace exception has priority, and is processed first. Before instruction processing resumes, however, the interrupt exception is also processed, and instruction processing commences finally in the interrupt handler routine. A summary of exception grouping and priority is given in Table 8.

Table 8 Exception Grouping and Priority

Group	Exception	Processing
0	Reset Bus Error Address Error	Exception processing begins within two clock cycles.
1	Trace Interrupt Illegal Privilege	Exception processing begins before the next instruction
2	TRAP, TRAPV CHK, Zero Divide	Exception processing is started by normal instruction execution

RECOGNITION TIMES OF EXCEPTIONS, HALT, AND BUS ARBITRATION

END OF A CLOCK CYCLE
- RESET
- BUS ERROR
- ADDRESS ERROR

END OF A BUS CYCLE
- HALT
- BUS ARBITRATION

END OF AN INSTRUCTION CYCLE
- TRACE EXCEPTION
- INTERRUPT EXCEPTIONS
- ILLEGAL INSTRUCTION
- UNIMPLEMENTED INSTRUCTION
- PRIVILEGE VIOLATION

WITHIN AN INSTRUCTION CYCLE
- TRAP, TRAPV
- CHK
- ZERO DIVIDE

• EXCEPTION PROCESSING DETAILED DISCUSSION

Exceptions have a number of sources, and each exception has processing which is peculiar to it. The following paragraphs detail the sources of exceptions, how each arises, and how each is processed.

RESET

The reset input provides the highest exception level. The processing of the reset signal is designed for system initiation, and recovery from catastrophic failure. Any processing in progress at the time of the reset is aborted and cannot be recovered. The processor is forced into the supervisor state, and the trace state is forced off. The processor interrupt priority mask is set at level seven. The vector number is internally generated to reference the reset exception vector at location 0 in the supervisor program space. Because no assumptions can be made about the validity of register contents, in particular the supervisor stack pointer, neither the program counter nor the status register is saved. The address contained in the first two words of the reset exception vector is fetched as the initial supervisor stack pointer, and the address in the last two words of the reset exception vector is fetched as the initial program counter. Finally, instruction execution is started at the address in the program counter. The power-up/restart code should be pointed to by the initial program counter.

The RESET instruction does not cause loading of the reset vector, but does assert the reset line to reset external devices. This allows the software to reset the system to a known state and then continue processing with the next instruction.

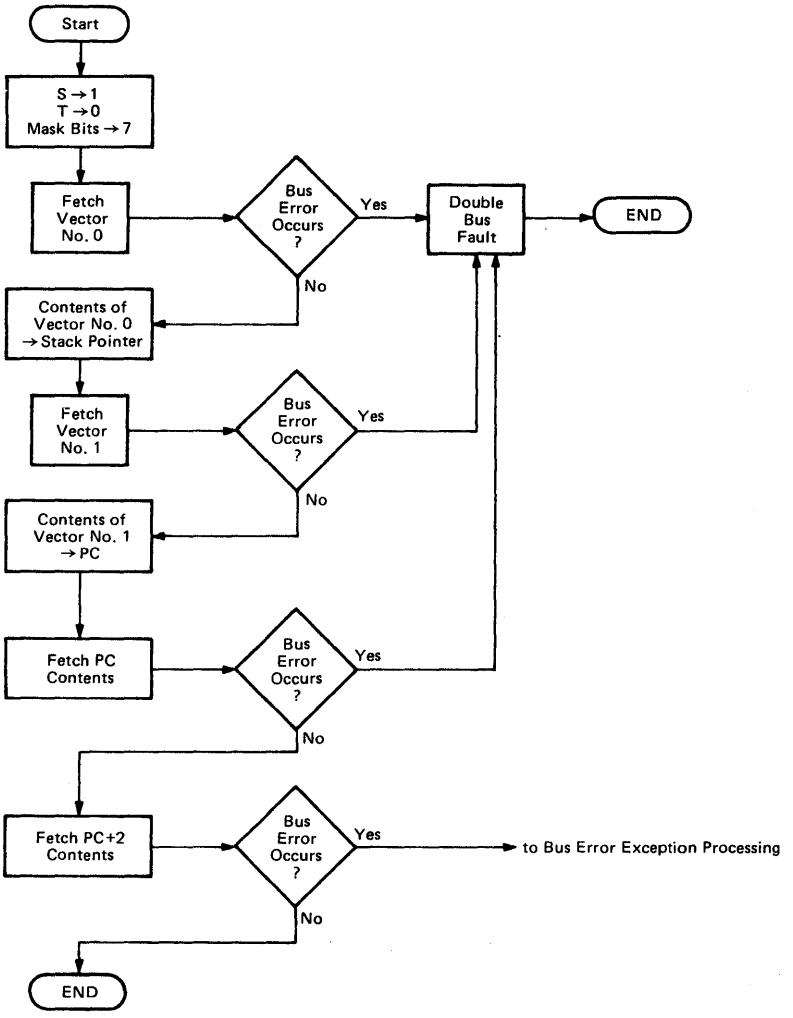


Figure 36 Reset Exception Processing

INTERRUPTS

Seven levels of interrupt priorities are provided. Devices may be chained externally within interrupt priority levels, allowing an unlimited number of peripheral devices to interrupt the processor. Interrupt priority levels are numbered from one to seven, level seven being the highest priority. The status register contains a three-bit mask which indicates the current processor priority, and interrupts are inhibited for all priority levels less than or equal to the current processor priority.

An interrupt request is made to the processor by encoding the interrupt request level on the interrupt request lines; a zero indicates no interrupt request. Interrupt requests arriving at the processor do not force immediate exception processing, but are made pending. Pending interrupts are detected between instruction executions. If the priority of the pending interrupt is lower than or equal to the current processor priority, execution continues with the next instruction and the interrupt exception processing is postponed. (The recognition of level seven is slightly different, as explained in a following paragraph.)

If the priority of the pending interrupt is greater than the current processor priority, the exception processing sequence is started. First a copy of the status register is saved, and the privilege state is set to supervisor, tracing is suppressed, and the processor priority level is set to the level of the interrupt being acknowledged. The processor fetches the vector number from the interrupting device, classifying the reference as an interrupt acknowledge and displaying the level number of

the interrupt being acknowledged on the address bus. If external logic requests an automatic vectoring, the processor internally generates a vector number which is determined by the interrupt level number. If external logic indicates a bus error, the interrupt is taken to be spurious, and the generated vector number references the spurious interrupt vector. The processor then proceeds with the usual exception processing, saving the program counter and status register on the supervisor stack. The saved value of the program counter is the address of the instruction which would have been executed had the interrupt not been present. The content of the interrupt vector whose vector number was previously obtained is fetched and loaded into the program counter, and normal instruction execution commences in the interrupt handling routine. A flow chart for the interrupt acknowledge sequence is given in Figure 37, a timing diagram is given in Figure 38, and the interrupt exception timing sequence is shown in Figure 39.

Table 9 Internal Interrupt Level

Level	I2	I1	I0	Interrupt
7	1	1	1	Non-Maskable Interrupt
6	1	1	0	Maskable Interrupt
5	1	0	1	
4	1	0	0	
3	0	1	1	
2	0	1	0	
1	0	0	1	
0	0	0	0	No Interrupt

(NOTE) The internal interrupt mask level (I2, I1, I0) are inverted to the logic level applied to the pins ($\overline{IPL_2}$, $\overline{IPL_1}$, $\overline{IPL_0}$).

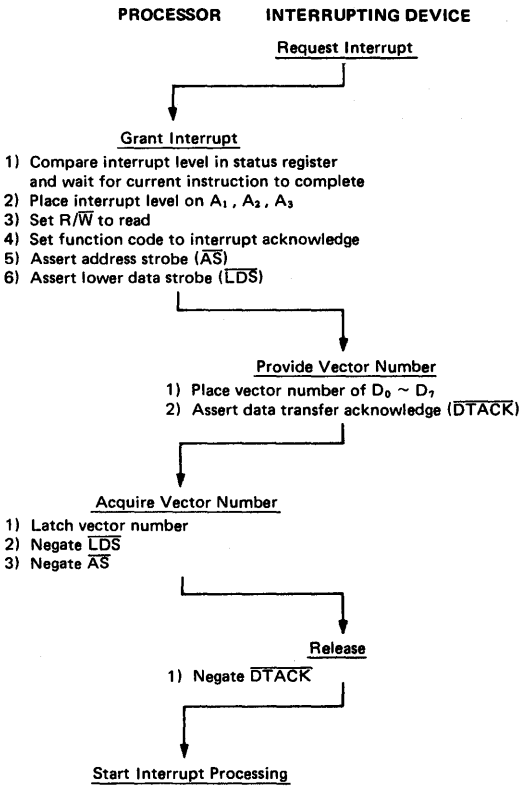


Figure 37 Interrupt Acknowledge Sequence Flow Chart

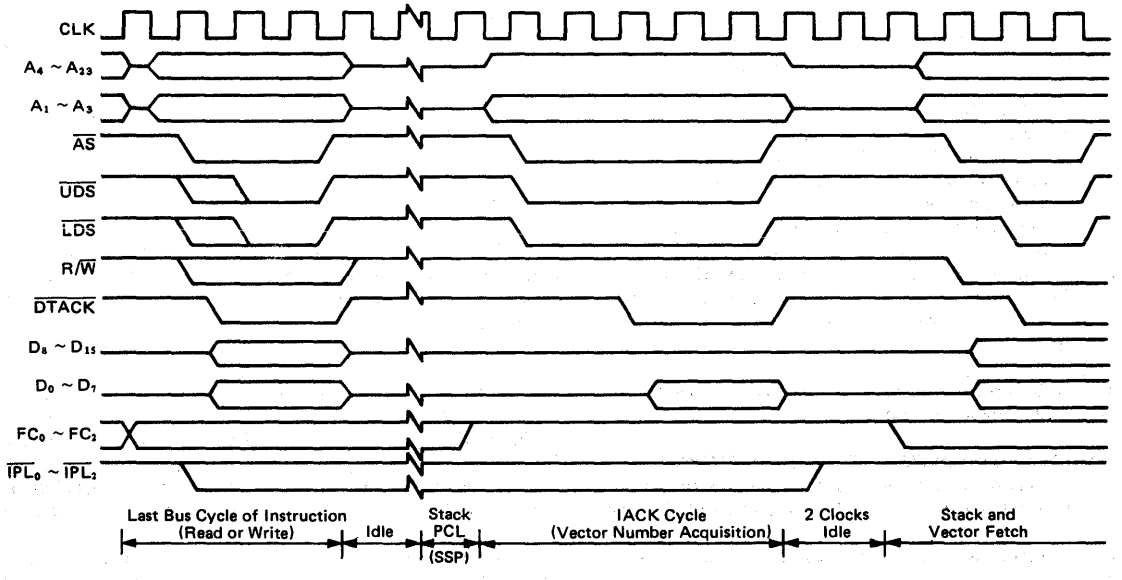


Figure 38 Interrupt Acknowledge Sequence Timing Diagram

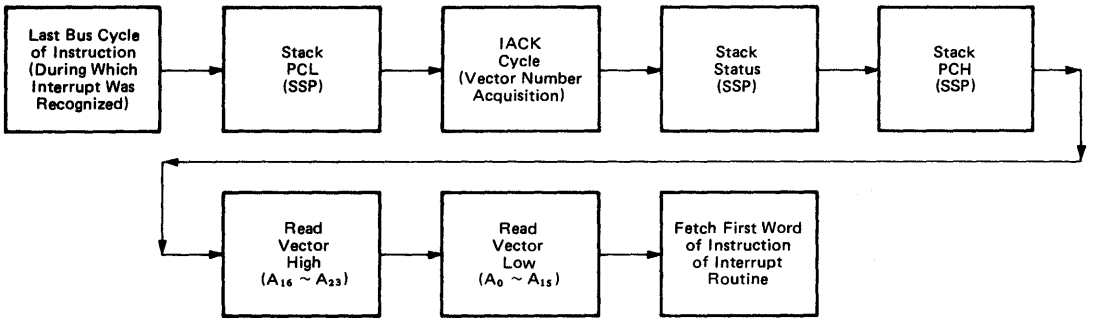


Figure 39 Interrupt Exception Timing Sequence

Priority level seven is a special case. Level seven interrupts cannot be inhibited by the interrupt priority mask, thus providing a "non-maskable interrupt" capability. An interrupt is generated each time the interrupt request level changes from some lower level to level seven. Note that a level seven interrupt may still be caused by the level comparison if the request level is a seven and the processor priority is set to a lower level by an instruction.

UNINITIALIZED INTERRUPT

An interrupting device asserts $\overline{VPA}$ or provides an interrupt vector during an interrupt acknowledge cycle to the HD68000. If the vector register has not been initialized, the responding HMCS68000 Family peripheral will provide vector 15, the unitialized interrupt vector. This provides a uniform way to recover from a programming error.

SPURIOUS INTERRUPT

If during the interrupt acknowledge cycle no device responds by asserting $\overline{DTACK}$ or $\overline{VPA}$, the bus error line should be asserted to terminate the vector acquisition. The processor separates the processing of this error from bus error by fetching the spurious interrupt vector instead of the bus error vector. The processor then proceeds with the usual exception processing.

INSTRUCTION TRAPS

Traps are exceptions caused by instructions. They arise either from processor recognition of abnormal conditions during instruction execution, or from use of instructions whose normal behavior is trapping.

Some instructions are used specifically to generate traps. The TRAP instruction always forces an exception, and is useful for implementing system calls for user programs. The TRAPV and CHK instructions force an exception if the user program detects a runtime error, which may be an arithmetic overflow or a subscript out of bounds.

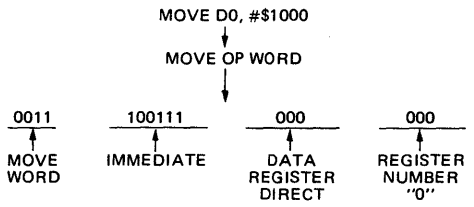
The signed divide (DIVS) and unsigned divide (DIVU) instructions will force an exception if a division operation is attempted with a divisor of zero.

ILLEGAL AND UNIMPLEMENTED INSTRUCTIONS

Illegal instruction is the term used to refer to any of the word bit patterns which are not the bit pattern of the first word of a legal instruction. During instruction execution, if such an instruction is fetched, an illegal instruction exception occurs.

Word patterns with bits 15 through 12 equaling 1010 or 1111 are distinguished as unimplemented instructions and separate exception vectors are given to these patterns to permit efficient emulation. This facility allows the operating system to detect program errors, or to emulate unimplemented instructions in software.

ILLEGAL INSTRUCTION EXAMPLE



PRIVILEGE VIOLATIONS

In order to provide system security, various instructions are privileged. An attempt to execute one of the privileged instructions while in the user state will cause an exception. The privileged instruction are:

STOP	AND (word) Immediate to SR
RESET	EOR (word) Immediate to SR
RTE	OR (word) Immediate to SR
MOVE to SR	MOVE USP

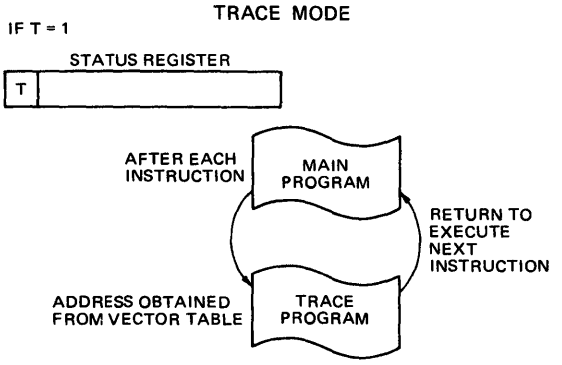
TRACING

To aid in program development, the HD68000 includes a facility to allow instruction by instruction tracing. In the trace state, after each instruction is executed an exceptions is forced, allowing a debugging program to monitor the execution of the program under test.

The trace facility uses the T-bit in the supervisor portion of the status register. If the T-bit is negated (off), tracing is disabled, and instruction execution proceeds from instruction to instruction as normal. If the T-bit is asserted (on) at the beginning of the execution of an instruction, a trace exception will be generated after the execution of that instruction is completed. If the instruction is not executed. either because an interrupt is taken, or the instruction is illegal or privileged, the trace exception does not occur. The trace exception also does not occur if the instruction is aborted by a reset, bus

error, or address error exception. If the instruction is indeed executed and an interrupt is pending on completion, the trace exception is processed before the interrupt exception. If, during the execution of the instruction, an exception is forced by that instruction, the forced exception is processed before the trace exception.

As an extreme illustration of the above rules, consider the arrival of an interrupt during the execution of a TRAP instruction while tracing is enabled. First the trap exception is processed, then the trace exception, and finally the interrupt exception. Instruction execution resumes in the interrupt handler routine.



1. If, upon completion of an instruction, T = 1, go to trace exception processing.
2. Execute trace exception sequence.
3. Execute trace service routine.
4. At the end of the service routine, execute return from exception (RTE).

BUS ERROR

Bus error exceptions occur when the external logic requests that a bus error be processed by an exception. The current bus cycle which the processor is making is then aborted. Whether the processor was doing instruction or exception processing, that processing is terminated, and the processor immediately begins exception processing.

Exception processing for bus error follows the usual sequence of steps. The status register is copied, the supervisor state is entered, and the trace state is turned off. The vector number is generated to refer to the bus error vector. Since the processor was not between instructions when the bus error exception request was made, the context of the processor is more detailed. To save more of this context, additional information is saved on the supervisor stack. The program counter and the copy of the status register are of course saved. The value saved for the program counter is advanced by some amount, two to ten bytes beyond the address of the first word of the instruction which made the reference causing the bus error. If the bus error occurred during the fetch of the next instruction, the saved program counter has a value in the vicinity of the current instruction, even if the current instruction is a branch, a jump, or a return instruction. Besides the usual information, the processor saves its internal copy of the first word of the instruction being processed, and the address which was being accessed by the aborted bus cycle. Specific information about the access is also saved: whether it was a read or a write, whether the processor was processing an instruction or not, and the classification displayed on the function code outputs when the bus error occurred. The processor is processing an instruction if it is in the normal state or processing a Group 2 exception; the processor is not processing an instruction if it is processing a Group 0 or a Group 1 exception. Figure 40 illustrates how this information is organized on the supervisor stack. Although this information is not sufficient in general to effect full recovery from the bus error, it does allow software diagnosis. Finally, the processor commences instruction processing at the address contained in the vector. It is the responsibility of the error handler routine to clean up the stack and determine where to continue execution.

If a bus error occurs during the exception processing for a bus error, address error, or reset, the processor is halted, and all processing cases. This simplifies the detection of catastrophic system failure, since the processor removes itself from the system rather than destroy all memory contents. Only the RESET pin can restart a halted processor.

ADDRESS ERROR

Address error exceptions occur when the processor attempts to access a word or a long word operand or an instruction at an odd address. The effect is much like an internally generated bus error, so that the bus cycle is aborted, and the processor ceases whatever processing it is currently doing and begins exception processing. After exception processing commences, the sequence is the same as that for bus error including the information that is stacked, except that the vector number refers to the address error vector instead. Likewise, if an address error occurs during the exception processing for a bus error, address error, or reset, the processor is halted. As shown in Figure 42, an address error will execute a short bus cycle followed by exception processing.

■ INTERFACE WITH HMCS6800 PERIPHERALS

Hitachi's extensive line of HMCS6800 peripherals are directly compatible with the HD68000. Some of these devices that are particularly useful are:

HD6821	Peripheral Interface Adapter
HD6843	Floppy Disk Controller
HD6845S	CRT Controller
HD46508	Data Acquisition Unit
HD6850	Asynchronous Communication Interface Adapter
HD6852	Synchronous Serial Data Adapter

To interface the synchronous HMCS6800 peripherals with the asynchronous HD68000, the processor modifies its bus cycle to meet the HMCS6800 cycle requirements whenever an HMCS6800 device address is detected. This is possible since both processors use memory mapped I/O. Figure 44 is a flow chart of the interface operation between the processor and HMCS6800 devices.

• DATA TRANSFER OPERATION

Three signal on the processor provide the HMCS6800 interface. They are: enable (E), valid memory address ($\overline{\text{VMA}}$), and valid peripheral address ($\overline{\text{VPA}}$). Enable corresponds to the E or ϕ_2 signal in existing HMCS6800 systems. The bus frequency is one tenth of the incoming HD68000 clock frequency. The timing of E allows 1 MHz peripherals to be used with an 8 MHz HD68000. Enable has a 60/40 duty cycle; that is, it is low for six input clocks and high for four input clocks. This duty cycle allows the processor to do successive VPA accesses on successive E pulses.

HMCS6800 cycle timing is given in Figure 45 and 46. At

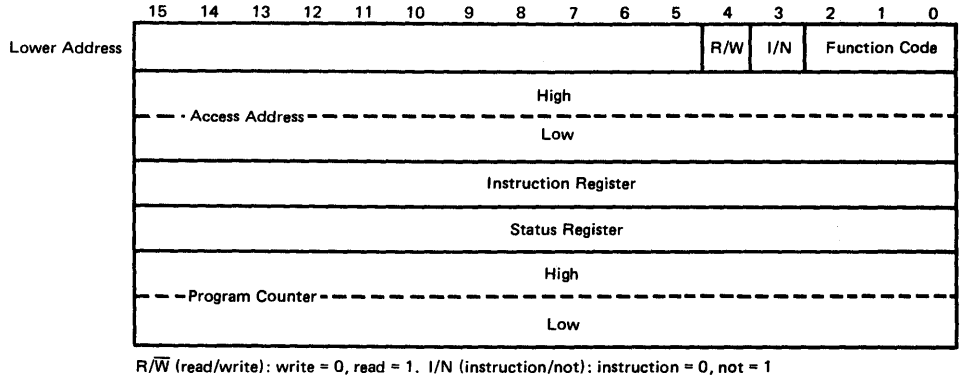


Figure 40 Supervisor Stack Order (Group 0)

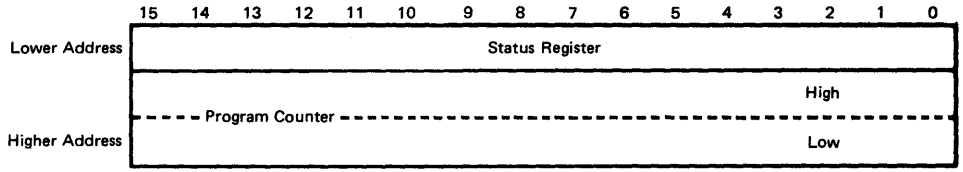


Figure 41 Supervisor Stack Order (Group 1, 2)

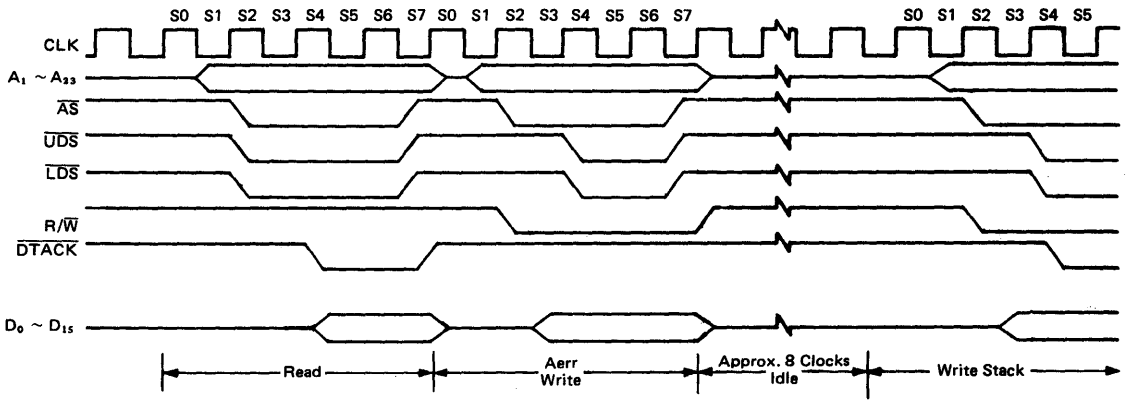


Figure 42 Address Error Timing

state zero (S0) in the cycle, the address bus is in the high-impedance state. A function code is asserted on the function code output lines. One-half clock later, in state 1 the address bus is released from the high-impedance state.

During state 2, the address strobe ($\overline{AS}$) is asserted to indicate that there is a valid address on the address bus. If the bus cycle is a read cycle, the upper and/or lower data strobes are also asserted in state 2. If the bus cycle is a write cycle,

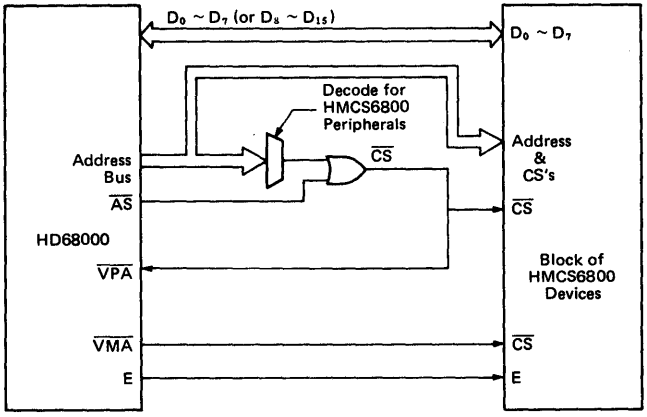


Figure 43 Connection of HMCS6800 Peripherals

the read/write (R/$\overline{W}$) signal is switched to low (write) during state 2. One half clock later, in state 3, the write data is placed on the data bus, and in state 4 the data strobes are issued to indicate valid data on the data bus. The processor now inserts wait states until it recognizes the assertion of $\overline{VPA}$.

The $\overline{VPA}$ input signals the processor that the address on the bus is the address of an HMCS6800 device (or an area reserved for HMCS6800 devices) and that the bus should conform to the ϕ_2 transfer characteristics of the HMCS6800 bus. Valid peripheral address is derived by decoding the address bus, conditioned by address strobe.

After the recognition of $\overline{VPA}$, the processor assures that the Enable (E) is low, by waiting if necessary, and subsequently asserts $\overline{VMA}$. Valid memory address is then used as part of the chip select equation of the peripheral. This ensures that the HMCS6800 peripherals are selected and deselected at the correct time. The peripheral now runs its cycle during the high portion of the E signal. Figures 45 and 46 depict the best and worst case HMCS6800 cycle timing. This cycle length is dependent strictly upon when $\overline{VPA}$ is asserted in relationship to the E clock.

During a read cycle, the processor latches the peripheral data in state 6. For all cycles, the processor negates the address and data strobes one half clock cycle later in state 7, and the Enable signal goes low at this time. Another half clock later, the address bus is put in the high-impedance state. During a write cycle, the data bus is put in the high-impedance state and the read/write signal is switched high. The peripheral logic must remove $\overline{VPA}$ within one clock after address strobe is negated.

Figure 47 shows the timing required by HMCS6800 peripherals, the timing specified for HDCS6800, and the corresponding timing for the HD68000. Two example systems with HMCS6800 peripherals are showin in Figures 48 and 49. The system in Figure 48 reserves the upper eight megabytes of memory for HMCS6800 peripherals. The system in Figure 49 is more efficient with memory and easily expandable, but more complex.

$\overline{DTACK}$ should not be asserted while $\overline{VPA}$ is asserted. Notice that the HD68000 $\overline{VMA}$ is active low, contrasted with the active high HMCS6800 VMA. This allows the processor to put its buses in the high-impedance state on DMA requests without inadvertently selecting peripherals.

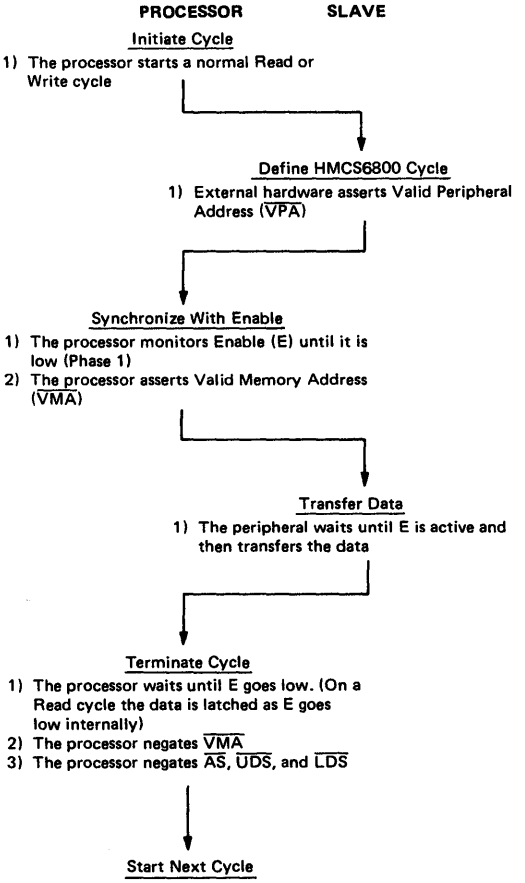
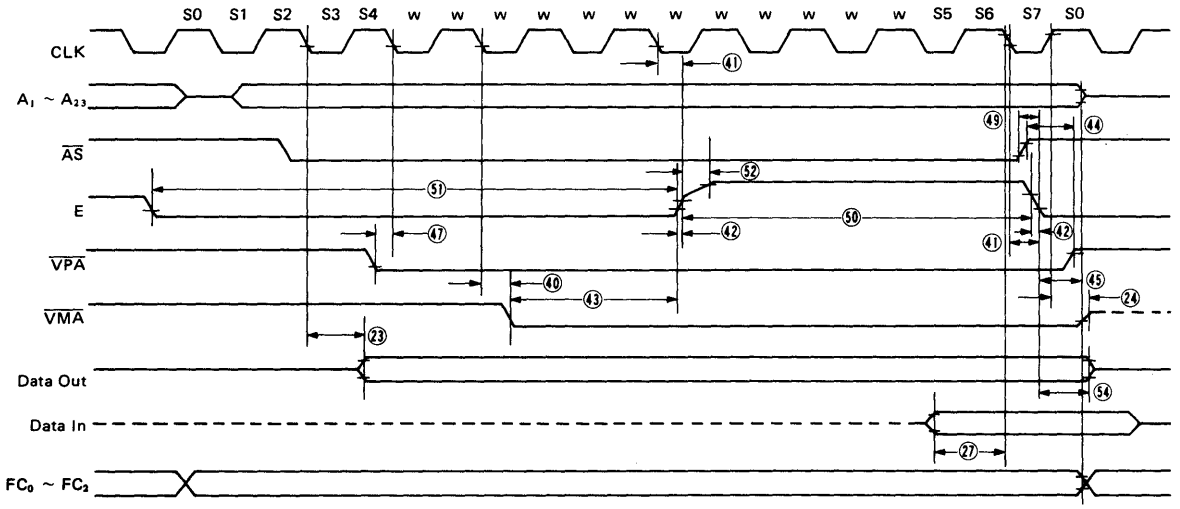


Figure 44 HMCS6800 Interface Flow Chart



(NOTE) This figure represents the best case HMCS6800 timing where $\overline{VPA}$ falls before the third system clock cycle after the falling edge of E.

Figure 45 HMCS6800 Timing – Best Case

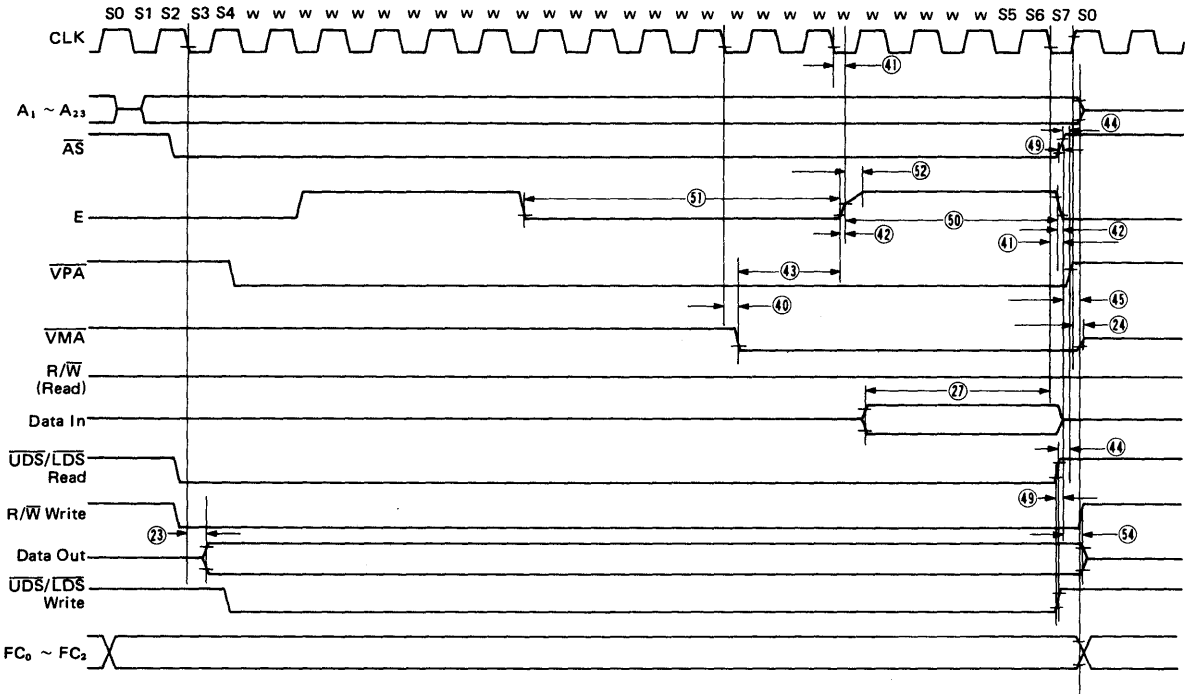
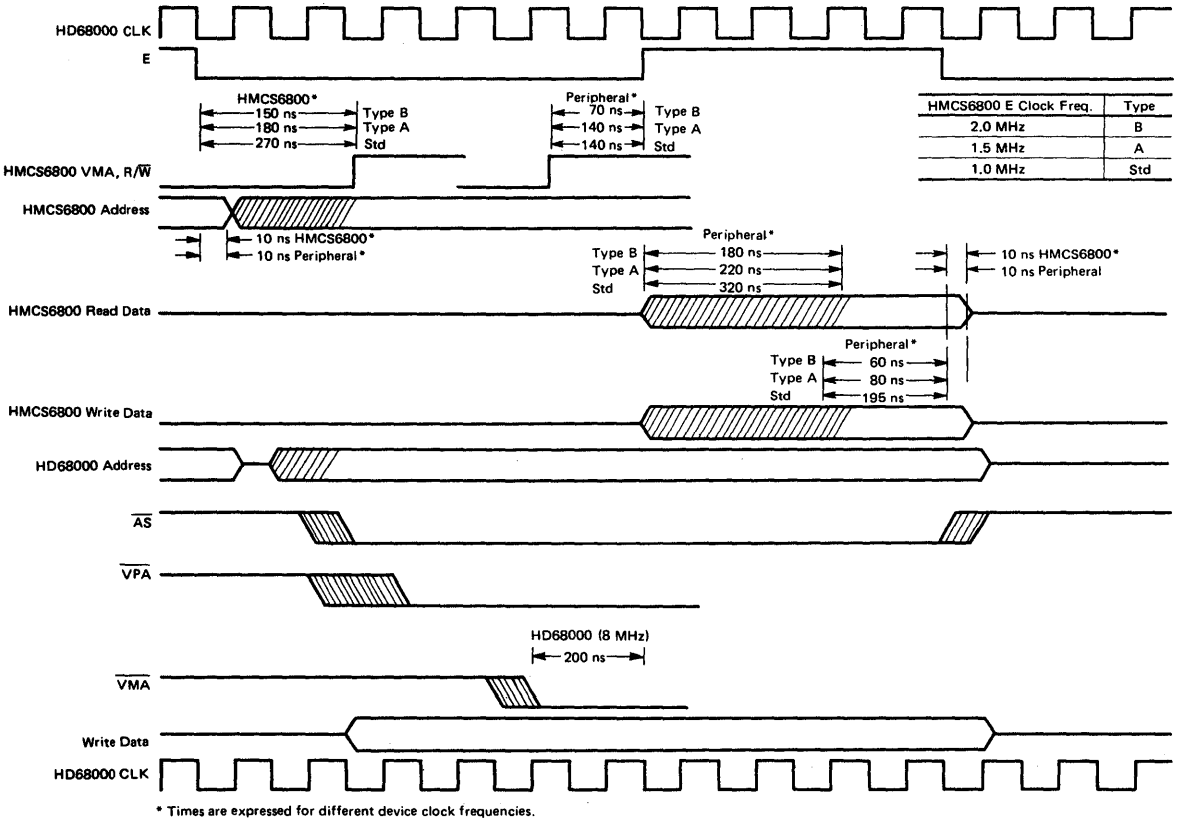


Figure 46 HMCS6800 Timing – Worst Case



* Times are expressed for different device clock frequencies.

Figure 47 HD68000 to HMCS6800 Peripheral Timing Diagram

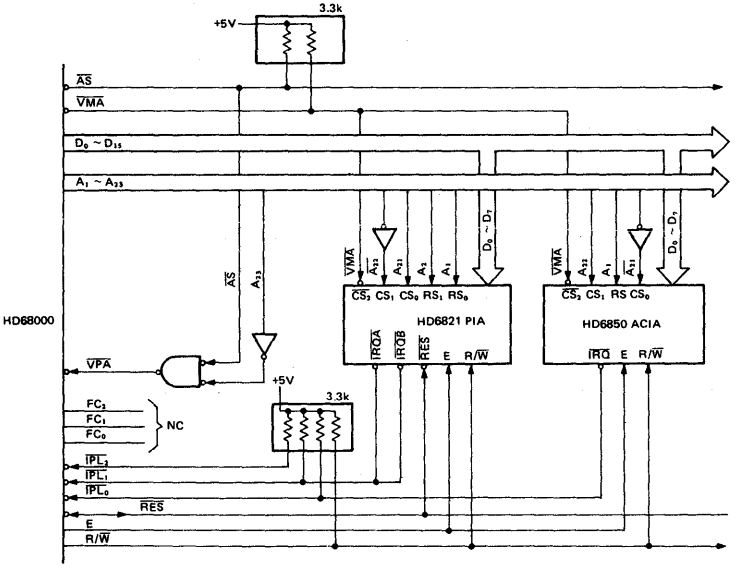


Figure 48 HMCS6800 Interface – Example 1

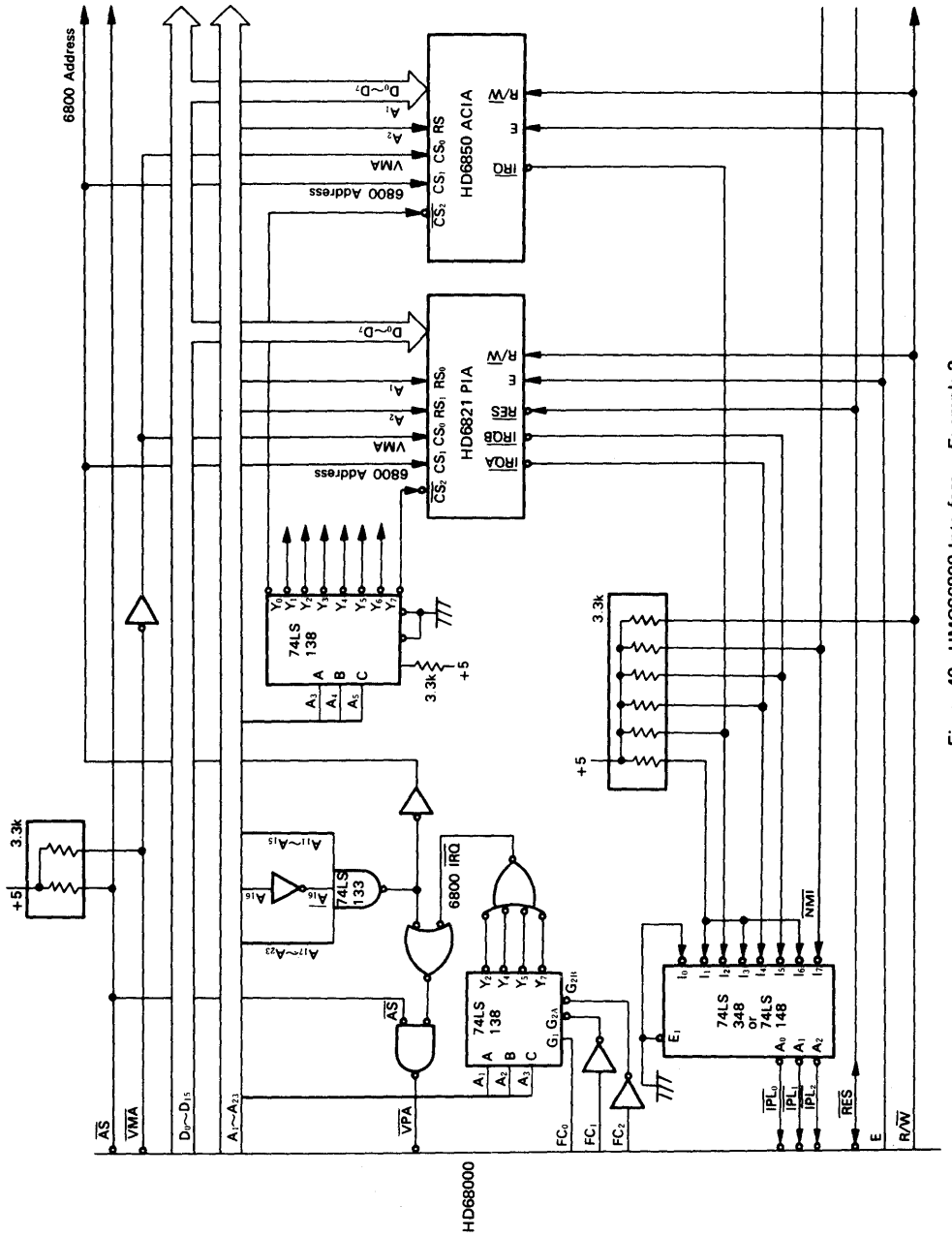


Figure 49 HMCS6800 Interface – Example 2

- **INTERRUPT OPERATION**

During an interrupt acknowledge cycle while the processor is fetching the vector, if $\overline{VPA}$ is asserted, the HD68000 will assert $\overline{VMA}$ and complete a normal HMCS6800 read cycle as shown in Figure 50. The processor will then use an internally generated vector that is a function of the interrupt being serviced. This process is known as autovectoring. The seven autovectors are vector numbers 25 through 31 (decimal).

This operates in the same fashion (but is not restricted to) the HMCS6800 interrupt sequence. The basic difference is that there are six normal interrupt vectors and one NMI type vector. As with both the HMCS6800 and the HD68000's normal vectored interrupt, the interrupt service routine can be located anywhere in the address space. This is due to the fact that while the vector numbers are fixed, the contents of the vector table entries are assigned by the user.

Since $\overline{VMA}$ is asserted during autovectoring, the HMCS6800 peripheral address decoding should prevent unintended accesses.

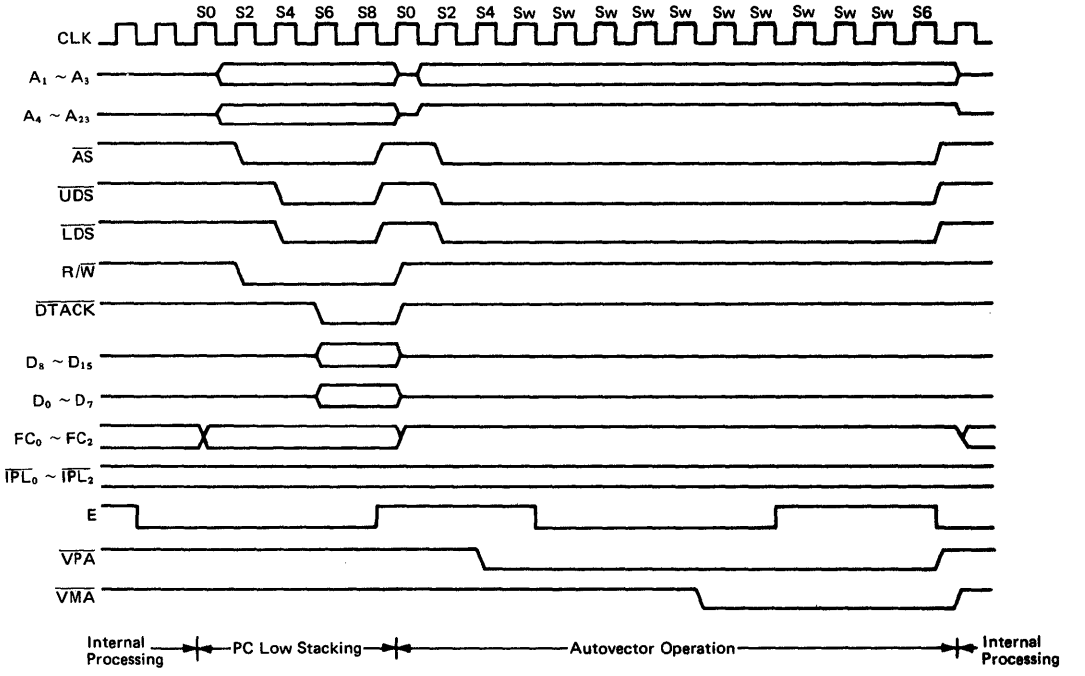


Figure 50 Autovector Operation Timing Diagram

- **DATA TYPES AND ADDRESSING MODES**

Five basic data types are supported. These data types are:

- Bits
- BCD Digits (4-bits)
- Bytes (8-bits)
- Word (16-bits)
- Long Words (32-bits)

In addition, operations on other data types such as memory addresses, status word data, etc., are provided for in the instruction set.

The 14 addressing modes, shown in Table 10, includs six basic types:

- Register Direct
- Register Indirect
- Absolute
- Immediate
- Program Counter Relative
- Implied

Included in the register indirect addressing modes is the capability to do postincrementing, predecrementing, offsetting and indexing. Program counter relative mode can also be modified via indexing and offsetting.

Table 10 Addressing Modes

Mode	Generation
Register Direct Addressing	
Data Register Diredt	EA = Dn
Address Register Direct	EA = An
Absolute Data Addressing	
Absolute Short	EA = (Next Word)
Absolute Long	EA = (Next Two Words)
Program Counter Relative Addressing	
Relative with Offset	EA = (PC) + d_{16}
Relative with Index and Offset	EA = PC) + (Xn) + d_8
Register Indirect Addressing	
Register Indirect	EA = (An)
Postincrement Register Indirect	EA = (AN), An ← An + N
Predecrement Register Indirect	An ← An − N, EA = (An)
Register Indirect with Offset	EA = (An) + d_{16}
Indexed Register Indirect with Offset	EA = (An) + (Xn) + d_8
Immediate Data Addressing	
Immediate	DATA = Next Word(s)
Quick Immediate	Inherent Data
Implied Addressing	
Implied Register	EA = SR, USP, SP, PC

(NOTES)

EA = Effective Address
An = Address Register
Dn = Data Register
Xn = Address or Data Register used as Index Register
SR = Status Register
PC = Program Counter
( ) = Contents of
d_8 = Eight-bit Offset (displacement)
d_{16} = Sixteen-bit Offset (displacement)
N = 1 for Byte, 2 for Words and 4 for Long Words
← = Replaces

■ INSTRUCTION SET OVERVIEW

The HD68000 instruction set is shown in Table 11. Some additional instructions are variations, or subsets, of these and they appear in Table 12. Special emphasis has been given to the instruction set's support of structured high-level languages to facilitate ease of programming. Each instruction, with few exceptions, operates on bytes, words, and long words and most instructions can use any of the 14 addressing modes. Combining instruction types, data types, and addressing modes, over 1000 useful instructions are provided. These instructions include signed and unsigned multiply and divide, "quick" arithmetic operations, BCD arithmetic and expanded operations (through traps).

The following paragraphs contain an overview of the form and structure of the HD68000 instruction set. The instructions form a set of tools that include all the machine functions to perform the following operations:

Data Movement
Integer Arithmetic
Logical
Shift and Rotate
Bit Manipulation
Binary Coded Decimal
Program Control
System Control

The complete range of instruction capabilities combined with the flexible addressing modes described previously provide a very flexible base for program development.

Table 11 Instruction Set

Mnemonic	Description
ABCD	Add Decimal with Extend
ADD	Add
AND	Logical And
ASL	Arithmetic Shift Left
ASR	Arithmetic Shift Right
B_{CC}	Branch Conditionally
BCHG	Bit Test and Change
BCLR	Bit Test and Clear
BRA	Branch Always
BSET	Bit Test and Set
BSR	Branch to Subroutine
BTST	Bit Test
CHK	Check Register Against Bounds
CLR	Clear Operand
CMP	Compare
DB_{CC}	Test Condition, Decrement and Branch
DIVS	Signed Divide
DIVU	Unsigned Divide
EOR	Exclusive Or
EXG	Exchange Registers
EXT	Sign Extend
JMP	Jump
JSP	Jump to Subroutine
LEA	Load Effective Address
LINK	Link Stack
LSL	Logical Shift Left
LSR	Logical Shift Right
MOVE	Move
MOVEM	Move Multiple Registers
MOVEP	Move Peripheral Data
MULS	Signed Multiply
MULU	Unsigned Multiply
NBCD	Negate Decimal with Extend
NEG	Negate
NOP	No Operation
NOT	One's Complement
OR	Logical Or
PEA	Push Effective Address
RESET	Reset External Devices
ROL	Rotate Left without Extend
ROR	Rotate Right without Extend
ROXL	Rotate Left with Extend
ROXR	Rotate Right with Extend
RTE	Return from Exception
RTR	Return and Restore
RTS	Return from Subroutine
SBCD	Subtract Decimal with Extend
S_{CC}	Set Conditional
STOP	Stop
SUB	Subtract
SWAP	Swap Data Register Halves
TAS	Test and Set Operand
TRAP	Trap
TRAPV	Trap on Overflow
TST	Test
UNLK	Unlink

Table 12 Variations of Instruction Types

Instruction Type	Variation	Description
ADD	**ADD**	Add
	ADDA	Add Address
	ADDQ	Add Quick
	ADDI	Add Immediate
	ADDX	Add with Extend
AND	**AND**	Logical And
	ANDI	And Immediate
CMP	**CMP**	Compare
	CMPA	Compare Address
	CMPM	Compare Memory
	CMPI	Compare Immediate
EOR	**EOR**	Exclusive Or
	EORI	Exclusive Or Immediate
MOVE	**MOVE**	Move
	MOVEA	Move Address
	MOVEQ	Move Quick
	MOVE from SR	Move from Status Register
	MOVE to SR	Move to Status Register
	MOVE to CCR	Move to Condition Codes
	MOVE USP	Move User Stack Pointer
NEG	**NEG**	Negate
	NEGX	Negate with Extend
OR	**OR**	Logical Or
	ORI	Or Immediate
SUB	**SUB**	Subtract
	SUBA	Subtract Address
	SUBI	Subtract Immediate
	SUBQ	Subtract Quick
	SUBX	Subtract with Extend

• ADDRESSING

Instructions for the HD68000 contain two kinds of information: the type of function to be performed, and the location of the operand(s) on which to perform that function. The methods used to locate (address) the operand(s) are explained in the following paragraphs.

Instructions specify an operand location in one of three ways:

Register Specification – the number of the register is given in the register field of the instruction.

Effective Address – use of the different effective address modes.

Implicit Reference – the definition of certain instructions implies the use of specific registers.

• DATA MOVEMENT OPERATIONS

The basic method of data acquisition (transfer and storage) is provided by the move (MOVE) instruction. The move instruction and the effective addressing modes allow both address and data manipulation. Data move instructions allow byte, word, and long word operands to be transferred from memory to memory, memory to register, register to memory, and register to memory, and register to register. Address move instructions allow word and long word operand transfers and ensure that only legal address manipulations are executed. In addition to the general move instruction there are several special data movement instructions: move multiple registers (MOVEM), move peripheral data (MOVEP), exchange registers (EXG), load effective address (LEA), push effective address (PEA), link stack (LINK), unlink stack (UNLK), and move quick (MOVEQ). Table 13 is a summary of the data movement operations.

• INTEGER ARITHMETIC OPERATIONS

The arithmetic operations include the four basic operations of add (ADD), subtract (SUB), multiply (MUL), and divide (DIV) as well as arithmetic compare (CMP), clear (CLR), and negate (NEG). The add and subtract instructions are available for both address and data operations, with data operations accepting all operand sizes. Address operations are limited to legal address size operands (16 or 32 bits). Data, address, and memory compare operations are also available. The clear and negate instructions may be used on all sizes of data operands.

The multiply and divide operations are available for signed and unsigned operands using word multiply to produce a long word product, and a long word dividend with word divisor to produce a word quotien with a word remainder.

Multiprecision and mixed size arithmetic can be accomplished using a set of extended instructions. These instructions are: add extended (ADDX), subtract extended (SUBX), sign extend (EXT), and negate binary with extend (NEGX).

A test operand (TST) instruction that will set the condition codes as a result of a compare of the operand with zero is also available. Test and set (TAS) is a synchronization instruction useful in multiprocessor systems. Table 14 is a summary of the integer arithmetic operations.

Table 13 Data Movement Operations

Instruction	Operand Size	Operation
EXG	32	Rx ↔ Ry
LEA	32	EA → An
LINK	–	An → SP@- SP → An SP + d → SP
MOVE	8, 16, 32	(EA)s → EAd
MOVEM	16, 32	(EA) → An, Dn An, Dn → EA
MOVEP	16, 32	(EA) → Dn Dn → EA
MOVEQ	8	#xxx → Dn
PEA	32	EA → SP@-
SWAP	32	Dn[31:16] ↔ Dn[15:0]
UNLK	–	An → Sp SP@+ → An

(NOTES)
s = source
d = destination
[ ] = bit numbers
@- = indirect with predecrement
@+ = indirect with postdecrement

Table 14 Integer Arithmetic Operations

Instruction	Operand Size	Operation
ADD	8, 16, 32	Dn + (EA) → Dn
		(EA + Dn → EA
		(EA) + #xxx → EA
	16, 32	AN + (EA) → An
ADDX	8, 16, 32	Dx + Dy + X → Dx
	16, 32	Ax@ - +Ay@ - + X → Ax@
CLR	8, 16, 32	0 → EA
CMP	8, 16, 32	Dn - (EA)
		(EA) - #xxx
		Ax@+ - Ay@+
	16, 32	An - (EA)
DIVS	32 ÷ 16	Dn/(EA) → Dn
DIVU	32 ÷ 16	Dn/(EA) → Dn
EXT	8 → 16	$(Dn)_8 \rightarrow Dn_{16}$
	16 → 32	$(Dn)_{16} \rightarrow Dn_{32}$
MULS	16*16 → 32	Dn*(EA) → Dn
MULU	16*16 → 32	Dn*(EA) → Dn
NEG	8, 16, 32	0 - (EA) → EA
NEGX	8, 16, 32	0 - (EA) - X - EA
SUB	8, 16, 32	Dn - (EA) → Dn
		(EA) - Dn → EA
		(EA) - #xxx → EA
	16, 32	An - (EA) → An
SUBX	8, 16, 32	Dx - Dy - X → Dx
		Ax@ - - Ay@ - - X → Ax@
TAS	8	(EA) - 0, 1 → EA[7]
TST	8, 16, 32	(EA) - 0

(NOTE) [ ] = bit number

- **INSTRUCTION FORMAT**

Instructions are from one to five words in length, as shown in Figure 51. The length of the instruction and the operation to be performed is specified by the first word of the instruction which is called the operation word. The remaining words further specify the operands. These words are either immediate operands or extensions to the effective address mode specified in the operation word.

- **PROGRAM/DATA REFERENCES**

The HD68000 separates memory references into two classes: program references, and data references. Program references, as the name implies, are references to that section of memory that contains the program being executed. Data references refer to that section of memory that contains data. Generally, operand reads are from the data space. All operand writes are to the data space.

- **REGISTER SPECIFICATION**

The register field within an instruction specifies the register to be used. Other fields within the instruction specify whether the register selected is an address or data register and how the register is to be used.

- **EFFECTIVE ADDRESS**

Most instructions specify the location of an operand by using the effective address field in the operation word. For example, Figure 52 shows the general format of the single effective address is composed of two 3-bit fields: the mode field, and the register field. The value in the mode field selects the different address modes. The register field contains the number of a register.

The effective address field may require additional information to fully specify the operand. This additional information, called the effective address extension, is contained in the following word or words and is considered part of the instruction, as shown in Figure 51. The effective address modes are grouped into three categories: register direct, memory addressing, and special.

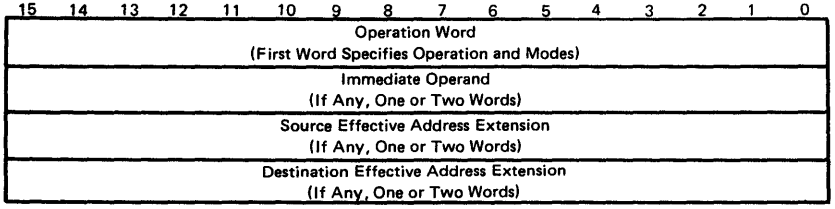


Figure 51 Instruction Format

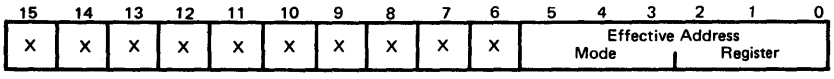


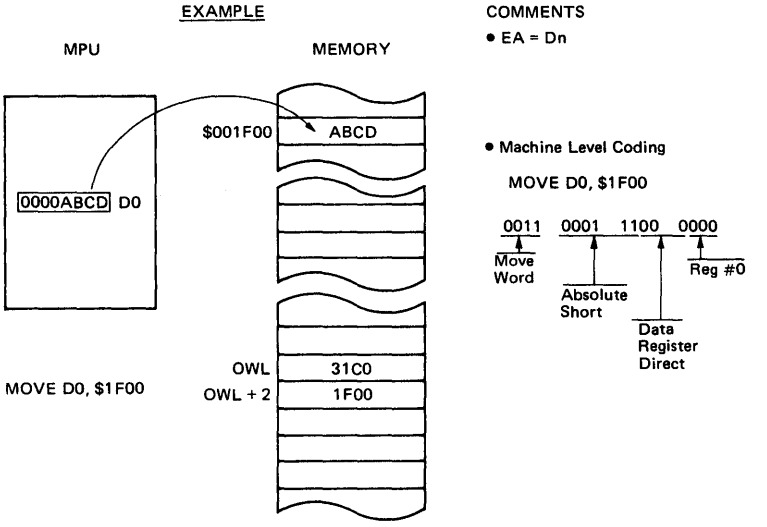
Figure 52 Single-Effective-Address Instruction Operation Word General Format

REGISTER DIRECT MODES

These effective addressing modes specify that the operand is in one of the 16 multifunction registers.

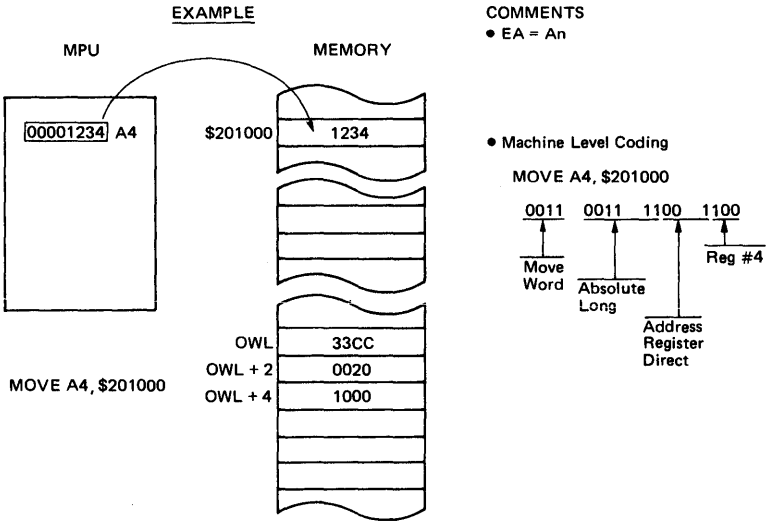
Data Register Direct

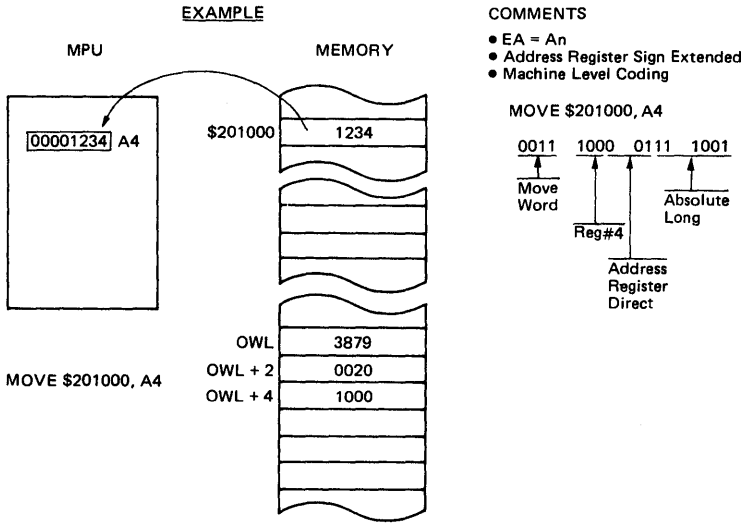
The operand is in the data register specified by the effective address register field.



Address Register Direct

The operand is in the address register specified by the effective address register field.



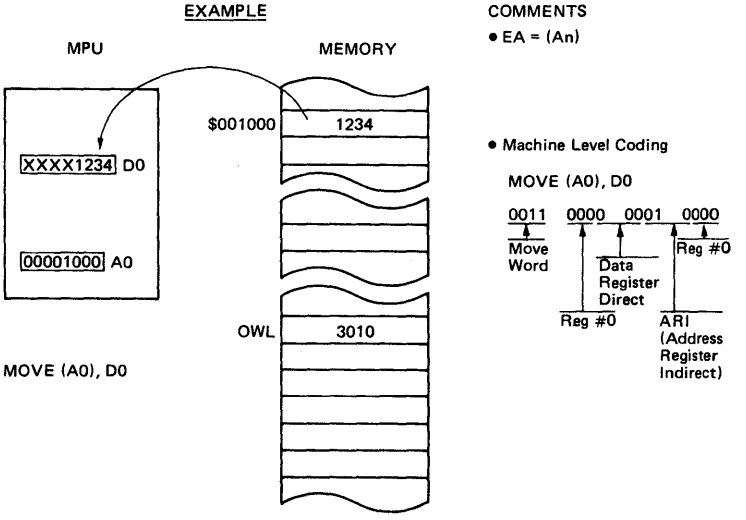


MEMORY ADDRESS MODES

These effective addressing modes specify that the operand is in memory and provide the specific address of the operand.

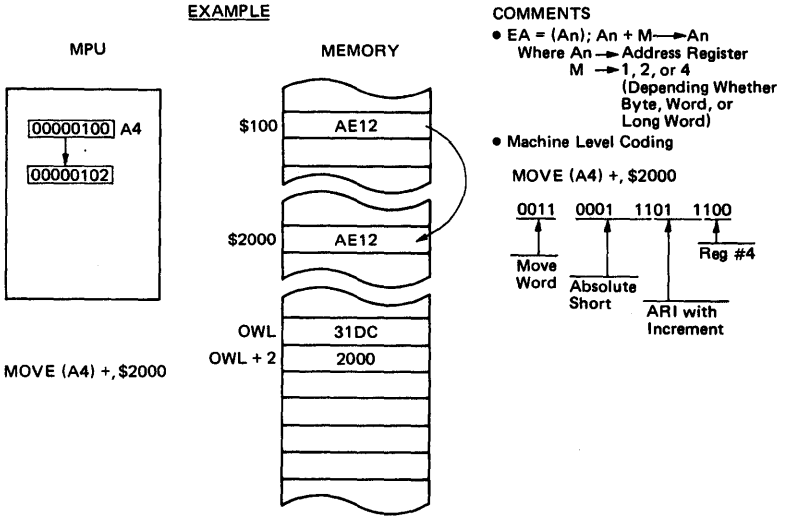
Address Register Indirect

The address of the operand is in the address register specified by the register field. The reference is classified as a data reference with the exception of the jump and jump to subroutine instructions.



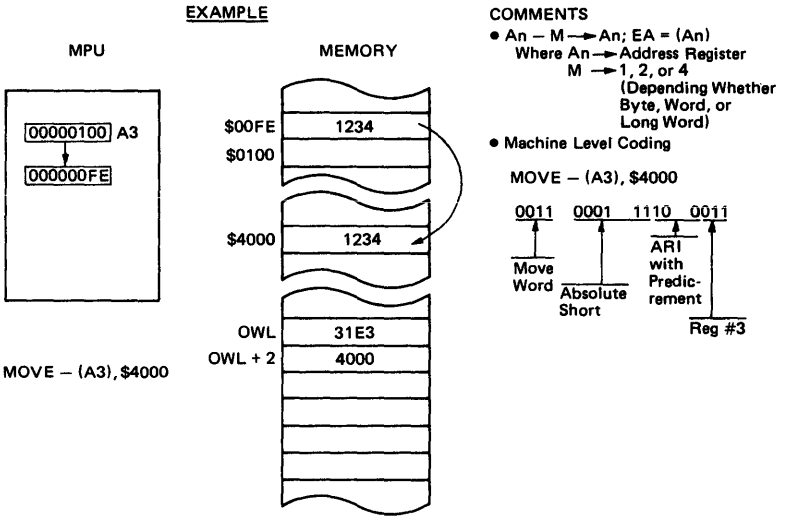
Address Register Indirect With Postincrement

The address of the operand is in the address register specified by the register field. After the operand address is used, it is incremented by one, two, or four depending upon whether the size of the operand is byte, word, or long word. If the address register is the stack pointer and the operand size is byte, the address is incremented by two rather than one to keep the stack pointer on a word boundary. The reference is classified as a data reference.



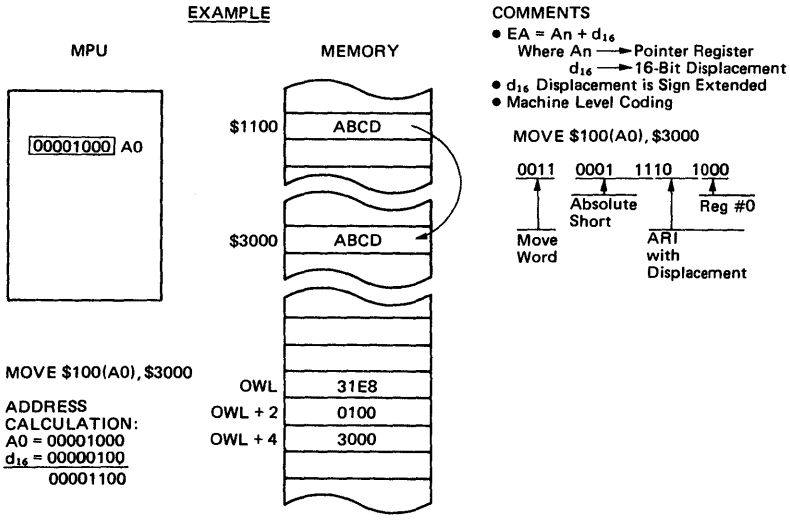
Address Register Indirect With Predecrement

The address of the operand is in the address register specified by the register field. Before the operand address is used, it is decremented by one, two, or four depending upon whether the operand size is byte, word, or long word. If the address register is the stack pointer and the operand size is byte, the address is decremented by two rather than one to keep the stack pointer on a word boundary. The reference is classified as a data reference.



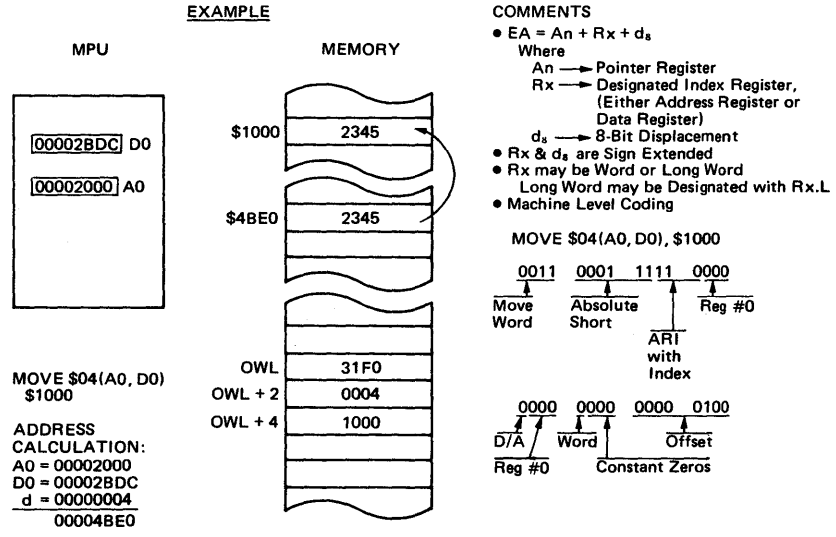
Address Register Indirect With Displacement

This address mode requires one word of extension. The address of the operand is the sum of the address in the address register and the sign-extended 16-bit displacement integer in the extension word. The reference is classified as a data reference with the exception of the jump to subroutine instructions.



Address Register Indirect With Index

This address mode requires one word of extension. The address of the operand is the sum of the address in the address register, the sign-extended displacement integer in the low order eight bits of the extension word, and the contents of the index register. The reference is classified as a data reference with the exception of the jump and jump to subroutine instructions.

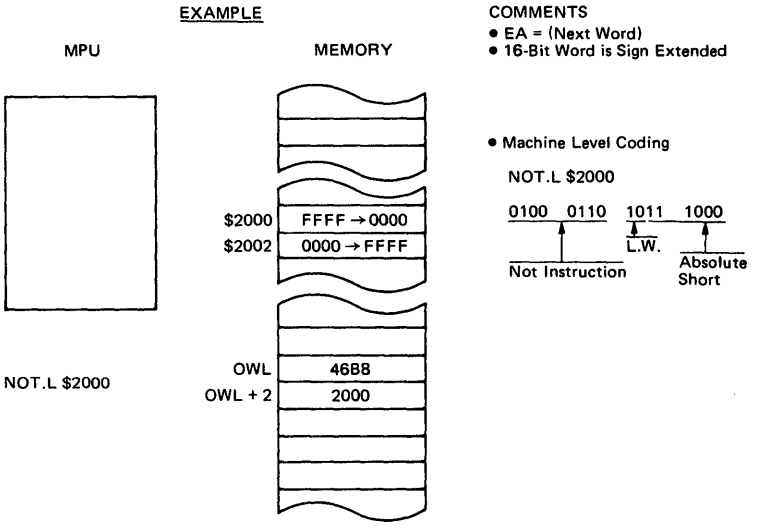


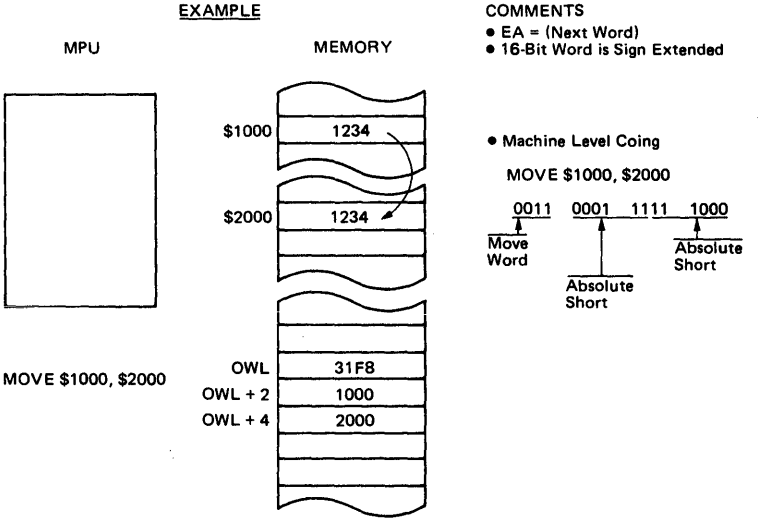
SPECIAL ADDRESS MODE

The special address modes use the effective address register field to specify the special addressing mode instead of a register number.

Absolute Short Address

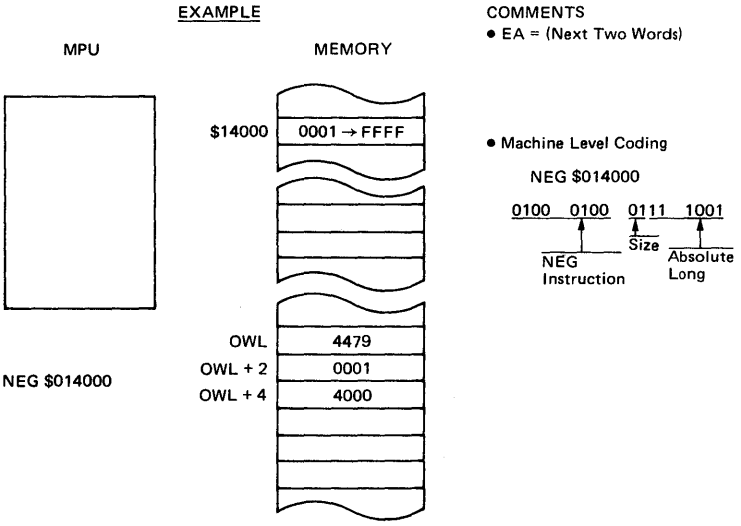
This address mode requires one word of extension. The address of the operand is the extension word. The 16-bit address is sign extended before it is used. The reference is classified as a data reference with the exception of the jump and jump to subroutine instructions.





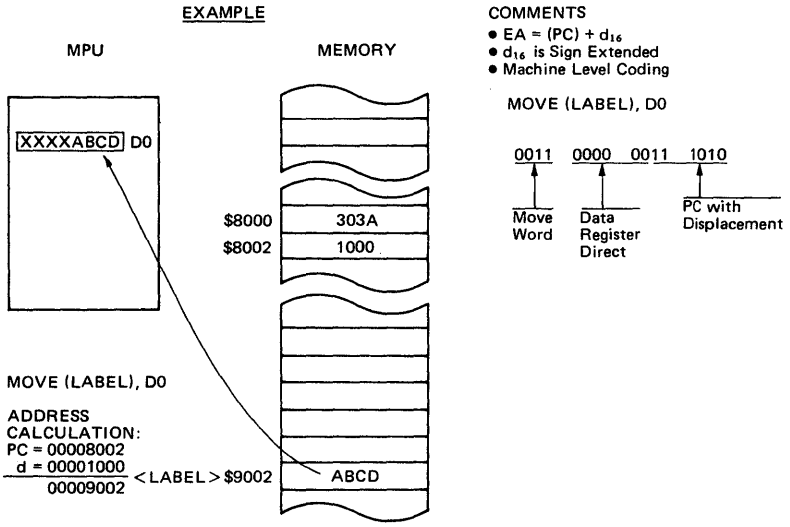
Absolute Long Address

This address mode requires two words of extension. The address of the operand is developed by the concatenation of the extension words. The high-order part of the address is the first extension word; the low-order part of the address is the second extension word. The reference is classified as a data reference with the exception of the jump and jump to subroutine instructions.



Program Counter With Displacement

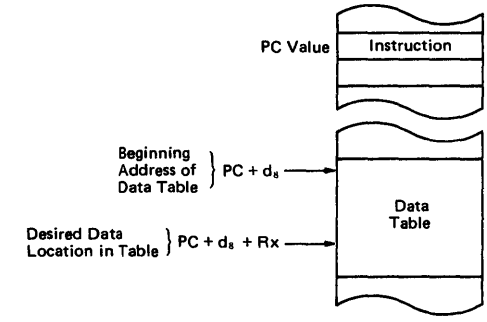
This address mode requires one word of extension. The address of the operand is the sum of the address in the program counter and the sign-extended 16-bit displacement integer in the extension word. The value in the program counter is the address of the extension word. The reference is classified as a program reference.



Program Counter With Index

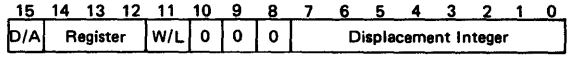
This address mode requires one word of extension. This address is the sum of the address in the program counter, the sign-extended displacement integer in the lower eight bits of the extension word, and the contents of the index register. The value in the program counter is the address of the extension word. This reference is classified as a program reference.

EA = (PC) + (Rx) + d_8



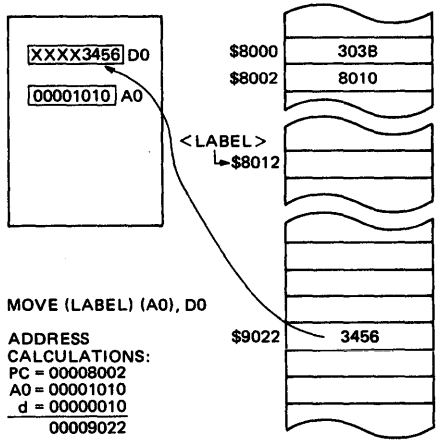
(NOTE)

Extension Word



D/A : Data Register = 0, Address Register = 1
Register : Index Register Number
W/L : Sign-extented, low order Word integer
in Index Register = 0
Long Word in Index Register = 1

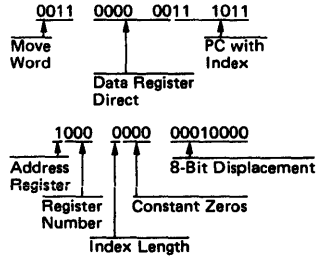
EXAMPLE



COMMENTS

- EA = (PC) + (Rx) + d_8
  Where
  PC → Current Program Counter
  Rx → Designated Index Register
  (Either Data or Address Register)
  d_8 → 8-Bit Displacement
- Rx and d_8 are Sign Extended
- Rx may be Word or Long Word
  Long Word is Designated with Rx.L
- Machine Level Coding

MOVE (LABEL) (A0), D0



Immediate Data

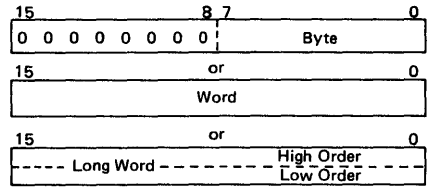
This address mode requires either one or two words of extension depending on the size of the operation.

Byte operation – operand is low order byte of extension word

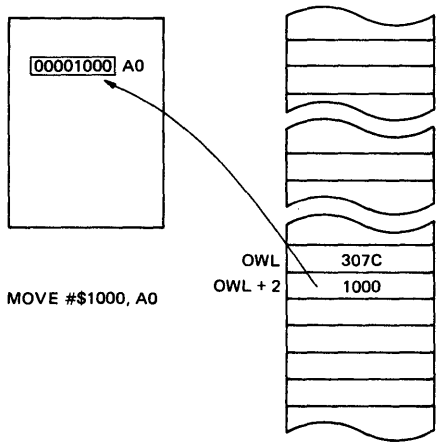
Word operation – operand is extension word

Long word operation – operand is in the two extension words, high-order 16 bits are in the first extension word, low-order 16 bits are in the second extension word.

Extension Word



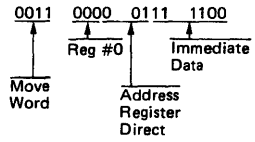
EXAMPLE



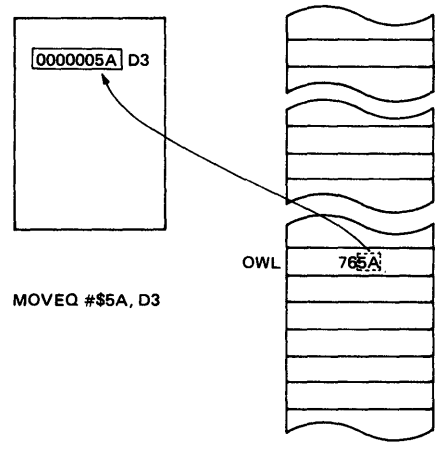
COMMENTS

- Data = Next Word(s)
- Data is Sign Extended for Address Register but not Data Register
- Machine Level Coding

MOVE #$1000, A0



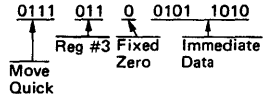
EXAMPLE



COMMENTS

- Inherent Data
- Data is Sign Extended to Long Word
- Destination must be a Data Register
- Machine Level Coding

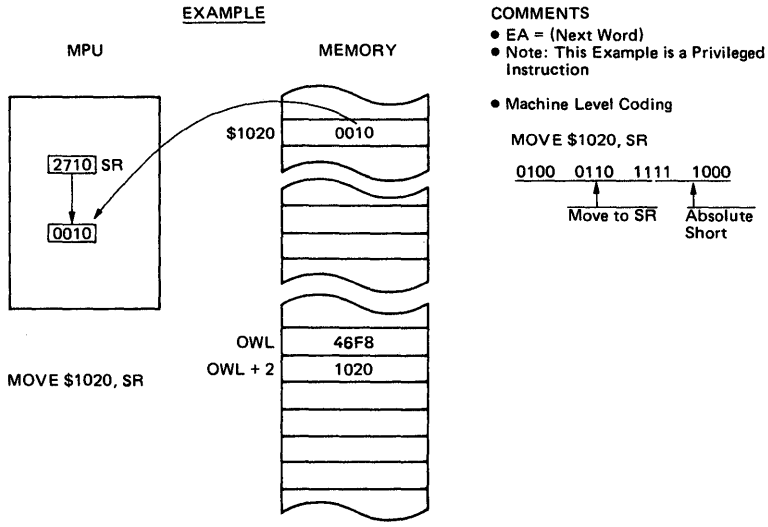
MOVEQ #$5A, D3



Condition Codes or Status Register

A selected set of instructions may reference the status register by means of the effective address field. These are:

ANDI to CCR
ANDI to SR
EORI to CCR
EORI to SR
ORI to CCR
ORI to SR



• EFFECTIVE ADDRESS ENCODING SUMMARY

Table 15 is a summary of the effective addressing modes discussed in the previous paragraphs.

Table 15 Effective Address Encoding Summary

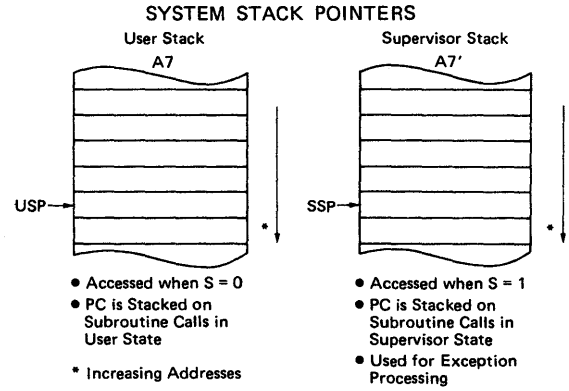
Addressing Mode	Mode	Register
Data Register Direct	000	register number
Address Register Direct	001	register number
Address Register Indirect	010	register number
Address Register Indirect with Postincrement	011	register number
Address Register Indirect with Predecrement	100	register number
Address Register Indirect with Displacement	101	register number
Address Register Indirect with Index	110	register number
Absolute Short	111	000
Absolute Long	111	001
Program Counter with Displacement	111	010
Program Counter with Index	111	011
Immediate	111	100

• IMPLICIT REFERENCE

Some instructions make implicit reference to the program counter (PC), the system stack pointer (SP), the supervisor stack pointer (SSP), the user stack pointer (USP), or the status register (SR).

SYSTEM STACK

The system stack is used implicitly by many instructions; user stacks and queues may be created and maintained through the addressing modes. Address register seven (A7) is the system stack pointer (SP). The system stack pointer is either the supervisor stack pointer (SSP) or the user stack pointer (USP), depending on the state of the S-bit in the status register. If the S-bit indicates supervisor state, SSP is the active system stack pointer, and the USP cannot be referenced as an address register. If the S-bit indicates user state, the USP is the active system stack pointer, and the SSP cannot be referenced. Each system stack fills from high memory to low memory.



The address mode SP @- creates a new item on the active system stack, and the address mode SP @+ deletes an item from the active system stack.

The program counter is saved on the active system stack on subroutine calls, and restored from the active system stack on returns. On the other hand, both the program counter and the status register are saved on the supervisor stack during the processing of traps and interrupts. Thus, the correct execution of the supervisor state code is not dependent on the behavior of user code and user programs may use the user stack pointer arbitrarily.

In order to keep data on the system stack aligned properly, data entry on the stack is restricted so that data is always put in the stack on a word boundary. Thus byte data is pushed on or pulled from the system stack in the high order half of the word; the lower half is unchanged.

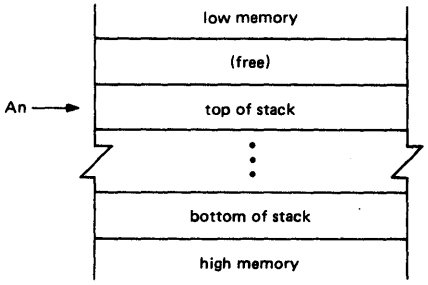
USER STACKS

User stacks can be implemented and manipulated by employing the address register indirect with postincrement and predecrement addressing modes. Using an address register (on of A0 through A6), the user may implement stacks which are filled either from high memory to low memory, or vice versa. The important things to remember are:

- using predecrement, the register is decremented before its contents are used as the pointer into the stack,
- using postincrement, the register is incremented after its contents are used as the pointer into the stack,
- byte data must be put on the stack in pairs when mixed with word or long data so that the stack will not get misaligned when the data is retrieved. Word and long accesses must be on word boundary (even) addresses.

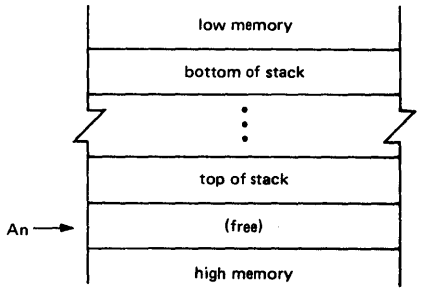
Stack growth from high to low memory is implemented with
An@- to push data on the stack,
An@+ to pull data from the stack.

After eigher a push or a pull operation, register An points to the last (top) item on the stack. This is illustrated as:



Stack growth from low to high memory is implemented with
An@+ to push data on the stack,
An@- to pull data from the stack.

After either a push or a pull operation, register An points to the next available space on the stack. This is illustrated as:

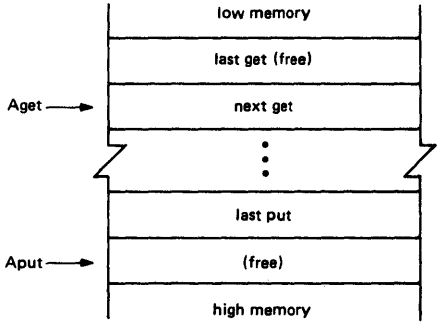


QUEUES

User queues can be implemented and manipulated with the address register indirect with postincrement or predecrement addressing modes. Using a pair of address registers (two of A0 through A6), the user may implement queues which are filled either from high memory to low memory, or vice versa. Because queues are pushed from one end and pulled from the other, two registers are used: the put and get pointers.

Queue growth from low to high memory is implemented with
Aput@+ to put data into the queue,
Aget@+ to get data from the queue.

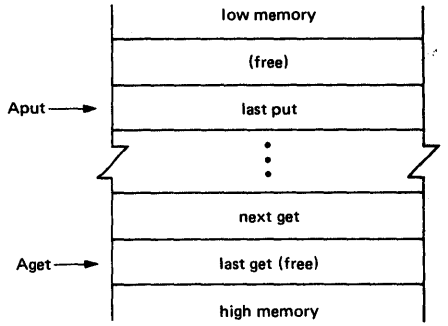
After a put operation, the put address register points to the next available space in the queue and the unchanged get address register points to the next item to remove from the queue. After a get operation, the get address register points to the next item to remove from the queue and the unchanged put address register points to the next available space in the queue. This is illustrated as:



If the queue is to be implemented as a circular buffer, the address register should be checked and, if necessary, adjusted before the put or get operation is performed. The address register is adjusted by subtracting the buffer length (in bytes).

Queue growth from high to low memory is implemented with
Aput@- to put data into the queue,
Aget@ - to get data from the queue.

After a put operation, the put address register points to the last item put in the queue, and the unchanged get address register points to the last item removed from the queue. After a get operation, the get address register points to the last item removed from the queue and the unchanged put address register points to the last item put in the queue. This is illustrated as:



If the queue is to be implemented as a circular buffer, the get or put operation should be performed first, and then the address register should be checked and, if necessary, adjusted. The address register is adjusted by adding the buffer length (in bytes).

• LOGICAL OPERATIONS

Logical operation instructions AND, OR, EOR, and NOT are available for all sizes of integer data operands. A similar set of immediate instructions (ANDI, ORI, and EORI) provide these logical operations with all sizes of immediate data. Table 16 is a summary of the logical operations.

Table 16 Logical Operations

Instruction	Operand Size	Operation
AND	8, 16, 32	Dn ∧ (EA) → Dn (EA) ∧ Dn → EA (EA) ∧ #xxx → EA
OR	8, 16, 32	Dn v (EA) → Dn (EA) v Dn → EA (EA) v #xxx → EA
EOR	8, 16, 32	(EA) ⊕ Dy → EA (EA) ⊕ #xxx → EA
NOT	8, 16, 32	~ (EA) → EA

[NOTE] ~ = invert

• SHIFT AND ROTATE OPERATIONS

Shift operations in both directions are provided by the arithmetic instructions ASR and ASL and logical shift instructions LSR and LSL. The rotate instructions (with and without extend) available are ROXR, ROXL, ROR, and ROL. All shift and rotate operations can be performed in either registers or memory. Register shifts and rotates support all operand sizes and allow a shift count specified in the instruction of one to eight bits, or 0 to 63 specified in a data register.

Memory shifts and rotates are for word operands only and allow only single-bit shifts or rotates. Table 17 is a summary of the shift and rotate operations.

Table 17 Shift and Rotate Operations

Instruction	Operand Size	Operation
ASL	8, 16, 32	X/C ← ← ← 0
ASR	8, 16, 32	→ → → X/C
LSL	8, 16, 32	X/C ← ← ← 0
LSR	8, 16, 32	0 → → → X/C
ROL	8, 16, 32	C ← ← ←
ROR	8, 16, 32	→ → → C
ROXL	8, 16, 32	C ← ← ← X
ROXR	8, 16, 32	X → → → C

• BIT MANIPULATION OPERATIONS

Bit manipulation operations are accomplished using the following instructions: bit test (BTST), bit test and set (BSET), bit test and clear (BCLR), and bit test and change (BCHG). Table 18 is a summary of the bit manipulation operations. (Bit 2 of the status register is Z.)

Table 18 Bit Manipulation Operations

Instruction	Operand Size	Operation
BTST	8, 32	~ bit of (EA) → Z
BSET	8, 32	~ bit of (EA) → Z 1 → bit of EA
BCLR	8, 32	~ bit of (EA) → Z 0 → bit of EA
BCHG	8, 32	~ bit of (EA) → Z ~ bit of (EA) → bit of EA

• BINARY CODED DECIMAL OPERATIONS

Multiprecision arithmetic operations on binary coded decimal numbers are accomplished using the following instructions: add decimal with extend (ABCD), subtract decimal with extend (SBCD), and negate decimal with extend (NBCD). Table 19 is a summary of the binary coded decimal operations.

Table 19 Binary Coded Decimal Operations

Instruction	Operand Size	Operation
ABCD	8	$Dx_{10} + Dy_{10} + X \rightarrow Dx$ $Ax@ -_{10} + Ay@ -_{10} + X \rightarrow Ax@$
SBCD	8	$Dx_{10} - Dy_{10} - X \rightarrow Dx$ $Ax@ -_{10} - Ay@ -_{10} - X \rightarrow Ax@$
NBCD	8	$0 - (EA)_{10} - X \rightarrow EA$

• PROGRAM CONTROL OPERATIONS

Program control operations are accomplished using a series of conditional and unconditional branch instructions and return instructions. These instructions are summarized in Table 20.

The conditional instructions provide setting and branching for the following conditions:

CC – carry clear	LS – low or same
CS – carry set	LT – less than
EQ – equal	MI – minus
F – never true	NE – not equal
GE – greater or equal	PL – plus
GT – greater than	T – always true
HI – high	VC – no overflow
LE – less or equal	VS – overflow

Table 20 Program Control Operations

Instruction	Operation
Conditional	
B_{CC}	Branch conditionally (14 conditions) 8- and 16-bit displacement
DB_{CC}	Test condition, decrement, and branch 16-bit displacement
S_{CC}	Set byte conditionally (16 conditions)
Unconditional	
BRA	Branch always 8-and 16-bit displacement
BSR	Branch to subroutine 8- and 16-bit displacement
JMP	Jump
JSR	Jump to subroutine
Returns	
RTR	Return and restore condition codes
RTS	Return from subroutine

• SYSTEM CONTROL OPERATIONS

System control operations are accomplished by using privileged instructions, trap generating instructions, and instructions that use or modify the status register. These instructions are summarized in Table 21.

Table 21 System Control Operations

Instruction	Operation
Privileged	
RESET	Reset external devices
RTE	Return from exception
STOP	Stop program execution
ORI to SR	Logical OR to status register
MOVE USP	Move user stack pointer
ANDI to SR	Logical AND to status register
EORI to SR	Logical EOR to status register
MOVE EA to SR	Load new status register
Trap Generating	
TRAP	Trap
TRAPV	Trap on overflow
CHK	Check register against bounds
Status Register	
ANDI to CCR	Logical AND to condition codes
EORI to CCR	Logical EOR to condition codes
MOVE EA to CCR	Load new condition codes
ORI to CCR	Logical OR to condition codes
MOVE SR to EA	Store status register

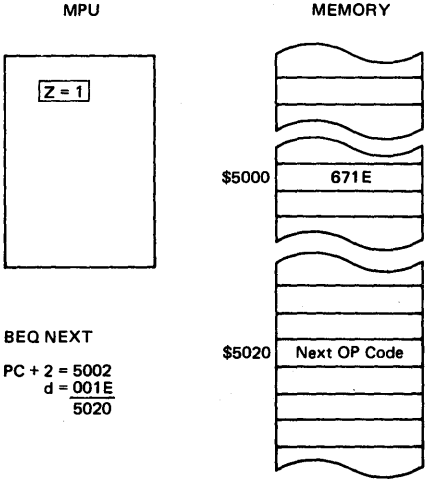
• BRANCH INSTRUCTION ADDRESSING

BRANCH INSTRUCTION FORMAT

	15 – 8	7 – 0
Operation Word	Operation Code	8 bit Displacement
Extension Word	16 bit Displacement if 8 bit Displacement = 0	

RELATIVE, FORWARD REFERENCE, 8-BIT OFFSET

EXAMPLE



COMMENTS
- Offset Contained in 8 LSBs of Op Word
- Offset is 2's Complement Number
- If Offset = 0 then Word Offset is Used
- Machine Level Coding

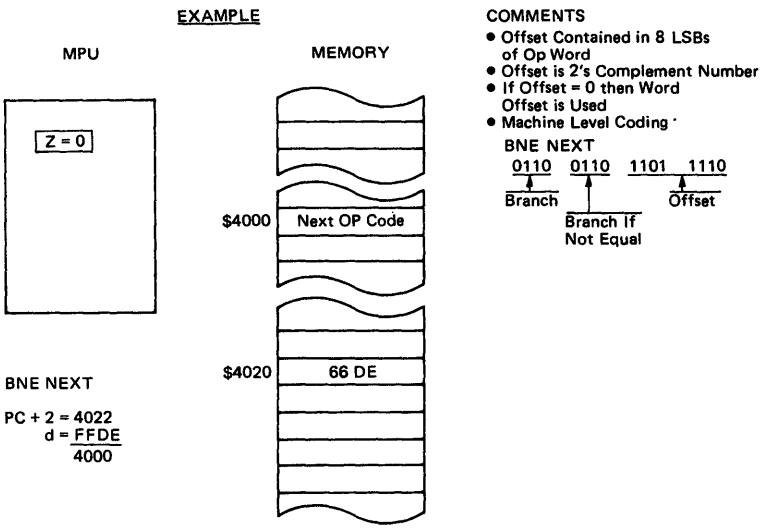
BEQ NEXT

0110 0111 0001 1110

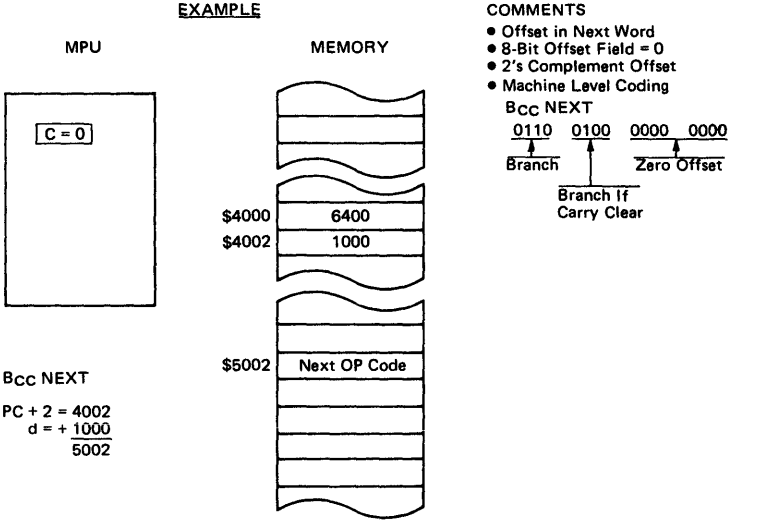
Branch — Offset

Branch If Equal

RELATIVE, BACKWARD REFERENCE 8-BIT OFFSET



RELATIVE, FORWARD REFERENCE, 16-BIT OFFSET



■ CONDITION CODES COMPUTATION

This provides a discussion of how the condition codes were developed, the meanings of each bit, how they are computed, and how they are represented in the instruction set details.

• CONDITION CODE REGISTER

The condition code register portion of the status register contains five bits:

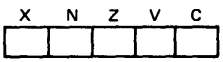
N – Negative
Z – Zero
V – Overflow
C – Carry
X – Extend

The first four bits are true condition code bits in that they reflect the condition of the result of a processor operation. The X-bit is an operand for multiprecision computations. The carry bit (C) and the multiprecision operand extend bit (X) are separate in the HD68000 to simplify the programming model.

• CONDITION CODE REGISTER NOTATION

In the instruction set details, the description of the effect on the condition codes is given in the following form:

Condition Codes:



Where

N (negative) set if the most significant bit of the result is set. Cleared otherwise.

Z (zero) set if the result equals zero. Cleared otherwise.

V (overflow) set if there was an arithmetic overflow. This implies that the result is not representable in the operand size. Cleared otherwise.

C (carry) set if a carry is generated out of the most significant bit of the operands for an addition. Also set if a borrow is generated in a subtraction. Cleared otherwise.

X (extend) transparent to data movement. When affected, it is set the same as the C-bit.

The notational convention that appears in the representation of the condition code registers is:

* set according to the result of the operation
– not affected by the operation
0 cleared
1 set
U undefined after the operation

• CONDITION CODE COMPUTATION

Most operations take a source operand and a destination operand, compute, and store the result in the destination location. Unary operations take a destination operand, compute, and store the result in the destination location. Table 22 details how each instruction sets the condition codes.

Table 22 Condition Code Computations

Operations	X	N	Z	V	C	Special Definition
ABCD	*	U	?	U	?	C = Decimal Carry Z = Z · $\overline{Rm}$ · ... · $\overline{R0}$
ADD, ADDI, ADDQ	*	*	*	?	?	V = Sm · Dm · $\overline{Rm}$ + $\overline{Sm}$ · $\overline{Dm}$ · Rm C = Sm · Dm + $\overline{Rm}$ · Dm + Sm · $\overline{Rm}$
ADDX	*	*	?	?	?	V = Sm · Dm · $\overline{Rm}$ + $\overline{Sm}$ · $\overline{Dm}$ · Rm C = Sm · Dm + $\overline{Rm}$ · Dm + Sm · $\overline{Rm}$ Z = Z · $\overline{Rm}$ · ... · $\overline{R0}$
AND, ANDI, EOR, EORI, MOVEQ, MOVE, OR, ORI, CLR, EXT, NOT, TAS, TST	–	*	*	0	0	
CHK	–	*	U	U	U	
SUB, SUBI SUBQ	*	*	*	?	?	V = $\overline{Sm}$ · Dm · $\overline{Rm}$ + Sm · $\overline{Dm}$ · Rm C = Sm · $\overline{Dm}$ + Rm · $\overline{Dm}$ + Sm · Rm
SUBX	*	*	?	?	?	V = $\overline{Sm}$ · Dm · $\overline{Rm}$ + $\overline{Sm}$ · $\overline{Dm}$ · Rm C = Sm · $\overline{Dm}$ + Rm · $\overline{Dm}$ + Sm · Rm Z = Z · $\overline{Rm}$ · ... · $\overline{R0}$
CMP, CMPI, CMPM	–	*	*	?	?	V = $\overline{Sm}$ · Dm · $\overline{Rm}$ + Sm · $\overline{Dm}$ · Rm C = Sm · $\overline{Dm}$ + Rm · $\overline{Dm}$ + Sm · Rm
DIVS, DIVU	–	*	*	?	0	V = Division Overflow
MULS, MULU	–	*	*	0	0	
SBCD, NBCD	*	U	?	U	?	C = Decimal Borrow Z = Z · $\overline{Rm}$ · ... · $\overline{R0}$
NEG NEGX	* *	* *	* ?	? ?	? ?	V = Dm · Rm, C = Dm + Rm V = Dm · Rm, C = Dm + Rm Z = Z · $\overline{Rm}$ · ... · $\overline{R0}$
BTST, BCHG, BSET, BCLR	–	–	?	–	–	Z = $\overline{Dn}$
ASL	*	*	*	?	?	V = Dm · ($\overline{D_{m-1}}$ + ... + $\overline{D_{m-r}}$) + $\overline{Dm}$ · (D_{m-1} + ... + D_{m-r}) C = D_{m-r+1}
ASL (r = 0)	–	*	*	0	0	
LSL, ROXL	*	*	*	0	?	C = D_{m-r+1}
LSR (r = 0)	–	*	*	0	0	
ROXL (r = 0)	–	*	*	0	?	C = X
ROL	–	*	*	0	?	C = D_{m-r+1}
ROL (r = 0)	–	*	*	0	0	
ASR, LSR, ROXR	*	*	*	0	?	C = D_{r-1}
ASR, LSR (r = 0)	–	*	*	0	0	
ROXR (r = 0)	–	*	*	0	?	C = X
ROR	–	*	*	0	?	C = D_{r-1}
ROR (r = 0)	–	*	*	0	0	

– Not affected
U Undefined
? Other— see Special Definition

* General Case:
X = C
N = Rm
Z = $\overline{Rm}$ · ... · $\overline{R0}$

Sm — Source operand most significant bit
Dm — Destination operand most significant bit
Rm — Result bit most significant bit
n — bit number
r — shift amount

• CONDITIONAL TESTS

Table 23 lists the condition names, encodings, and tests for the conditional branch and set instructions. The test associated with each condition is a logical formula based on the current state of the condition codes. If this formula evaluates to 1, the condition succeeds, or is true. If the formula evaluates to 0, the condition is unsuccessful, or false. For example, the T condition always succeeds, while the EQ condition succeeds only if the Z bit is currently set in the condition codes.

Table 23 Conditional Tests

Mnemonic	Condition	Encoding	Test
T	true	0000	1
F	false	0001	0
HI	high	0010	$\overline{C} \cdot \overline{Z}$
LS	low or same	0011	$C + Z$
CC	carry clear	0100	$\overline{C}$
CS	carry set	0101	C
NE	not equal	0110	$\overline{Z}$
EQ	equal	0111	Z
VC	overflow clear	1000	$\overline{V}$
VS	overflow set	1001	V
PL	plus	1010	$\overline{N}$
MI	minus	1011	N
GE	greater or equal	1100	$N \cdot V + \overline{N} \cdot \overline{V}$
LT	less than	1101	$N \cdot \overline{V} + \overline{N} \cdot V$
GT	greater than	1110	$N \cdot V \cdot \overline{Z} + \overline{N} \cdot \overline{V} \cdot \overline{Z}$
LE	less or equal	1111	$Z + N \cdot \overline{V} + \overline{N} \cdot V$

■ INSTRUCTION SET

The following paragraphs provide information about the addressing categories and instruction set of the HD68000.

• ADDRESSING CATEGORIES

Effective address modes may be categorized by the ways in which they may used. The following classifications will be used in the instruction definitions.

Data — If an effective address mode may be used to refer to data operands, it is considered a data addressing effective address mode.

Memory — If an effective address mode may be used to refer to memory operands, it is considered a memory addressing effective address mode.

Alterable — If an effective address mode may be used to refer to alterable (writeable) operands, it is considered an alterable addressing effective address mode.

Control — If an effective address mode may be used to refer to memory operands without an associated size, it is considered a control addressing effective address mode.

Table 24 shows the various categories to which each of the effective address modes belong. Table 25 is the instruction set summary.

The status register addressing mode is not permitted unless it is explicitly mentioned as a legal addressing mode.

These categories may be combined so that additional, more restrictive, classifications may be defined. For example, the instruction descriptions use such classifications as alterable memory or data alterable. The former refers to those addressing modes which are both alterable and memory addresses, and the latter refers to addressing modes which are both data and alterable.

• INSTRUCTION PRE-FETCH

The HD68000 uses a 2-word tightly-coupled instruction prefetch mechanism to enhance performance. This mechanism is described in terms of the microcode operations involved. If the execution of an instruction is defined to begin when the microroutine for that instruction is entered, some features of the prefetch mechanism can be described.

1) When execution of an instruction begins, the operation word and the word following have already been fetched. The operation word is in the instruction decoder.
2) In the case of multi-word instructions, as each additional word of the instruction is used internally, a fetch is made to the instruction stream to replace it.
3) The last fetch from the instruction stream is made when the operation word is discarded and decoding is started on the next instruction.
4) If the instruction is a single-word instruction causing a branch, the second word is not used. But because this word is fetched by the preceding instruction, it is impossible to avoid this superfluous fetch. In the case of an interrupt or trace exception, both words are not used.
5) The program counter usually points to the last word fetched from the instruction stream.

Table 24 Effective Addressing Mode Categories

Effective Address Modes	Mode	Register	Data	Addressing Categories		
				Memory	Control	Alterable
Dn	000	register number	X	–	–	X
An	001	register number	–	–	–	X
An@	010	register number	X	X	X	X
An@+	011	register number	X	X	–	X
An@-	100	register number	X	X	–	X
An@(d)	101	register number	X	X	X	X
An@(d, ix)	110	register number	X	X	X	X
xxx.W	111	000	X	X	X	X
xxx.L	111	001	X	X	X	X
PC@(d)	111	010	X	X	X	–
PC@(d, ix)	111	011	X	X	X	–
#xxx	111	100	X	X	–	–

Table 25 Instruction Set

Mnemonic Operation	Size	Addr. Mode	Dn		An		(An)		(An)+		-(An)		d(An)		d(An.Xi)		Abs.W		Abs.L		d(PC)		d(PC.Xi)		s=Immed d=SR/CC		Opcode Bit Pattern 1111 11 5432 1098 7654 3210	Boolean	Condition Codes XNZVC
			#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~			
ABCD	B	s:Dn d	2	6																							1100 RRR1 0000 0rrr	d10+s10+X →d	*U*U*
Add Digits		s:-(An) d									2	18															1100 RRR1 0000 1rrr		
ADD	B W	s:Dn d			ADDA		2	12	2	12	2	14	4	16	4	18	4	16	6	20							1101 DDD1 SSEE EEEE	d+Dn →d	*****
Add		d:Dn s	2	4	2*	4	2	8	2	8	2	10	4	12	4	14	4	12	6	16	4	12	4	14	4	8	1101 DDD0 SSee eeee	Dn+s →Dn	
Binary	L	s:Dn d			ADDA		2	20	2	20	2	22	4	24	4	26	4	24	6	28							1101 DDD0 10EE EEEE	d+Dn →d	
		d:Dn s	2	8	2	8	2	14	2	14	2	16	4	18	4	20	4	18	6	22	4	18	4	20	6	14	1101 DDD1 10ee eeee	Dn+s →Dn	
ADDA	W	d:An s	2	8	2	8	2	12	2	12	2	14	4	16	4	18	4	16	6	20	4	16	4	18	4	12	1101 AAA0 11ee eeee	An+s →An	- - - - -
Add Address	L	d:An s	2	8	2	8	2	14	2	14	2	16	4	18	4	20	4	18	6	22	4	18	4	20	6	14	1101 AAA1 11ee eeee		
ADDI	B W	s:Imm d	4	8	ADDA		4	16	4	16	4	18	6	20	6	22	6	20	8	24							0000 0110 SSEE EEEE	d+# →d	*****
Add Immed	L	s:Imm d	6	16	ADDA		6	28	6	28	6	30	8	32	8	34	8	32	10	36									
ADDQ	B W	s:Imm3 d	2	4	2*	4	2	12	2	12	2	14	4	16	4	18	4	16	6	20							0101 QQQ0 SSEE EEEE	d+# →d	*****
Add Quick	L	s:Imm3 d	2	8	2	8	2	20	2	20	2	22	4	24	4	26	4	24	6	28									
ADDX	B W	s:Dn d	2	4																							1101 RRR1 SS00 0rrr	d+s+X →d	*****
Add Multi-		s:-(An) d									2	18															1101 RRR1 SS00 1rrr		
precision	L	s:Dn d	2	8																							1101 RRR1 1000 0rrr		
		s:-(An) d									2	30															1101 RRR1 1000 1rrr		
AND	B W	s:Dn d					2	12	2	12	2	14	4	16	4	18	4	16	6	20							1100 DDD1 SSEE EEEE	d<and>Dn→d	-**00
Logical And		d:Dn s	2	4			2	8	2	8	2	10	4	12	4	14	4	12	6	16	4	12	4	14	4	8	1100 DDD0 SSee eeee	Dn<and>s→Dn	
	L	s:Dn d					2	20	2	20	2	22	4	24	4	26	4	24	6	28							1100 DDD1 10EE EEEE	d<and>Dn→d	
		d:Dn s	2	8			2	14	2	14	2	16	4	18	4	20	4	18	6	22	4	18	4	20	6	14	1100 DDD0 10ee eeee	Dn<and>s→Dn	
ANDI	B W	s:Imm d	4	8			4	16	4	16	4	18	6	20	6	22	6	20	8	24					4	20	0000 0010 SSEE EEEE	d<and>#→d	-**00
And Immed.	L	s:Imm d	6	16			6	28	6	28	6	30	8	32	8	24	8	32	10	36									
ASL, ASR	B W	count:Dn d	2	6+2n																							1110 rrrf SS10 0DDD	C ← 0	*****
Arithmetic		count:#1~8 d	2	6+2n																							1110 QQQf SS00 0DDD	X ← Left	
Shift	L	count:Dn d	2	8+2n																							1110 rrrf 1010 0DDD		
		count:#1~8 d	2	8+2n																							1110 QQQf 1000 0DDD	C → Right X	
Memory	W	count:1 d					2*	12	2*	12	2*	14	4*	16	4*	18	4*	16	6*	20							1110 000f 11EE EEEE		
BCHG	B	bit#:Dn d					2	12	2	12	2	14	4	16	4	18	4	16	6	20							0000 rrr1 01EE EEEE	~(bit)# of d→Z,	- - *- -
Test and		bit#:Imm d					4	16	4	16	4	18	6	20	6	22	6	20	8	24							0000 1000 01EE EEEE	~(bit)# of d→	
Change	L	bit#:Dn d	2	<8																							0000 rrr1 01EE EEEE	(bit)# of d	
		bit#:Imm d	4	<12																							0000 1000 01EE EEEE		
BCLR	B	bit#:Dn d					2	12	2	12	2	14	4	16	4	18	4	17	6	20							0000 rrr1 10EE EEEE		
Test and		bit#:Imm d					4	16	4	16	4	18	6	20	6	22	6	20	8	24							0000 1000 10EE EEEE	~(bit)# of d→Z,	- - *- -
Clear	L	bit#:Dn d	2	<10																							0000 rrr1 10EE EEEE	0→(bit)# of d	
		bit#:Imm d	4	<14																							0000 1000 10EE EEEE		
BSET	B	bit#:Dn d					2	12	2	12	2	14	4	16	4	18	4	16	6	20							0000 rrr1 11EE EEEE	~(bit)#of d→Z,	- - *- -
Test and		bit#:Imm d					4	16	4	16	4	18	6	20	6	22	6	20	8	24							0000 1000 11EE EEEE	1→(bit)# of d	
Set	L	bit#:Dn d	2	<8																							0000 rrr1 11EE EEEE		
		bit#:Imm d	4	<12																							0000 1000 11EE EEEE		
BTST	B	bit#:Dn d					2	8	2	8	2	10	4	12	4	14	4	12	6	16	4	12	4	14			0000 rrr1 00EE EEEE	~(bit)# of d→Z	- - *- -
Bit Test		bit#:Imm d					4	12	4	12	4	14	6	16	6	18	6	16	8	20	6	16	6	18			0000 1000 00EE EEEE		
	L	bit#:Dn d	2	6																							0000 rrr1 00EE EEEE		
		bit#:Imm d	4	10																							0000 1000 00EE EEEE		
CHK	W			<40	←trap→			<44		<44		<46		<48		<50		<48		<52		<48		<50		<44	0100 DDD1 10ee eeee	If Dn<0, or	-*UUU
Check Reg-		d:Dn s	2				2		2		2		4		4		4		6		4		4		4			Dn>(bound),	
ister Against		(bound)		8	←no→			12		12		14		16		18		16		20		16		18		12		then trap	
Bounds					trap																								
CLR	B W	d:	2	4			2	12	2	12	2	14	4	16	4	18	4	16	6	20							0100 0010 SSEE EEEE	d→MPU	-0100
Clear Operand	L	d:	2	6			2	20	2	20	2	22	4	24	4	26	4	24	6	28								0→d	
CMP	B W	d:Dn s:	2	4	2*	4	2	8	2	8	2	10	4	12	4	14	4	12	6	16	4	12	4	14	4	8	1011 DDD0 SSee eeee	Dn-s	-****
Compare	L	d:Dn s:	2	6	2	6	2	14	2	14	2	16	4	18	4	20	4	18	6	22	4	18	4	20	6	14			
Binary																													
CMPA	W	d:An s:	2	6	2	6	2	10	2	10	2	12	4	14	4	16	4	12	6	18	4	14	4	16	4	10	1011 AAA0 11ee eeee	An-s	-****
Compare	L	d:An s:	2	6	2	6	2	14	2	14	2	16	4	18	4	20	4	18	6	22	4	18	4	20	6	14	1011 AAA1 11ee eeee		
Address																													
CMPI	B W	s:Imm d:	4	8	CMPA		4	12	4	12	4	14	6	16	6	18	6	16	8	20							0000 1100 SSEE EEEE	d-#	-****
Compare Imm.	L	s:Imm d:	6	14	CMPA		6	20	6	20	6	22	8	24	8	26	8	24	10	28									
CMPM	B W	s:(An)+ d:							2	12																	1011 RRR1 SS00 1rrr	d-s	-****
Compare	L	s:(An)+ d:							2	20																			
Memory																													
DIVS	W	d:Dn s:	2	<158			2	<162	2	<162	2	<164	4	<166	4	<168	4	<166	6	<170	4	<166	4	<168	4	<162	1000 DDD1 11ee eeee	Dn32/s16→	-***0
Divide Signed																												Dn(rq)	
DIVU	W	d:Dn s:	2	<140			2	<144	2	<144	2	<146	4	<148	4	<150	4	<148	6	<152	4	<148	4	<150	4	<144	1000 DDD0 11ee eeee	Dn32/s16→	-***0
Divide																												Dn(rq)	
Unsigned																													
EOR	B W	s:Dn d:	2	4			2	12	2	12	2	14	4	16	4	18	4	16	6	20							1011 rrr1 SSEE EEEE	d⊕Dn→d	-**00
Exclusive OR	L	s:Dn d:	2	8			2	20	2	20	2	22	4	24	4	26	4	24	6	28									
Logical																													
EORI	B W	s:Imm d	4	8			4	16	4	16	4	18	6	20	6	22	6	20	8	24					4	20	0000 1010 SSEE EEEE	d⊕#→d	-**00
Exclusive OR	L	s:Imm d:	6	16			6	28	6	28	6	30	8	32	8	34	8	32	10	36									
Immediate																													
EXG	L	s:Dn d:	2	6																							1100 DDD1 0100 0DDD	s↔d	- - - - -
Exchange		s:An d:	2	6	2	6																					1100 AAA1 0100 1AAA		
Registers																											1100 DDD1 1000 1AAA		
EXT	W	d:	2	4																							0100 1000 1000 0DDD	bit 7→bit 8~15	
Sign Extend	L	d:	2	4																							0100 1000 1100 0DDD	bit 15→bit 16~31	-**00
LEA	L	d:An s:					2	4					4	8	4	12	4	8	6	12	4	8	4	12			0100 AAA1 11ee eeee	s→An	- - - - -
Load Effect-																													
ive Address																													
LINK		disp:Imm s:			4	16																					0100 1110 0101 0AAA	An→-(SP)	- - - - -
Link and																												SP→An	
Allocate																												SP+disp→SP	

Note : Refer to "Condition Code Computations" as for condition Code.
* Word only
<: Maximum value
#: Number of Program Bytes
~: Number of Clock Periods

Opcode Bit Pattern Key

A: Address Register #
C: Test Condition
D: Data Register #
e: Source Effective Address
E: Destination Effective Address
f: Direction: 0—Right, 1—Left
M: Destination EA Mode
P: Displacement
Q: Quick Immediate Data
r: Source Register
R: Destination Register
S: Size: 00—Byte, 01—Word, 10—Long Word, 11—Another Operation (In the MOVE Instruction: 01—Byte, 10—Long Word, 11—Word)
V: Vector #

(to be continued)

Mnemonic / Operation	Size	Addr. Mode	Dn		An		(An)		(An)+		-(An)		d(An)		d(An,Xi)		Abs.W		Abs.L		d(PC)		d(PC,Xi)		s=Immed d=SR/CC		Opcode Bit Pattern 1111 11 / 5432 1098 / 7654 3210	Boolean	Condition Codes XNZVC
			#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~			
LSL, LSR	B W	count Dn	d	2	6+2n																					1110 rrrf SS10 1DDD	C ← 0	***0*
Logical Shift		count #1~8	d	2	6+2n																					1110 QQQf SS00 1DDD	X ← Left	
	L	count:Dn	d	2	8+2n																					1110 rrrf 1010 1DDD	0 → C	
		count #1~8	d	2	8+2n																					1110 QQQf 1000 1DDD	X Right	
Memory	W	count 1	d				2*	12	2*	12	2*	14	4*	16	4*	18	4*	16	6*	20							1110 001f 11EE EEEE		
MOVE	B W	s:Dn	d	2	4	MOVEA	2	8	2	8	2	8	4	12	4	14	4	12	6	16							00SS RRRM MMee eeee	s →d	- **0 0
Move Data		s:An	s	2	4	MOVEA	2	8	2	8	2	8	4	12	4	14	4	12	6	16									
		s:(An)	d	2	8	MOVEA	2	12	2	12	2	12	4	16	4	18	4	16	6	20									
		s:(An)+	d	2	8	MOVEA	2	12	2	12	2	12	4	16	4	18	4	16	6	20									
		s -(An)	d	2	10	MOVEA	2	14	2	14	2	14	4	18	4	20	4	18	6	22									
		s:d(An)	d	4	12	MOVEA	4	16	4	16	4	16	6	20	6	22	6	20	8	24									
		s:d(An,X)	d	4	14	MOVEA	4	18	4	18	4	18	6	22	6	24	6	22	8	26									
		s:Abs.W	d	4	12	MOVEA	4	16	4	16	4	16	6	20	6	22	6	20	8	24									
		s:Abs.L	d	6	16	MOVEA	6	20	4	20	6	20	8	24	8	26	8	24	10	28									
		s:d(PC)	d	4	12	MOVEA	4	16	4	16	4	16	6	20	6	22	6	20	8	24									
		s:d(PC,X)	d	4	14	MOVEA	4	18	4	18	4	18	6	22	6	24	6	22	8	26									
		s:Imm	d	4	8	MOVEA	4	12	4	12	4	12	6	16	6	18	6	16	8	20									
	L	s:Dn	d	2	4	MOVEA	2	12	2	12	2	12	4	16	4	18	4	16	6	20									
		s:An	d	2	4	MOVEA	2	12	2	12	2	12	4	16	4	18	4	16	6	20									
		s:(An)	d	2	12	MOVEA	2	20	2	20	2	20	4	24	4	26	4	24	6	28									
		s:(An)+	d	2	12	MOVEA	2	20	2	20	2	20	4	24	4	26	4	24	6	28									
		s -(An)	d	2	14	MOVEA	2	22	2	22	2	22	4	26	4	28	4	26	6	30									
		s:d(An)	d	4	16	MOVEA	4	24	4	24	4	24	6	28	6	30	6	28	8	32									
		s:d(An,X)	d	4	18	MOVEA	4	26	4	26	4	26	6	30	6	32	6	30	8	34									
		s:Abs.W	d	4	16	MOVEA	4	24	4	24	4	24	6	28	6	30	6	28	8	32									
		s:Abs.L	d	6	20	MOVEA	6	28	4	28	6	28	8	32	8	34	8	32	10	36									
		s:d(PC)	d	4	16	MOVEA	4	24	4	24	4	24	6	28	6	30	6	28	8	32									
		s:d(PC,X)	d	4	18	MOVEA	4	26	4	26	4	26	6	30	6	32	6	30	8	34									
		s:Imm	d	6	12	MOVEA	6	20	6	20	6	20	8	24	8	26	8	24	10	28									
MOVE	W	d:CCR	s	2	12			2	16	2	16	2	18	4	20	4	22	4	20	6	24	4	20	4	22	4	16	0100 0100 11ee eeee	s→CCR	*****
Move to Condition Codes																													
MOVE	W	d:SR	s	2	12			2	16	2	16	2	18	4	20	4	22	4	20	6	24	4	20	4	22	4	16	0100 0110 11ee eeee	s→SR	*****
Move to from Status Reg		s:SR	d	2	6			2	12	2	12	2	14	4	16	4	18	4	16	6	20							0100 0000 11EE EEEE	d→MPU	- - - - -
																													SR→d
MOVE	L	s:USP	d			2	4																					0100 1110 0110 1AAA	USP→An	- - - - -
Move to from User SP(A7)		d:USP	s			2	4																					0100 1110 0110 0AAA	An→USP	
MOVEA	W	d:An	s	2	4	2	4	2	8	2	8	2	10	4	12	4	14	4	12	6	16	4	12	4	14	4	8	0011 AAA0 01ee eeee	s *An	- - - - -
Move Address	L	d:An	s	2	4	2	4	2	12	2	12	2	14	4	16	4	18	4	16	6	20	4	16	4	18	6	12	0010 AAA0 01ee eeee		
MOVEM	W	s:Xn	d					4	8·4n			4	8·4n	6	12·4n	6	14·4n	6	12·4n	8	16·4n							0100 1000 10EE EEEE	Xn *d	- - - - -
Move Multiple Registers																													a7~a0 d7~d0↑
		d:Xn	s					4	12·4n	4	12·4n			6	16·4n	6	18·4n	6	16·4n	8	20·4n	6	16·4n	6	18·4n			0100 1100 10ee eeee	s *Xn **
																													a7~a0 d7~d0
	L	s:Xn	d					4	8·8n			4	8·8n	6	12·8n	6	14·8n	6	12·8n	8	16·8n							0100 1000 11EE EEEE	Xn *d	
																													a7~a0 d7~d0↑
		d:Xn	s					4	12·8n	4	12·8n			6	16·8n	6	18·8n	6	16·8n	8	20·8n	6	16·8n	6	18·8n			0100 1100 11ee eeee	s *Xn **
																													a7~a0 d7~d0
MOVEP	W	s:Dn	d											4	16													0000 DDD1 1000 1AAA	Dn→d by bytes	- - - - -
Move		s:d(An)	d	4	16																							0000 DDD1 0000 1AAA	s→Dn by bytes	
Periphperal	L	s:Dn	d											4	24													0000 DDD1 1100 1AAA	Dn→d by bytes	
		s:d(An)	d	4	24																							0000 DDD1 0100 1AAA	s *Dn by bytes	
MOVEQ	L	s:Imm8	d	2	4																							0111 DDD0 QQQQ QQQQ	#→Dn	- **0 0
Move Quick																													
MULS	W	s:Dn	d	2	<70			2	<74	2	<74	2	<76	4	<78	4	<80	4	<78	6	<82	4	<78	4	<80	4	<74	1100 DDD1 11ee eeee	Dn×s →Dn	- **0 0
Multiply Signed																													
MULU	W	s:Dn	d	2	<70			2	<74	2	<74	2	<76	4	<78	4	<80	4	<78	6	<82	4	<78	4	<80	4	<74	1100 DDD0 11ee eeee	Dn×s →Dn	- **0 0
Multiply Unsigned																													
NBCD	B		d	2	6			2	12	2	12	2	14	4	16	4	18	4	16	6	20							0100 1000 00EE EEEE	0-d10-X→d	*U*U*
Negate Digit																													
NEG	B W		d	2	4			2	12	2	12	2	14	4	16	4	18	4	16	6	20							0100 0100 SSEE EEEE	0-d →d	*****
Negate Binary	L		d	2	6			2	20	2	20	2	22	4	24	4	26	4	24	6	28									
NEGX	B W		d	2	4			2	12	2	12	2	14	4	16	4	18	4	16	6	20							0100 0000 SSEE EEEE	0-d-X →d	*****
Negate Multi-precision	L		d	2	6			2	20	2	20	2	22	4	24	4	26	4	24	6	28									
NOT	B W		d	2	4			2	12	2	12	2	14	4	16	4	18	4	16	6	20							0100 0110 SSEE EEEE	~d →d	- **0 0
Logical Complement	L		d	2	6			2	20	2	20	2	22	4	24	4	26	4	24	6	28									
OR	B W	s:Dn	d					2	12	2	12	2	14	4	16	4	18	4	16	6	20							1000 DDD1 SSEE EEEE	d<or>Dn →d	- **0 0
Inclusive OR		d:Dn	s	2	4			2	8	2	8	2	10	4	12	4	14	4	12	6	16	4	12	4	14	4	8	1000 DDD0 SSee eeee	Dn<or>s→Dn	
Logical	L	s:Dn	d					2	20	2	20	2	22	4	24	4	26	4	24	6	28							1000 DDD1 10EE EEEE	d<or>Dn→d	
		d:Dn	s	2	8			2	14	2	14	2	16	4	18	4	20	4	18	6	22	4	18	4	20	6	14	1000 DDD0 10ee eeee	Dn<or>s→Dn	
ORI	B W	s:Imm	d	4	8			4	16	4	16	4	18	6	20	6	22	6	20	8	24					4	20	0000 0000 SSEE EEEE	d<or>#→d	- **0 0
OR Immediate	L	s:Imm	d	6	16			6	30	6	30	6	32	8	34	8	36	8	34	10	38									
PEA	L		s					2	14					4	18	4	22	4	18	6	22	4	18	4	22			0100 1000 01ee eeee	s→-(SP)	- - - - -
Push Effective Address																													
ROR, ROL	B W	count:Dn	d	2	6+2n																							1110 rrrf SS11 1DDD	Right n C	- **0 0
Rotate		count:#1~8	d	2	6+2n																								
without X	L	count:Dn	d	2	8+2n																							1110 QQQf SS01 1DDD		
		count:#1~8	d	2	8+2n																							1110 rrrf 1011 1DDD	C Left n	
Memory	W	count:1	d					2*	12	2*	12	2*	14	4*	16	4*	18	4*	16	6*	20							1110 QQQf 1001 1DDD		
																												1110 011f 11EE EEEE		

Note : Refer to "Condition Code Computations" as for condition Code.
* : Word only
< : Maximum value
: Number of Program Bytes
~ : Number of Clock Periods

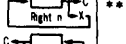
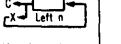
** : The MPU goes through an extra null read cycle after a multiple read is done (The last EA+2).

Opcode Bit Pattern Key

A: Address Register #
C: Test Condition
D: Data Register #
e: Source Effective Address
E: Destination Effective Address
f: Direction; 0-Right, 1-Left
M: Destination EA Mode
P: Displacement
Q: Quick Immediate Data
r: Source Register
R: Destination Register
S: Size; 00-Byte, 01-Word, 10-Long Word, 11-Another Operation
V: Vector #

(In the MOVE Instruction: 01-Byte, 10-Long Word, 11-Word)

(to be continued)

Mnemonic Operation	Size	Addr. Mode	Dn		An		(An)		(An)+		−(An)		d(An)		d(An,Xi)		Abs.W		Abs.L		d(PC)		d(PC,Xi)		s=Immed d=SR/CC		Opcode Bit Pattern 1111 11 5432 1098 7654 3210	Boolean	Condition Codes XNZVC
			#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~			
ROXR,ROXL	B W	count:Dn d:	2	6+2n																							1110 rrrf SS11 0DDD		***0*
Rotate		count: #]~8 d:	2	6+2n																							1110 QQQf SS01 0DDD		
through X	L	count:Dn d:	2	8+2n																							1110 rrrf 1011 0DDD		
		count: #]~8 d:	2	8+2n																							1110 QQQf 1001 0DDD		
Memory	W	count:1 d:					2*	12	2*	12	2*	14	4*	16	4*	18	4*	16	6*	20							1110 010f 11EE EEEE		
SBCD	B	s:Dn d:	2	6																							1000 RRR1 0000 0rrr		
Subtract		s (An) d:									2	18															1000 RRR1 0000 1rrr	d10 s10 X →d	*U*U*
digits																													
Scc	B	cc d:	2	6 4			2	12	2	12	2	14	4	16	4	18	4	16	6	20							0101 CCCC 11EE EEEE	If cc true,1's →d	-----
Set																												Else, 0's →d	
Conditionally																													
SUB	B W	s:Dn d:			SUBA		2	12	2	12	2	14	4	16	4	18	4	16	6	20							1001 DDD1 SSEE EEEE	d Dn →d	*****
Subtract		d:Dn s:	2	4	2*	4	2	8	2	8	2	10	4	12	4	14	4	12	6	16	4	12	4	14	4	8	1001 DDD0 SSee eeee	Dn s→Dn	
Binary	L	s:Dn d			SUBA		2	20	2	20	2	22	4	24	4	26	4	24	6	28							1001 DDD1 10EE EEEE	d Dn→d	
		d:Dn s:	2	8	2	8	2	14	2	14	2	16	4	18	4	20	4	18	6	22	4	18	4	20	6	14	1001 DDD0 10ee eeee	Dn s→Dn	
SUBA	W	d:An s:	2	8	2	8	2	12	2	12	2	14	4	16	4	18	4	16	6	20	4	16	4	18	4	12	1001 AAA0 11ee eeee	An s →An	-----
Subtract	L	d:An s	2	8	2	8	2	14	2	14	2	16	4	18	4	20	4	18	6	22	4	18	4	20	6	14	1001 AAA1 11ee eeee		
Address																													
SUBI	B W	s:Imm d:	4	8		SUBA	4	16	4	16	4	18	6	20	6	22	6	20	8	24							0000 0100 SSEE EEEE	d = →d	*****
Subtract	L	s:Imm d	6	16		SUBA	6	28	6	28	6	30	8	32	8	34	8	32	10	36									
Immediate																													
SUBQ	B W	s:Imm3 d	2	4	2*	4	2	12	2	12	2	14	4	16	4	18	4	16	6	20							0101 QQQ1 SSEE EEEE	d = →d	*****
Subtract	L	s:Imm3 d	2	8	2	8	2	16	2	16	2	22	4	24	4	26	4	24	6	28									
Quick																													
SUBX	B W	s:Dn d	2	4																							1001 RRR1 SS00 0rrr	d s-X→d	*****
Subtract		s (An) d									2	18															1001 RRR1 SS00 1rrr		
Multiprecision	L	s:Dn d	2	8																							1001 RRR1 1000 0rrr		
		s:(An) d									2	30															1001 RRR1 1000 1rrr		
SWAP	W	d	2	4																							0100 1000 0100 0DDD	Dn(31:16)←→ Dn(15:0)	-**00
Swap Register Halves																													
TAS	B	d	2	4																							0100 1010 11EE EEEE	test d →cc 1→bit 7 of d	-**00
Test and Set Operand																													
TST	B W	d	2	4																							0100 1010 SSEE EEEE	test d →cc	-**00
Test	L	d:	2	4																									
UNLK																											0100 1110 0101 1AAA	An→SP; (SP)+ →An	-----
Unlink					2	12																							

Bcc	B	disp																			bra taken	2	10	0110 CCCC PPPP PPPP	if cc true, PC+disp →PC	-----
Branch	W	disp																			bra not taken	2	8			
Conditionally																					bra taken	4	10			
																					bra not taken	4	14			
BRA	B	disp																				2	10	0110 0000 PPPP PPPP	PC+disp →PC	-----
Branch	W	disp																				4	10			
Always																										
BSR	B	disp																				2	20	0110 0001 PPPP PPPP	PC→ (SP). PC+disp →PC	-----
Branch to Subroutine	W	disp																				4	20			
DBcc	W	disp:Imm				cc	Counter		Branch															0101 CCCC 1100 1DDD	If cc false, Dn-1 →Dn & if Dn≠-1,PC+disp →PC Else, NOP	-----
Decrement				10		false	≠ 1		yes																	
Counter, &		counter:	4	12		true	≠ 1		no																	
Branch Until				14		false	expired		no																	
Condition True or Count=1																										
JMP		d:					2	8					4	10	4	14	4	10	6	12	4	10	4	14			0100 1110 11EE EEEE	d→PC	-----
Jump to																										
JSR		d:					2	16					4	18	4	22	4	18	6	20	4	18	4	22			0100 1110 10EE EEEE	PC→ (SP).d →PC	-----
Jump to Subroutine																										
NOP			2	4																							0100 1110 0111 0001	none	-----
No Operation																										
RESET			2	132																							0100 1110 0111 0000	assert RESET pin	-----
Reset External Devices																										
RTE			2	20																							0100 1110 0111 0011	(SP)+ →SR, (SP)+ →PC	*****
Return from Exception																										
RTR			2	20																							0100 1110 0111 0111	(SP)+ →CC, (SP)+ →PC	*****
Return from Subroutine Restore CC																										
RTS			2	16																							0100 1110 0111 0101	(SP)+ →PC	-----
Return from Subroutine																										
STOP																						4	4	0100 1110 0111 0010	#→SR, Wait for Interrupt	*****
Load SR Stop																										
TRAP			2	34																							0100 1110 0100 VVVV	PC→ (SSP). SR→ (SSP). (Vector) →PC	-----
Trap																										
TRAPV			2	34	Trap taken																			0100 1110 0111 0110	If V=1 then PC+ -(SSP).SR+ (SSP) (TRAPV vector) →PC Else, NOP	-----
Trap if Overflow Set				4	Trap not taken																					

Note : Refer to "Condition Code Computations" as for condition Code.
*: Word only
<: Maximum value
#: Number of Program Bytes
~: Number of clock Periods

A: Address Register #
C: Test Condition
D: Data Register #
e: Source Effective Address
E: Destination Effective Address

Opcode Bit Pattern Key

f: Direction; 0—Right, 1—Left
M: Destination EA Mode
P: Displacement
Q: Quick Immediate Data
r: Source Register

R: Destination Register
S: Size; 00—Byte
01—Word
10—Long Word
11—Another Operation
V: Vector #

(In the MOVE Instruction)
01—Byte
10—Long Word
11—Word

■ INSTRUCTION FORMAT SUMMARY

This provides a summary of the first word in each instruction of the instruction set. Table 26 is an operation code (op-code) map which illustrates how bits 15 through 12 are used to specify the operations. The remaining paragraph groups the instructions according to the op-code map.

where, Size; Byte = 00 Sz; Word = 0
Word = 01 Long Word = 1
Long Word = 10

Table 26 Operation Code Map

Bits 15 thru 12	Operation
0000	Bit Manipulation/MOVEP/Immediate
0001	Move Byte
0010	Move Long
0011	Move Word
0100	Miscellaneous
0101	ADDQ/SUBQ/S_{CC}/DB_{CC}
0110	B_{CC}
0111	MOVEQ
1000	OR/DIV/SBCD
1001	SUB/SUBX
1010	(Unassigned)
1011	CMP/EOR
1100	AND/MUL/ABCD/EXG
1101	ADD/ADDX
1110	Shift/Rotate
1111	(Unassigned)

(1) BIT MANIPULATION, MOVE PERIPHERAL, IMMEDIATE INSTRUCTIONS

Dynamic Bit

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	Register			1	Type		Effective Address					

Static Bit

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	0	0	0	Type		Effective Address					

Bit Type Codes: TST = 00, CHG = 01, CLR = 10, SET = 11

MOVEP

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	Register			Op-Mode			0	0	1	Register		

Op-Mode; Word to Reg = 100, Long to Reg = 101, Word to Mem = 110, Long to Mem = 111

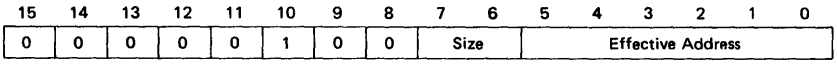
OR Immediate

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	Size		Effective Address					

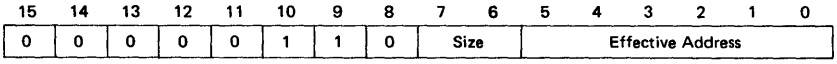
AND Immediate

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	0	Size		Effective Address					

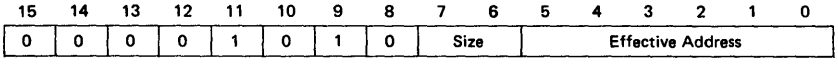
SUB Immediate



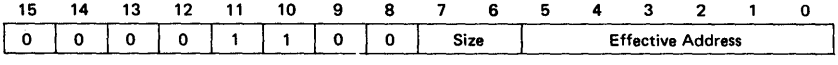
ADD Immediate



EOR Immediate

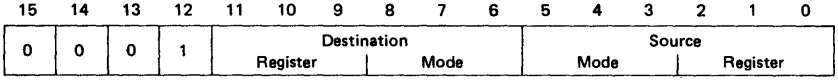


CMP Immediate



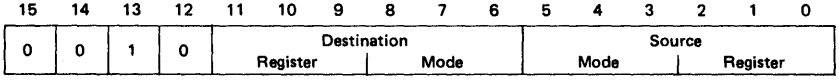
(2) MOVE BYTE INSTRUCTION

MOVE Byte



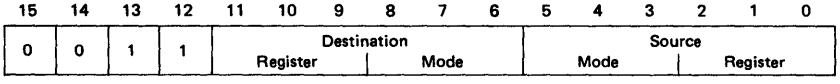
(3) MOVE LONG INSTRUCTION

MOVE Long



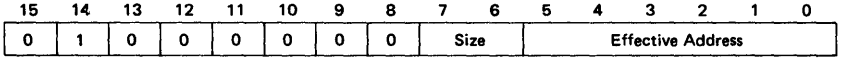
(4) MOVE WORD INSTRUCTION

MOVE Word

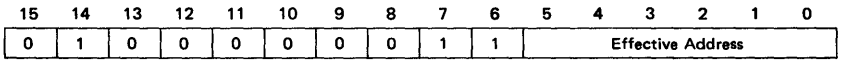


(5) MISCELLANEOUS INSTRUCTIONS

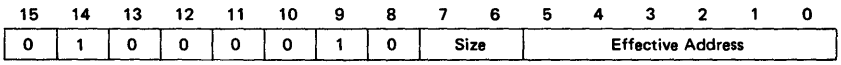
NEGX



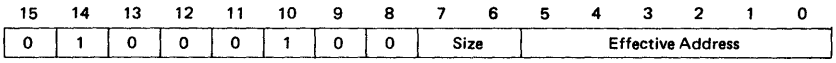
MOVE from SR



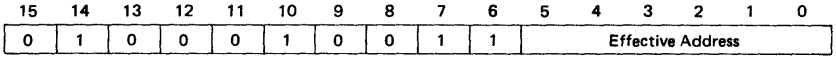
CLR



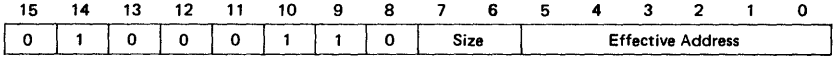
NEG



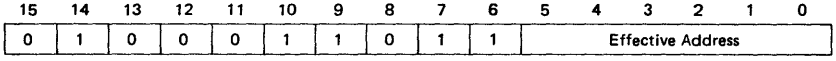
MOVE to CCR



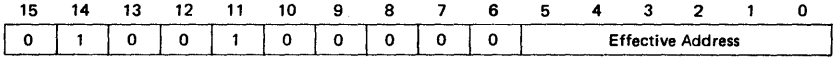
NOT



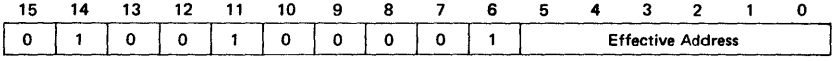
MOVE to SR



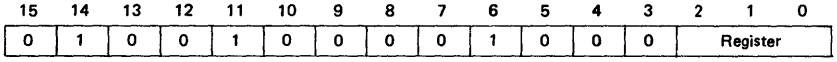
NBCD



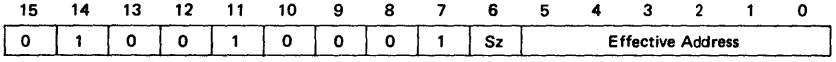
PEA



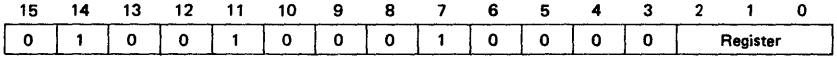
SWAP



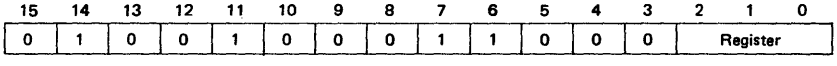
MOVEM Registers to EA



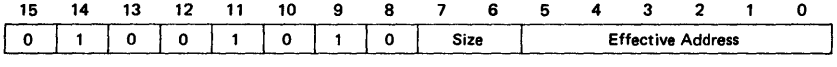
EXTW



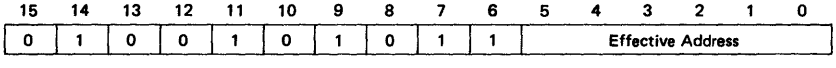
EXTL



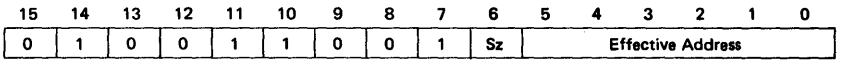
TST



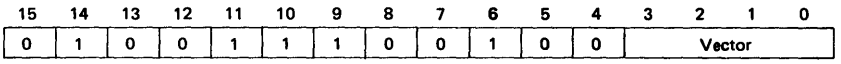
TAS



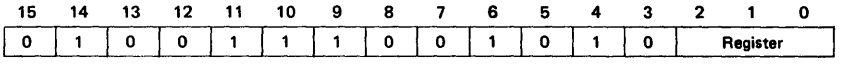
MOVEM EA to Registers



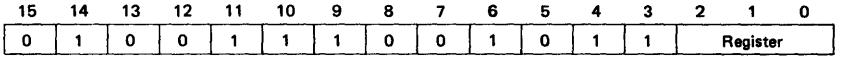
TRAP



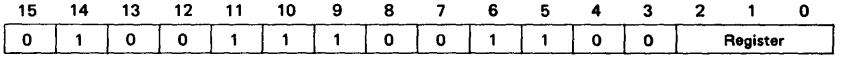
LINK



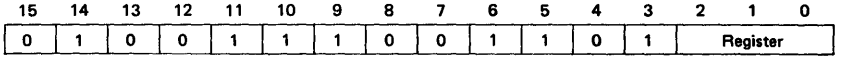
UNLK



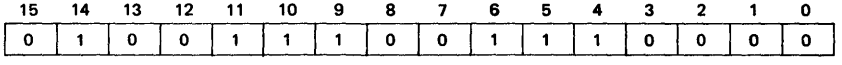
MOVE to USP



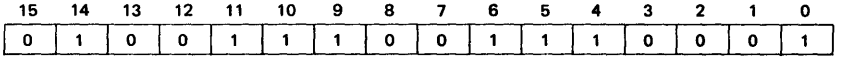
MOVE from USP



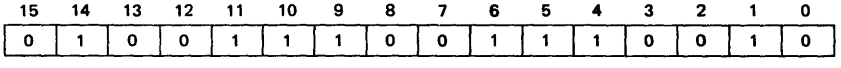
RESET



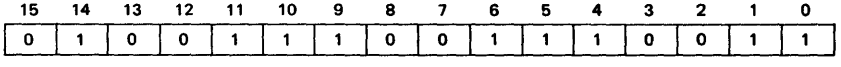
NOP



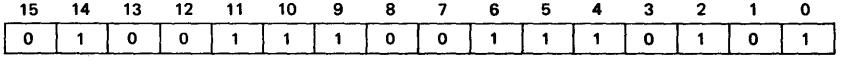
STOP



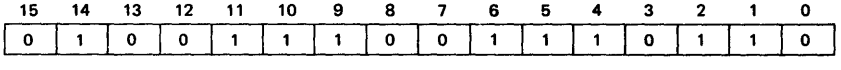
RTE



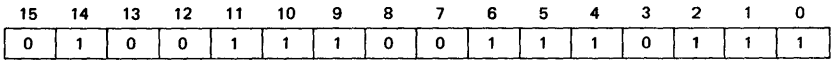
RTS



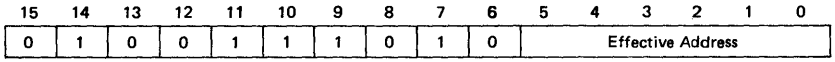
TRAPV



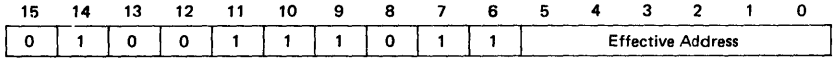
RTR



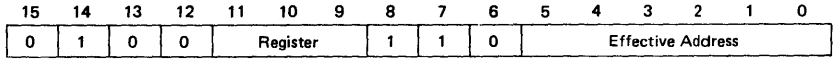
JSR



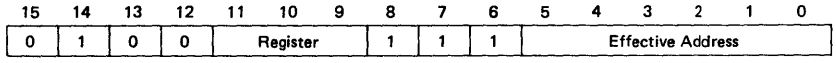
JMP



CHK

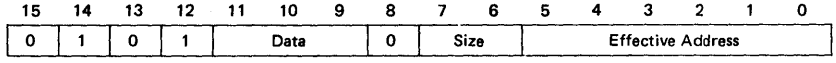


LEA

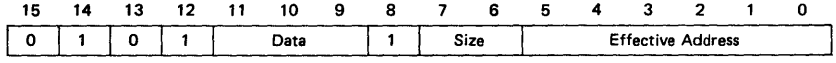


(6) ADD QUICK, SUBTRACT QUICK, SET CONDITIONALLY, DECREMENT INSTRUCTIONS

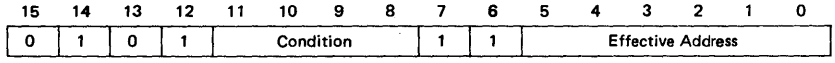
ADDQ



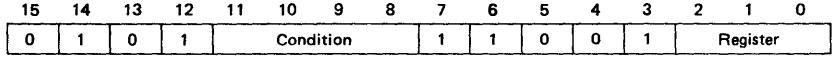
SUBQ



S_{CC}

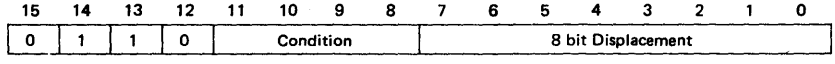


DB_{CC}

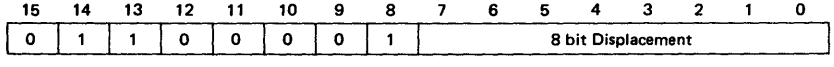


(7) BRANCH CONDITIONALLY, BRANCH TO SUBROUTINE INSTRUCTION

B_{CC}

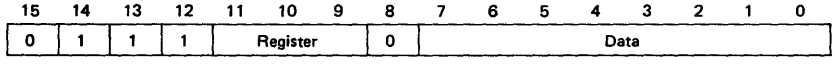


BSR



(8) MOVE QUICK INSTRUCTION

MOVEQ



(9) OR, DIVIDE, SUBTRACT DECIMAL INSTRUCTIONS

OR

15	14	13	12	11 10 9	8 7 6	5 4 3 2 1 0
1	0	0	0	Register	Op-Mode	Effective Address

Op-Mode B	W	L	
000	001	010	Dn ∨ EA → Dn
100	101	110	EA ∨ Dn → EA

DIVU

15	14	13	12	11 10 9	8	7	6	5 4 3 2 1 0
1	0	0	0	Register	0	1	1	Effective Address

DIVS

15	14	13	12	11 10 9	8	7	6	5 4 3 2 1 0
1	0	0	0	Register	1	1	1	Effective Address

SBCD

15	14	13	12	11 10 9	8	7	6	5	4	3	2 1 0
1	0	0	0	Destination Register	1	0	0	0	0	R/M	Source Register

R/M (register/memory): register – register = 0, memory – memory = 1

(10) SUBTRACT, SUBTRACT EXTENDED INSTRUCTIONS

SUB

15	14	13	12	11 10 9	8 7 6	5 4 3 2 1 0
1	0	0	1	Register	Op-Mode	Effective Address

Op-Mode B	W	L	
000	001	010	Dn–EA → Dn
100	101	110	EA–Dn → EA
–	011	111	An–EA → An

SUBX

15	14	13	12	11 10 9	8	7 6	5	4	3	2 1 0
1	0	0	1	Destination Register	1	Size	0	0	R/M	Source Register

R/M (register/memory): register – register = 0, memory – memory = 1

(11) COMPARE, EXCLUSIVE OR INSTRUCTIONS

CMP

15	14	13	12	11 10 9	8 7 6	5 4 3 2 1 0
1	0	1	1	Register	Op-Mode	Effective Address

Op-Mode B	W	L	
000	001	010	Dn–EA
–	011	111	An–EA

CMPM

15	14	13	12	11 10 9	8	7 6	5	4	3	2 1 0
1	0	1	1	Register	1	Size	0	0	1	Register

EOR

15	14	13	12	11 10 9	8	7 6	5 4 3 2 1 0
1	0	1	1	Register	1	Size	Effective Address

(12) AND, MULTIPLY, ADD DECIMAL, EXCHANGE INSTRUCTIONS

AND

15	14	13	12	11 10 9	8 7 6	5 4 3 2 1 0
1	1	0	0	Register	Op-Mode	Effective Address

Op-Mode B	W	L	
000	001	010	Dn ∧ EA → Dn
100	101	110	EA ∧ Dn → EA

MULU

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	Register			0	1	1	Effective Address					

MULS

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	Register			1	1	1	Effective Address					

ABCD

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	Destination Register			1	0	0	0	0	R/M	Source Register		

R/M (register/memory): register — register = 0, memory — memory = 1

EXGD

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	Data Register			1	0	1	0	0	0	Data Register		

EXGA

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	Address Register			1	0	1	0	0	1	Address Register		

EXGM

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	Data Register			1	1	0	0	0	1	Address Register		

(13) ADD, ADD EXTENDED INSTRUCTIONS

ADD

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	1	Register			Op-Mode			Effective Address					

	Op-Mode		
B	W	L	
000	001	010	Dn + EA → Dn
100	101	110	EA + Dn → EA
—	011	111	An + EA → An

ADDX

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	1	Destination Register			1	Size		0	0	R/M	Source Register		

R/M (register/memory): register — register = 0, memory — memory = 1

(14) SHIFT/ROTATE INSTRUCTIONS

Data Register Shifts

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	0	Count/Register			d	Size		i/r	Type		Register		

Memory Shifts

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	0	0	Type		d	1	1	Effective Address					

Shift Type Codes: AS = 00, LS = 01, ROX = 10, RO = 11
d (direction): Right = 0, Left = 1
i/r (count source): Immediate Count = 0, Register Count = 1

■ INSTRUCTION EXECUTION TIMES

The following paragraphs contain listings of the instruction execution times in terms of external clock (CLK) periods. In this timing data, it is assumed that both memory read and write cycle times are four clock periods. Any wait states caused by a longer memory cycle must be added to the total instruction time. The number of bus read and write cycles for each instruction is also included with the timing data. This data is enclosed in parenthesis following the execution periods and is shown as: (r/w) where r is the number of read cycles and w is the number of write cycles.

(NOTE) The number of periods includes instruction fetch and all applicable operand fetches and stores.

• EFFECTIVE ADDRESS OPERAND CALCULATION TIMING

Table 27 lists the number of clock periods required to compute an instruction's effective address. It includes fetching of any extension words, the address computation, and fetching of the memory operand. The number of bus read and write cycles is shown in parenthesis as (r/w). Note there are no write cycles involved in processing the effective address.

• MOVE INSTRUCTION CLOCK PERIODS

Table 28 and 29 indicate the number of clock periods for the move instruction. This data includes instruction fetch, operand reads, and operand writes. The number of bus read and write cycles is shown in parenthesis as: (r/w).

• STANDARD INSTRUCTION CLOCK PERIODS

The number of clock periods shown in Table 30 indicates the time required to perform the operations, store the results, and read the next instruction. The number of bus read and write cycles is shown in parenthesis as: (r/w). The number of clock periods and the number of read and write cycles must be added respectively to those of the effective address calculation where indicated.

In Table 30 the headings have the following meanings: An = address register operand, Dn = data register operand, ea = an operand specified by an effective address, and M = memory effective address operand.

• IMMEDIATE INSTRUCTION CLOCK PERIODS

The number of clock periods shown in Table 31 includes the time to fetch immediate operands, perform the operations, store the results, and read the next operation. The number of bus read and write cycles is shown in parenthesis as: (r/w). The number of clock periods and the number of read and write cycles must be added respectively to those of the effective address calculation where indicated.

In Table 31, the headings have the following meanings: # = immediate operand, Dn = data register operand, An = address register operand, M = memory operand, and SR = status register.

• SINGLE OPERAND INSTRUCTION CLOCK PERIODS

Table 32 indicates the number of clock periods for the single operand instructions. The number of bus read and write cycles is shown in parenthesis as: (r/w). The number of clock periods and the number of read and write cycles must be added respectively to those of the effective address calculation where indicated.

Table 27 Effective Address Calculation Timing

Addressing Mode		Byte, Word	Long
	Register		
Dn	Data Register Direct	0(0/0)	0(0/0)
An	Address Register Direct	0(0/0)	0(0/0)
	Memory		
An@	Address Register Indirect	4(1/0)	8(2/0)
An@ +	Address Register Indirect with Postincrement	4(1/0)	8(2/0)
An@ -	Address Register Indirect with Predecrement	6(1/0)	10(2/0)
An@(d)	Address Register Indirect with Displacement	8(2/0)	12(3/0)
An@(d, ix)*	Address Register Indirect with Index	10(2/0)	14(3/0)
xxx. W	Absolute Short	8(2/0)	12(3/0)
xxx. L	Absolute Long	12(3/0)	16(4/0)
PC@(d)	Program Counter with Displacement	8(2/0)	12(3/0)
PC@(d, ix)*	Program Counter with Index	10(2/0)	14(3/0)
#xxx	Immediate	4(1/0)	8(2/0)

* The size of the index register (ix) does not affect execution time.

Table 28 Move Byte and Word Instruction Clock Periods

Source	Destination								
	Dn	An	An@	An@ +	An@ -	An@(d)	An@(d, ix)*	xxx.W	xxx.L
Dn	4(1/0)	4(1/0)	8(1/1)	8(1/1)	8(1/1)	12(2/1)	14(2/1)	12(2/1)	16(3/1)
An	4(1/0)	4(1/0)	8(1/1)	8(1/1)	8(1/1)	12(2/1)	14(2/1)	12(2/1)	16(3/1)
An@	8(2/0)	8(2/0)	12(2/1)	12(2/1)	12(2/1)	16(3/1)	18(3/1)	16(3/1)	20(4/1)
An@+	8(2/0)	8(2/0)	12(2/1)	12(2/1)	12(2/1)	16(3/1)	18(3/1)	16(3/1)	20(4/1)
An@-	10(2/0)	10(2/0)	14(2/1)	14(2/1)	14(2/1)	18(3/1)	20(3/1)	18(3/1)	22(4/1)
An@(d)	12(3/0)	12(3/0)	16(3/1)	16(3/1)	16(3/1)	20(4/1)	22(4/1)	20(4/1)	24(5/1)
An@(d, ix)*	14(3/0)	14(3/0)	18(3/1)	18(3/1)	18(3/1)	22(4/1)	24(4/1)	22(4/1)	26(5/1)
xxx.W	12(3/0)	12(3/0)	16(3/1)	16(3/1)	16(3/1)	20(4/1)	22(4/1)	20(4/1)	24(5/1)
xxx.L	16(4/0)	16(4/0)	20(4/1)	20(4/1)	20(4/1)	24(5/1)	26(5/1)	24(5/1)	28(6/1)
PC@(d)	12(3/0)	12(3/0)	16(3/1)	16(3/1)	16(3/1)	20(4/1)	22(4/1)	20(4/1)	24(5/1)
PC@(d, ix)*	14(3/0)	14(3/0)	18(3/1)	18(3/1)	18(3/1)	22(4/1)	24(4/1)	22(4/1)	26(5/1)
#xxx	8(2/0)	8(2/0)	12(2/1)	12(2/1)	12(2/1)	16(3/1)	18(3/1)	16(3/1)	20(4/1)

* The size of the index register (ix) does not affect execution time.

Table 29 Move Long Instruction Clock Periods

Source	Destination								
	Dn	An	An@	An@ +	An@ -	An@(d)	An@(d, ix)*	xxx.W	xxx.L
Dn	4(1/0)	4(1/0)	12(1/2)	12(1/2)	12(1/2)	16(2/2)	18(2/2)	16(2/2)	20(3/2)
An	4(1/0)	4(1/0)	12(1/2)	12(1/2)	12(1/2)	16(2/2)	18(2/2)	16(2/2)	20(3/2)
An@	12(3/0)	12(3/0)	20(3/2)	20(3/2)	20(3/2)	24(4/2)	26(4/2)	24(4/2)	28(5/2)
An@+	12(3/0)	12(3/0)	20(3/2)	20(3/2)	20(3/2)	24(4/2)	26(4/2)	24(4/2)	28(5/2)
An@-	14(3/0)	14(3/0)	22(3/2)	22(3/2)	22(3/2)	26(4/2)	28(4/2)	26(4/2)	30(5/2)
An@(d)	16(4/0)	16(4/0)	24(4/2)	24(4/2)	24(4/2)	28(5/2)	30(5/2)	28(5/2)	32(6/2)
An@(d, ix)*	18(4/0)	18(4/0)	26(4/2)	26(4/2)	26(4/2)	30(5/2)	32(5/2)	30(5/2)	34(6/2)
xxx.W	16(4/0)	16(4/0)	24(4/2)	24(4/2)	24(4/2)	28(5/2)	30(5/2)	28(5/2)	32(6/2)
xxx.L	20(5/0)	20(5/0)	28(5/2)	28(5/2)	28(5/2)	32(6/2)	34(6/2)	32(6/2)	36(7/2)
PC@(d)	16(4/0)	16(4/0)	24(4/2)	24(4/2)	24(4/2)	28(5/2)	30(5/2)	28(5/2)	32(5/2)
PC@(d, ix)*	18(4/0)	18(4/0)	26(4/2)	26(4/2)	26(4/2)	30(5/2)	32(5/2)	30(5/2)	34(6/2)
#xxx	12(3/0)	12(3/0)	20(3/2)	20(3/2)	20(3/2)	24(4/2)	26(4/2)	24(4/2)	28(5/2)

* The size of the index register (ix) does not affect execution time.

Table 30 Standard Instruction Clock Periods

Instruction	Size	op <ea>, An	op <ea>, Dn	op Dn, <M>
ADD	Byte, Word	8(1/0) +	4(1/0) +	8(1/1) +
	Long	6(1/0) + **	6(1/0) + **	12(1/2) +
AND	Byte, Word	–	4(1/0) +	8(1/1) +
	Long	–	6(1/0) + **	12(1/2) +
CMP	Byte, Word	6(1/0) +	4(1/0) +	–
	Long	6(1/0) +	6(1/0) +	–
DIVS	–	–	158(1/0) + *	–
DIVU	–	–	140(1/0) + *	–
EOR	Byte, Word	–	4(1/0) ***	8(1/1) +
	Long	–	8(1/0) ***	12(1/2) +
MULS	–	–	70(1/0) + *	–
MULU	–	–	70(1/0) + *	–
OR	Byte, Word	–	4(1/0) +	8(1/1) +
	Long	–	6(1/0) + **	12(1/2) +
SUB	Byte, Word	8(1/0) +	4(1/0) +	8(1/1) +
	Long	6(1/0) + **	6(1/0) + **	12(1/2) +

+ add effective address calculation time
* indicates maximum value
** total of 8 clock periods for instruction if the effective address is register direct
*** only available effective address mode is data register direct

Table 31 Immediation Instruction Clock Periods

Instruction	Size	op #, Dn	op #, An	op #, M
ADDI	Byte, Word	8(2/0)	–	12(2/1) +
	Long	16(3/0)	–	20(3/2) +
ADDQ	Byte, Word	4(1/0)	8(1/0)*	8(1/1) +
	Long	8(1/0)	8(1/0)	12(1/2) +
ANDI	Byte, Word	8(2/0)	–	12(2/1) +
	Long	16(3/0)	–	20(3/1) +
CMPI	Byte, Word	8(2/0)	8(2/0)	8(2/0) +
	Long	14(3/0)	14(3/0)	12(3/0) +
EORI	Byte, Word	8(2/0)	–	12(2/1) +
	Long	16(3/0)	–	20(3/2) +
MOVEQ	Long	4(1/0)	–	–
ORI	Byte, Word	8(2/0)	–	12(2/1) +
	Long	16(3/0)	–	20(3/2) +
SUBI	Byte, Word	8(2/0)	–	12(2/1) +
	Long	16(3/0)	–	20(3/2) +
SUBQ	Byte, Word	4(1/0)	8(1/0)*	8(1/1) +
	Long	8(1/0)	8(1/0)	12(1/2) +

+ add effective address calculation time
* word only

Table 32 Single Operand Instruction Clock Periods

Instruction	Size	Register	Memory
CLR	Byte, Word	4(1/0)	8(1/1) +
	Long	6(1/0)	12(1/2) +
NBCD	Byte	6(1/0)	8(1/1) +
NEG	Byte, Word	4(1/0)	8(1/1) +
	Long	6(1/0)	12(1/2) +
NEGX	Byte, Word	4(1/0)	8(1/1) +
	Long	6(1/0)	12(1/2) +
NOT	Byte, Word	4(1/0)	8(1/1) +
	Long	6(1/0)	12(1/2) +
S_{cc}	Byte, False	4(1/0)	8(1/1) +
	Byte, True	6(1/0)	8(1/1) +
TAS	Byte	4(1/0)	10(1/1) +
TST	Byte, Word	4(1/0)	4(1/0) +
	Long	4(1/0)	4(1/0) +

+ add effective address calculation time

• SHIFT/ROTATE INSTRUCTION CLOCK PERIODS

Table 33 indicates the number of clock periods for the shift and rotate instructions. The number of bus read and write cycles is shown in parenthesis as: (r/w). The number of clock periods and the number of read and write cycles must be added respectively to those of the effective address calculation where indicated.

• BIT MANIPULATION INSTRUCTION CLOCK PERIODS

Table 34 indicates the number of clock periods required for the bit manipulation instructions. The number of bus read and write cycles is shown in parenthesis as: (r/w). The number of clock periods and the number of read and write cycles must be added respectively to those of the effective address calculation where indicated.

- **CONDITIONAL INSTRUCTION CLOCK PERIODS**

Table 35 indicates the number of clock periods required for the conditional instructions. The number of bus read and write cycles is indicated in parenthesis as: (r/w). The number of clock periods and the number of read and write cycles must be added respectively to those of the effective address calculation where indicated.

- **JMP, JSR, LEA, PEA, MOVEM INSTRUCTION CLOCK PERIODS**

Table 36 indicates the number of clock periods required for the jump, jump to subroutine, load effective address, push effective address, and move multiple registers instructions. The number of bus read and write cycles is shown in parenthesis as: (r/w).

Table 33 Shift/Rotate Instruction Clock Periods

Instruction	Size	Register	Memory
ASR, ASL	Byte, Word	6 + 2n(1/0)	8(1/1) +
	Long	8 + 2n(1/0)	–
LSR, LSL	Byte, Word	6 + 2n(1/0)	8(1/1) +
	Long	8 + 2n(1/0)	–
ROR, ROL	Byte, Word	6 + 2n(1/0)	8(1/1) +
	Long	8 + 2n(1/0)	–
ROXR, ROXL	Byte, Word	6 + 2n(1/0)	8(1/1) +
	Long	8 + 2n(1/0)	–

Table 34 Bit Manipulation Instruction Clock Periods

Instruction	Size	Dynamic		Static	
		Register	Memory	Register	Memory
BCHG	Byte	–	8(1/1) +	–	12(2/1) +
	Long	8(1/0)*	–	12(2/0)*	–
BCLR	Byte	–	8(1/1) +	–	12(2/1) +
	Long	10(1/0)*	–	14(2/0)*	–
BSET	Byte	–	8(1/1) +	–	12(2/1) +
	Long	8(1/0)*	–	12(2/0)*	–
BTST	Byte	–	4(1/0) +	–	8(2/0) +
	Long	6(1/0)	–	10(2/0)	–

+ add effective address calculation time
* indicates maximum value

Table 35 Conditional Instruction Clock Periods

Instruction	Displacement	Trap or Branch Taken	Trap of Branch Not Taken
B_{cc}	Byte	10(2/0)	8(1/0)
	Word	10(2/0)	12(2/0)
BRA	Byte	10(2/0)	–
	Word	10(2/0)	–
BSR	Byte	18(2/2)	–
	Word	18(2/2)	–
DB_{cc}	CC true	–	12(2/0)
	CC false	10(2/0)	14(3/0)
CHK	–	40(5/3) + *	10(1/0) +
TRAP	–	34(4/3)	–
TRAPV	–	34(5/3)	4(1/0)

+ add effective address calculation time
* indicates maximum value

Table 36 JMP, JSR, LEA, PEA, MOMEM Instruction Clock Periods

Instr	Size	An@	An@ +	An@ -	An@(d)	An@(d, ix)*	xxx.W	xxx.L	PC@(d)	PC@(d, ix)*
JMP	–	8(2/0)	–	–	10(2/0)	14(3/0)	10(2/0)	12(3/0)	10(2/0)	14(3/0)
JSR	–	16(2/2)	–	–	18(2/2)	22(2/2)	18(2/2)	20(3/2)	18(2/2)	22(2/2)
LEA	–	4(1/0)	–	–	8(2/0)	12(2/0)	8(2/0)	12(3/0)	8(2/0)	12(2/0)
PEA	–	12(1/2)	–	–	16(2/2)	20(2/2)	16(2/2)	20(3/2)	16(2/2)	20(2/2)
MOVEM	Word	12+4n (3+n/0)	12+4n (3+n/0)	– –	16+4n (4+n/0)	18+4n (4+n/0)	16+4n (4+n/0)	20+4n (5+n/0)	16+4n (4+n/0)	18+4n (4+n/0)
M → R	Long	12+8n (3+2n/0)	12+8n (3+2n/0)	– –	16+8n (4+2n/0)	18+8n (4+2n/0)	16+8n (4+2n/0)	20+8n (5+2n/0)	16+8n (4+2n/0)	18+8n (4+2n/0)
MOVEM	Word	8+4n (2/n)	– –	8+4n (2/n)	12+4n (3/n)	14+4n (3/n)	12+4n (3/n)	16+4n (4/n)	– –	– –
R → M	Long	8+8n (2/2n)	– –	8+8n (2/2n)	12+8n (3/2n)	14+8n (3/2n)	12+8n (3/2n)	16+8n (4/2n)	– –	– –

n is the number of registers to move

* is the size of the index register (ix) does not affect the instruction's execution time

• MULTI-PRECISION INSTRUCTION CLOCK PERIODS

Table 37 indicates the number of clock periods for the multi-precision instructions. The number of clock periods includes the time to fetch both operands, perform the operations, store the results, and read the next instructions. The number of read and write cycles is shown in parenthesis as: (r/w).

In Table 37, the headings have the following meanings: Dn = data register operand and M = memory operand.

Table 37 Multi-Precision Instruction Clock Periods

Instruction	Size	op Dn, Dn	op M, M
ADDX	Byte, Word	4(1/0)	18(3/1)
	Long	8(1/0)	30(5/2)
CMPM	Byte, Word	–	12(3/0)
	Long	–	20(5/0)
SUBX	Byte, Word	4(1/0)	18(3/1)
	Long	8(1/0)	30(5/2)
ABCD	Byte	6(1/0)	18(3/1)
SBCD	Byte	6(1/0)	18(3/1)

• MISCELLANEOUS INSTRUCTION CLOCK PERIODS

Table 38 indicates the number of clock periods for the following miscellaneous instructions. The number of bus read and write cycles is shown in parenthesis as: (r/w). The number of clock periods plus the number of read and write cycles must be added to those of the effective address calculation where indicated.

• EXCEPTION PROCESSING CLOCK PERIODS

Table 39 indicates the number of clock periods for exception processing. The number of clock periods includes the time for all stacking, the vector fetch, and the fetch of the first instruction of the handler routine. The number of bus read and write cycles is shown in parenthesis as: (r/w).

Table 38 Miscellaneous Instruction Clock Periods

Instruction	Size	Register	Memory	Register → Memory	Memory → Register
MOVE from SR	–	6(1/0)	8(1/1) +	–	–
MOVE to CCR	–	12(2/0)	12(2/0) +	–	–
MOVE to SR	–	12(2/0)	12(2/0) +	–	–
MOVEP	Word	–	–	16(2/2)	16(4/0)
	Long	–	–	24(2/4)	24(6/0)
EXG	–	6(1/0)	–	–	–
EXT	Word	4(1/0)	–	–	–
	Long	4(1/0)	–	–	–
LINK	–	16(2/2)	–	–	–
MOVE from USP	–	4(1/0)	–	–	–
MOVE to USP	–	4(1/0)	–	–	–
NOP	–	4(1/0)	–	–	–
RESET	–	132(1/0)	–	–	–
RTE	–	20(5/0)	–	–	–
RTR	–	20(5/0	–	–	–
RTS	–	16(4/0)	–	–	–
STOP	–	4(0/0)	–	–	–
SWAP	–	4(1/0)	–	–	–
UNLK	–	12(3/0)	–	–	–

+ add effective address calculation time

Table 39 Exception Processing Clock Periods

Exception	Periods
Address Error	50(4/7)
Bus Error	50(4/7)
Interrupt	44(5/3)*
Illegal Instruction	34(4/3)
Privileged Instruction	34(4/3)
Trace	34(4/3)

* The interrupt acknowledge bus cycle is assumed to take four external clock periods.

■ APPENDIX

• The Mask Version Deviation Between 68000R and 68000

(1) Electrical Characteristics

The deviations of DC characteristics and AC characteristics between 68000R and 68000 are shown in Table 40 and 41.

(2) Function Code

The function code lines of 68000R are driven during the first phase one of a bus cycle (S0) to allow faster MMU operation. This is one half clock sooner than on 68000.

(3) Read-Modify-Write Cycle

The write half of a read-modify-write cycle of 68000R starts one clock later than on 68000 (See Figures 53 and 54).

(4) Retry Function

On 68000, $\overline{HALT}$ is required to be asserted before $\overline{BERR}$ to assure correct retry operation. 68000R allows $\overline{HALT}$ to skew somewhat into the next sample period (See Figure 55).

(5) Three Cycle Read

Three cycle reads on 68000 are possible when $\overline{DTACK}$ is recognized as early as S2. 68000R allows $\overline{DTACK}$ to be recognized in S4, S6, etc., which terminates the cycle. 68000R makes a four cycle read the fastest read cycle (See Figure 56).

(6) Write Cycle Bus Error

On 68000 during a write cycle where $\overline{BERR}$ is asserted, the R/$\overline{W}$ line switches to the Read mode before $\overline{AS}$, $\overline{DS}$, and Address and Data lines negate. This differs from a normal write cycle termination. 68000R allows a normal termination in this case (See Figure 57).

(7) Interrupt Synchronization Problem

On 68000 the interrupt logic does not debounce and synchronize the interrupt inputs properly. Therefore $\overline{IPL_0} \sim \overline{IPL_2}$ require an external synchronization circuit (See Figure 58). On 68000R the change detector logic is modified to correct this problem.

Table 40 DC Characteristics (V_{CC} = 5V ±5%, V_{SS} = 0V, Ta = 0 ~ +70°C, unless otherwise noted.)

Item		Symbol	68000R Test Condition	68000R min	68000R typ	68000R max	68000 Test Condition	68000 min	68000 typ	68000 max	Unit
Input "High" Voltage		V_{IH}		2.0	–	V_{CC}		2.0	–	V_{CC}	V
Input "Low" Voltage		V_{IL}		V_{SS}-0.3	–	0.8		V_{SS}-0.3	–	0.8	V
Input Leakage Current	$\overline{BERR}$, $\overline{BGACK}$, $\overline{BR}$, $\overline{DTACK}$, CLK, $\overline{IPL_0} \sim \overline{IPL_2}$, $\overline{VPA}$	I_{in}	@5.25V	–	–	2.5		–	1.0	–	μA
	$\overline{HALT}$, $\overline{RES}$			–	–	20		–	2.0	–	
Three-State (Off State) Input Current	$\overline{AS}$, $A_1 \sim A_{23}$, $D_0 \sim D_{15}$, $FC_0 \sim FC_2$, $\overline{LDS}$, R/$\overline{W}$, $\overline{UDS}$, $\overline{VMA}$	I_{TSI}	@2.4V/0.4V	–	–	20		–	7.0	–	μA
Output High Voltage	E*	V_{OH}	I_{OH}=-400μA	V_{CC}-0.75	–	–	I_{OH}=-400μA	2.4	–	–	V
	$\overline{AS}$, $A_1 \sim A_{23}$, $\overline{BG}$, $D_0 \sim D_{15}$, $FC_0 \sim FC_2$, $\overline{LDS}$, R/$\overline{W}$, $\overline{UDS}$, $\overline{VMA}$			2.4	–	–					
Output "Low" Voltage	$\overline{HALT}$	V_{OL}	I_{OL}=1.6mA	–	–	0.5	I_{OL}=1.6mA	–	–	0.5	V
	$A_1 \sim A_{23}$, $\overline{BG}$, $FC_0 \sim FC_2$		I_{OL}=3.2mA	–	–	0.5	I_{OL}=3.2mA	–	–	0.5	
	$\overline{RES}$		I_{OL}=5.3mA	–	–	0.5	I_{OL}=5.0mA	–	–	0.5	
	E, $\overline{AS}$, $D_0 \sim D_{15}$, $\overline{LDS}$, R/$\overline{W}$, $\overline{UDS}$, $\overline{VMA}$		I_{OL}=5.3mA	–	–	0.5	I_{OL}=5.3mA	–	–	0.5	
Power Dissipation		P_D	Clock Frequency=8MHz	–		1.5	Clock Frequcney=8MHz	–	1.0	–	W
Capacitance		C_{in}	V_{in}=0V, T_a = 25°C, Frequency=1MHz	–	10.0	20.0	V_{in}=0V, T_a = 25°C, Frequency=1MHz	–	10.0	20.0	pF

* With external pullup resistor of 470Ω

Table 41 AC Characteristics (V_{CC} = 5V ±5%, V_{SS} = 0V, Ta = 0 ~ +70°C, unless otherwise noted.)

Item	Symbol	Mask Version	68000R								68000						Unit
			4 MHz HD68000-4		6 MHz HD68000-6		8 MHz HD68000-8		10 MHz HD68000-10		4 MHz HD68000-4		6 MHz HD68000-6		8 MHz HD68000-8		
			min	max	min	max	min	max	min	max	min	max	min	max	min	max	
Frequency of Operation	f		2.0	4.0	2.0	6.0	2.0	8.0	2.0	10.0	2.0	4.0	2.0	6.0	2.0	8.0	MHz
Clock Period	t_{cyc}		250	500	167	500	125	500	100	500	250	500	167	500	125	500	ns
Clock Width Low	t_{CL}		115	250	75	250	55	250	45	250	115	250	75	250	55	250	ns
Clock Width High	t_{CH}		115	250	75	250	55	250	45	250	115	250	75	250	55	250	ns
Clock Fall Time	t_{Cf}		–	10	–	10	–	10	–	10	–	10	–	10	–	10	ns
Clock Rise Time	t_{Cr}		–	10	–	10	–	10	–	10	–	10	–	10	–	10	ns
Clock Low to Address	t_{CLAV}		–	90	–	80	–	70	–	55	–	90	–	90	–	90	ns
Clock High to FC Valid	t_{CHFCV}		–	90	–	80	–	70	–	60	–	–	–	–	–	–	ns
Clock High to Address Data High Impedance (Maximum)	t_{CHAZx}		–	120	–	100	–	80	–	70	–	120	–	100	–	100	ns
Clock High to Address/FC Invalid (Minimum)	t_{CHAZn}		0	–	0	–	0	–	0	–	20	–	20	–	20	–	ns
Clock High to $\overline{AS}$, $\overline{DS}$ Low (Maximum)	t_{CHSLx}		–	80	–	70	–	60	–	55	–	80	–	70	–	70	ns
Clock High to $\overline{AS}$, $\overline{DS}$ Low (Minimum)	t_{CHSLn}		0	–	0	–	0	–	0	–	20	–	20	–	20	–	ns
Address to $\overline{AS}$, $\overline{DS}$ (Read) Low/$\overline{AS}$ Write	t_{AVSL}		55	–	35	–	30	–	20	–	55	–	35	–	30	–	ns
FC Valid to $\overline{AS}$, $\overline{DS}$, (Read) Low/$\overline{AS}$ Write	t_{FCVSL}		80	–	70	–	60	–	50	–	–	–	–	–	–	–	ns
Clock Low to $\overline{AS}$, $\overline{DS}$ High	t_{CLSH}		–	90	–	80	–	70	–	55	–	90	–	80	–	70	ns
$\overline{AS}$, $\overline{DS}$ High to Address/FC Invalid	t_{SHAZ}		60	–	40	–	30	–	20	–	60	–	40	–	30	–	ns
$\overline{AS}$, $\overline{DS}$ Width Low (Read)/$\overline{AS}$ Write	t_{SL}		535	–	337	–	240	–	195	–	285	–	170	–	115	–	ns
$\overline{DS}$ Width Low (Write)	–		285	–	170	–	115	–	95	–	–	–	–	–	–	–	ns
$\overline{AS}$, $\overline{DS}$ Width High	t_{SH}		285	–	180	–	150	–	105	–	285	–	180	–	150	–	ns
Clock High to $\overline{AS}$, $\overline{DS}$ High Impedance	t_{CHSZ}		–	120	–	100	–	80	–	70	–	120	–	100	–	80	ns
$\overline{AS}$, $\overline{DS}$ High to R/$\overline{W}$ High	t_{SHRH}		60	–	50	–	40	–	20	–	60	–	50	–	30	–	ns
Clock High to R/$\overline{W}$ High (Maximum)	t_{CHRHx}		–	90	–	80	–	70	–	60	–	90	–	80	–	70	ns
Clock High to R/$\overline{W}$ High (Minimum)	t_{CHRHn}		0	–	0	–	0	–	0	–	10	–	10	–	10	–	ns
Clock High to R/$\overline{W}$ Low	t_{CHRL}		–	90	–	80	–	70	–	60	–	90	–	80	–	70	ns
Address Valid to R/$\overline{W}$ Low	t_{AVRL}		45	–	25	–	20	–	0	–	45	–	25	–	10	–	ns
FC Valid to R/$\overline{W}$ Low	t_{FCVRL}		80	–	70	–	60	–	50	–	80	–	70	–	60	–	ns
R/$\overline{W}$ Low to $\overline{DS}$ Low (Write)	t_{RLSL}		200	–	140	–	80	–	50	–	200	–	140	–	80	–	ns
Clock Low to Data Out Valid	t_{CLDO}		–	90	–	80	–	70	–	55	–	90	–	80	–	80	ns
Clock High to R/$\overline{W}$, $\overline{VMA}$ High Impedance	t_{CHRZ}		–	120	–	100	–	80	–	70	–	120	–	100	–	80	ns
$\overline{DS}$ High to Data Out Invalid	t_{SHDO}		60	–	40	–	30	–	20	–	60	–	40	–	30	–	ns
Data Out Valid to $\overline{DS}$ Low (Write)	t_{DOSL}		55	–	35	–	30	–	20	–	55	–	35	–	30	–	ns
Data In to Clock Low (Setup Time)	t_{DICL}		30	–	25	–	15	–	15	–	30	–	25	–	15	–	ns
$\overline{AS}$, $\overline{DS}$ High to $\overline{DTACK}$ High	t_{SHDAH}		0	240	0	160	0	120	0	90	0	240	0	160	0	120	ns
$\overline{DS}$ High to Data Invalid (Hold Time)	t_{SHDI}		0	–	0	–	0	–	0	–	0	–	0	–	0	–	ns
$\overline{AS}$, $\overline{DS}$ High to $\overline{BERR}$ High	t_{SHBEH}		0	–	0	–	0	–	0	–	0	–	0	–	0	–	ns
$\overline{DTACK}$ Low to Data In (Setup Time)	t_{DALDI}		–	180	–	120	–	90	–	65	–	180	–	120	–	90	ns
$\overline{HALT}$ and $\overline{RES}$ Input Transition Time	t_{RHrf}		0	200	0	200	0	200	0	200	0	200	0	200	0	200	ns
Clock High to $\overline{BG}$ Low	t_{CHGL}		–	90	–	80	–	70	–	60	–	90	–	80	–	70	ns
Clock High to $\overline{BG}$ High	t_{CHGH}		–	90	–	80	–	70	–	60	–	90	–	80	–	70	ns
$\overline{BR}$ Low to $\overline{BG}$ Low	t_{BRLGL}		1.5	3.0	1.5	3.0	1.5	3.0	1.5	3.0	1.5	3.0	1.5	3.0	1.5	3.0	Clk. Per.
$\overline{BR}$ High to $\overline{BG}$ High	t_{BRHGH}		1.5	3.0	1.5	3.0	1.5	3.0	1.5	3.0	1.5	3.0	1.5	3.0	1.5	3.0	Clk. Per.
$\overline{BGACK}$ Low to $\overline{BG}$ High	t_{GALGH}		1.5	3.0	1.5	3.0	1.5	3.0	1.5	3.0	1.5	3.0	1.5	3.0	1.5	3.0	Clk. Per.
$\overline{BG}$ Low to Bus High Impedance (With $\overline{AS}$ High)	t_{GLZ}		–	120	–	100	–	80	–	70	0	1.5*	0	1.5*	0	1.5*	ns
$\overline{BG}$ Width High	t_{GH}		1.5	–	1.5	–	1.5	–	1.5	–	1.5	–	1.5	–	1.5	–	Clk. Per.
Clock Low to $\overline{VMA}$ Low	t_{CLVML}		–	90	–	80	–	70	–	70	–	90	–	80	–	70	ns
Clock Low to E Transition	t_{CLE}		–	100	–	85	–	70	–	55	–	100	–	85	–	70	ns
E Output Rise and Fall Time	t_{Erf}		–	25	–	25	–	25	–	25	–	25	–	25	–	25	ns
$\overline{VMA}$ Low to E High	t_{VMLEH}		325	–	240	–	200	–	150	–	325	–	240	–	200	–	ns
$\overline{AS}$, $\overline{DS}$ High to $\overline{VPA}$ High	t_{SHVPH}		0	240	0	160	0	120	0	90	0	240	0	160	0	120	ns
E Low to Address/$\overline{VMA}$/FC Invalid	t_{ELAI}		55	–	35	–	30	–	10	–	55	–	35	–	30	–	ns
$\overline{BGACK}$ Width	t_{BGL}		1.5	–	1.5	–	1.5	–	1.5	–	1.5	–	1.5	–	1.5	–	Clk. Per.
Asynchronous Input Setup Time	t_{ASI}		30	–	25	–	20	–	20	–	30	–	25	–	20	–	ns
$\overline{BERR}$ Low to $\overline{DTACK}$ Low	t_{BELDAL}		50	–	50	–	50	–	50	–	50	–	50	–	50	–	ns
E Low to $\overline{AS}$, $\overline{DS}$ Invalid	t_{ELSI}		-80	–	-80	–	-80	–	-80	–	-80	–	-80	–	-80	–	ns
E Width High	t_{EH}		900	–	600	–	450	–	350	–	900	–	600	–	450	–	ns
E Width Low	t_{EL}		1400	–	900	–	700	–	550	–	1400	–	900	–	700	–	ns
E Extended Rise Time	t_{CIEHX}		80	–	80	–	80	–	80	–	–	–	–	–	–	–	ns
Data Hold from Clock High	t_{CHDO}		0	–	0	–	0	–	0	–	–	–	–	–	–	–	ns
Data Hold from E Low (Write)	t_{ELDOZ}		60	–	40	–	30	–	20	–	20	–	20	–	20	–	ns
R/$\overline{W}$ to Data Bus Impedance Change	t_{RLDO}		55	–	35	–	30	–	20	–	–	–	–	–	–	–	ns
Halt/$\overline{RES}$ Pulse Width	t_{HRPW}		10	–	10	–	10	–	10	–	–	–	–	–	–	–	Clk. Per.

* These unit are Clk. Per.

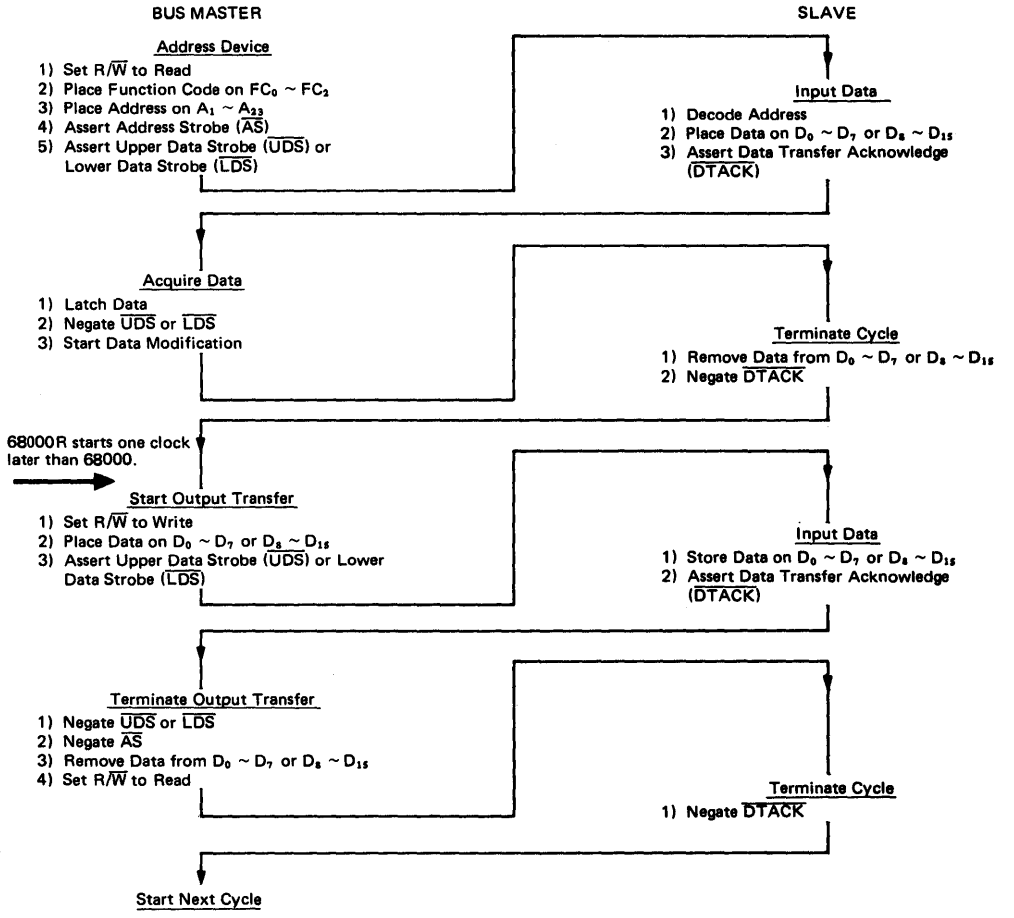


Figure 53 Read-Modify-Write Cycle Flow Chart

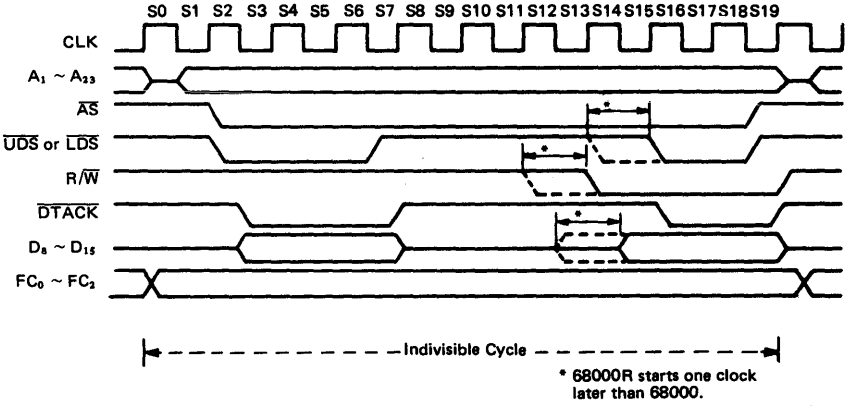


Figure 54 Read-Modify-Write Cycle Timing Diagram

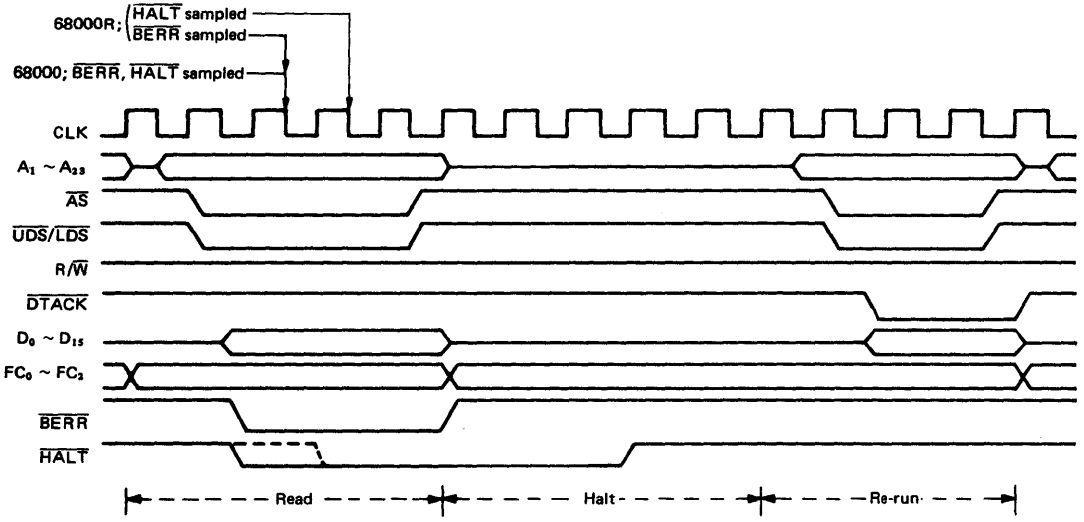


Figure 55 Re-Run Bus Cycle Timing Information

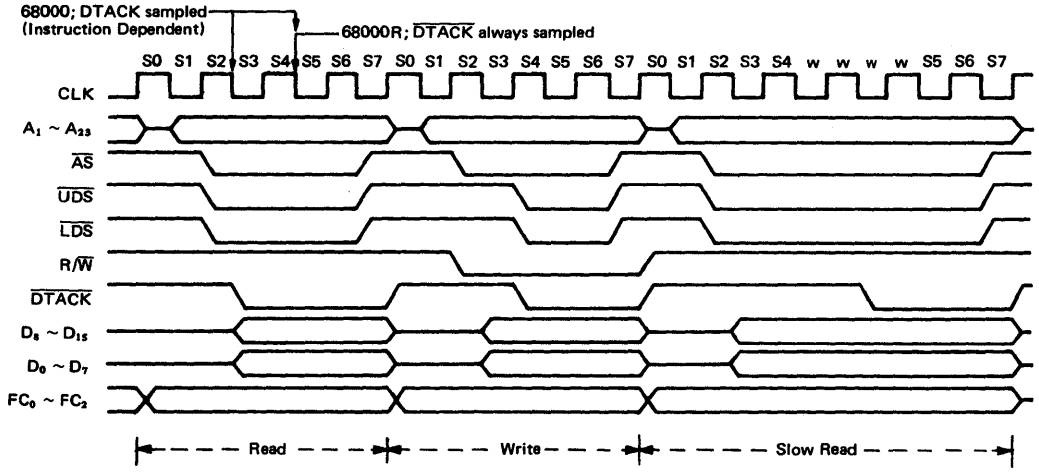


Figure 56 Read and Write Cycle Timing Diagram

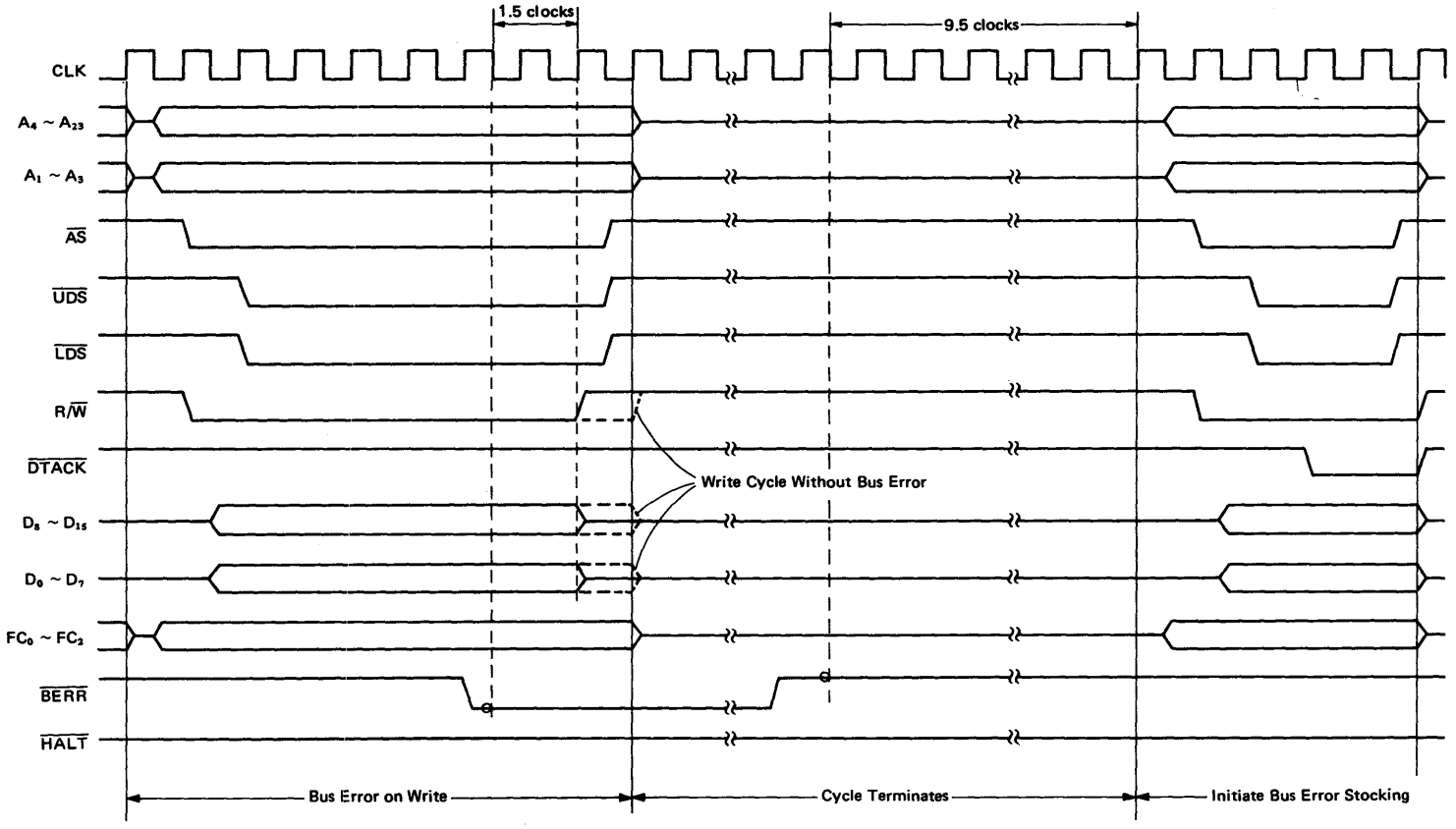
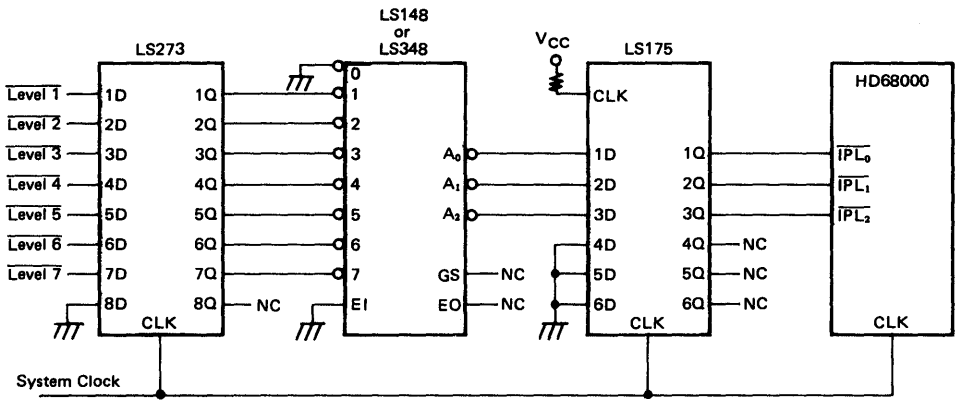


Figure 57 Bus Error (Write Cycle) Timing



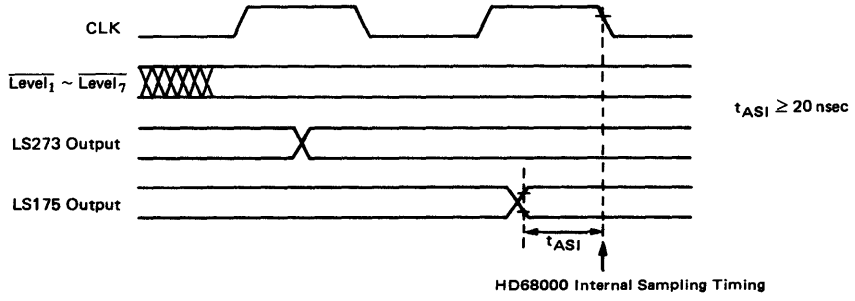


Figure 58 External Synchronizer

It is possible that the uncorrect interrupt exception occurs if the $\overline{IPL_0} \sim \overline{IPL_2}$ changes in TASI or in CLK low time.

In the circuit above, the LS273 latches the incoming asynchronous interrupt requests on the resing edge of the clock. The LS175 is needed because of the delay through the LS273 and LS148 (LS348), and the input setup requirements of HD68000 IPL pins.

HD68450

DMAC (Direct Memory Access Controller)

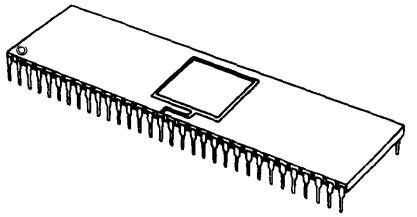
—ADVANCE INFORMATION—

HD68450 is a DMA Controller for the HMCS68000 16-bit microprocessor system. Increasingly large amounts of data are being processed by the 16-bit microprocessor systems and, consequently, the ability to transfer large amounts of data in a large memory space becomes a necessity. HD68450 has been designed to meet this requirement in a highly efficient manner.

HD68450 has 4 independent DMA channels of operation with programmable channel priorities. It can handle data sizes of byte, word (16-bits), and longword (32-bits), and has a direct addressing range of 16 megabytes. It performs 16-bit DMA transfers on an asynchronous bus as well as synchronous transfers with 8-bit HMCS6800 peripheral LSI's using the enable signal. It outputs function code signal for memory management and it can handle bus error, halt, and retry operations to compliment the highly reliable HMCS68000 system.

The transfer modes of HD68450 consists of transfer between memory and peripheral device, and also between memories. Transfer of blocks of data can be done by using the continue mode, array chain mode, or linked array chain mode. Single addressing mode is provided for transfer between memory and device having the same port size, as well as dual addressing mode for different port sizes. In the dual addressing mode, transfer is done in two bus cycles – memory to DMAC, then DMAC to device. As can be seen by its many features, HD68450 is a highly intelligent device to meet the different data transfer requirements for each individual applications.

HD68450

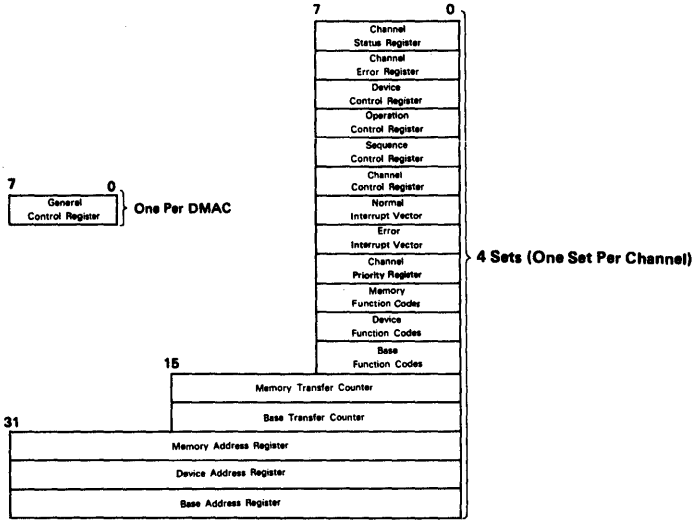


(DC-64)

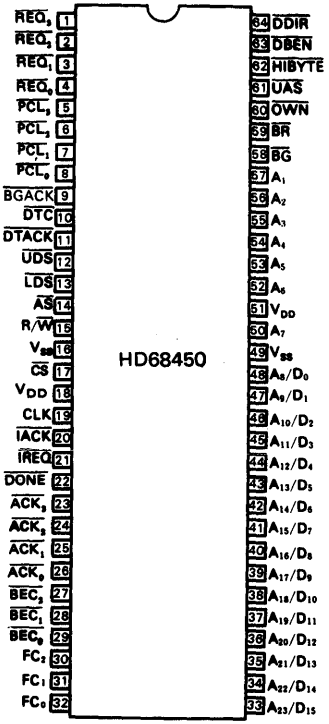
■ FEATURES

- HMCS68000 Bus Compatible
- Interfaces Directly with HMCS68000/HMCS6800 Peripherals
- Memory-to-Device, Device-to-Memory, and Memory-to-Memory Transfers.
- Continue Mode and Array Chained, Linked Array Chained Operations
- 4 Independent Channels with Programmable Priorities
- Handles Byte, Word, and Longword Data Sizes
- External Request Mode and Auto-Request Mode
- Maximum Transfer Rate of 2 Mega Word/Sec

■ PROGRAMMING MODEL



■ PIN ARRANGEMENT



(Top View)

INTRODUCTION OF THE RELATED DEVICES

- MOS Memories
- TTL HD74/HD74S/HD74LS Series
- Advanced Low Power Schottky TTL HD74ALS Series
- CMOS Logic HDI4000B/UB Series
- Linear ICs
- Interface Circuits

MOS Memories

■ TYPICAL CHARACTERISTICS OF MOS MEMORY

● MOS RAM

Mode	Total Bit	Type No.	Process	Organization (word x bit)	Access Time (ns) max	Cycle Time (ns) min	Supply Voltage (V)	Power Dissipation (W)	Package** Pin No.	C	G	P	FP	Replacement
Static	4k-bit	HM472114A-1	NMOS	1024x4	150	150	+5	0.2	18		●	●		
		HM472114A-2			200	200					●	●		2114L-2
		HM472114-3			300	300					●	●		2114L-3
		HM472114-4			450	450					●	●		2114L-4
		HM4334-3	CMOS	1024x4	300	460	+5	10μ/20m	18		●	●		HM-6514
		HM4334-4			450	640					●	●		
		HM4334-3L			300	460		10μ/20m				●		
		HM4334-4L			450	640						●		
		HM6148	CMOS	1024x4	70	70	+5	0.1m/0.2	18		●	●		2148
		HM6148-6			85	85					●	●		2148-6
		HM6148L			70	70		5μ/0.2				●		
		HM6148L-6			85	85						●		
		HM4315	CMOS	4096x1	450	640	+5	10μ/20m	18			●		HM-6504
		HM6147	CMOS	4096x1	70	70	+5	0.1m/75m	18		●	●		2147
		HM6147-3			55	55					●	●		2147-3
		HM6147L			70	70		5μ/5m				●		
		HM6147L-3			55	55						●		
		HM6147H-35*	CMOS	4096x1	35	35	+5	0.1m/0.15	18		●	●		2147H-1
		HM6147H-45*			45	45					●	●		2147H-2
		HM6147HL-35*			35	35		5μ/0.15				●		
		HM6147HL-45*			45	45						●		
	16k-bit	HM6116-2	CMOS	2048x8	120	120	+5	0.1m/0.18	24		●	●	●	
		HM6116-3			150	150					●	●	●	
		HM6116-4			200	200					●	●	●	
		HM6116L-2			120	120		20△μ/0.16			●	●	●	
		HM6116L-3			150	150					●	●	●	
		HM6116L-4			200	200					●	●	●	
		HM6117-3*	CMOS	2048x8	150	150	+5	0.1m/0.2	24			●	●	
		HM6117-4*			200	200						●	●	
		HM6117L-3*			150	150		10μ/0.2				●	●	
		HM6117L-4*			200	200						●	●	
		HM6167	CMOS	16384x1	70	70	+5	25m/0.15	20		●	●		2167
		HM6167-6			85	85					●	●		2167-6
		HM6167-8			100	100					●	●		2167-8
		HM6167L			70	70		5μ/0.15				●		
		HM6167L-6			85	85						●		
		HM6167L-8			100	100						●		
Dynamic	16k-bit	HM4716A-1	NMOS	16384x1	120	320	+12, +5, -5	0.35	16		●	●		
		HM4716A-2			150	320					●	●		MK4116-2
		HM4716A-3			200	375					●	●		MK4116-3
		HM4716A-4			250	410					●	●		MK4116-4
		HM4816A-3	NMOS	16384x1	100	235	+5	11m/0.15	16		●	●		2118-3
		HM4816A-3E			105	200					●	●		
		HM4816A-4			120	270					●	●		2118-4
		HM4816A-7			150	320					●	●		2118-7
	64k-bit	HM4864-2	NMOS	65536x1	150	270	+5	20m/0.33	16		●	●		
		HM4864-3			200	335					●	●		
		HM4864A-12*			120	230					●	●		
		HM4864A-15*			150	260		20m/0.275			●	●		
		HM4864A-20*			200	330					●	●		

* Preliminary △ HM6116LP Series: 10μW

** The package codes of P, G, C, and FP are applied to the package materials as follows.
P: Plastic DIP, G: Cerdip, C: Side-brazed Ceramic DIP, FP: Small Sized Flat Package.

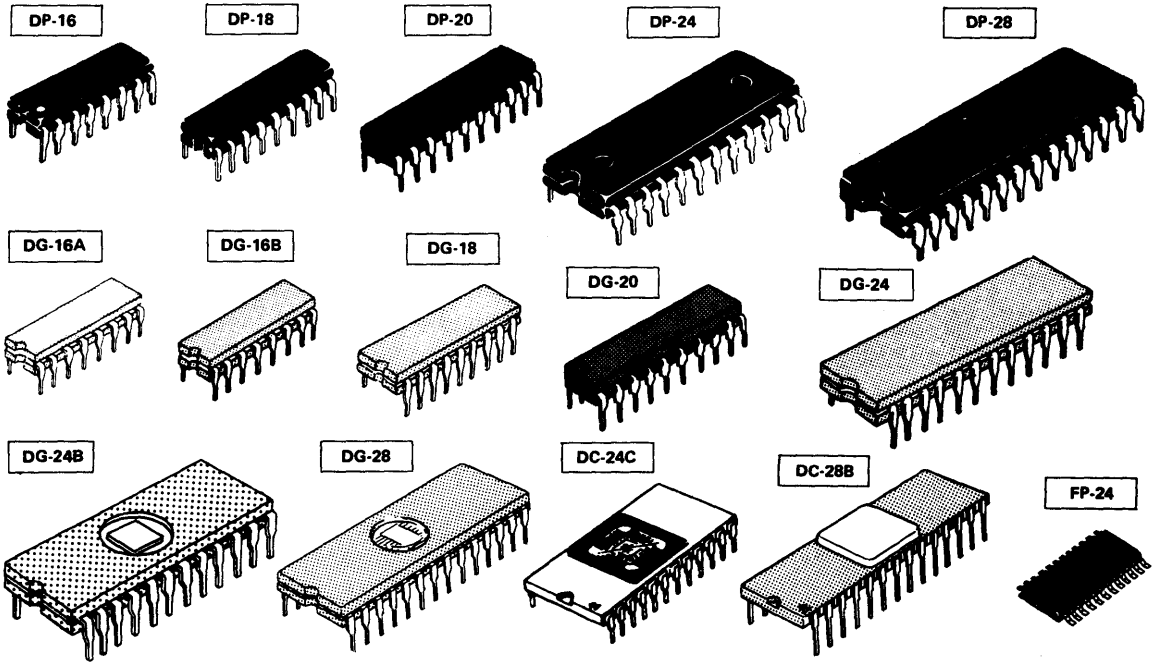
MOS Memories

• MOS ROM

Program	Total Bit	Type No.	Process	Organization (word x bit)	Access Time (ns) max	Supply Voltage (V)	Power Dissipation (W)	Package*** Pin No.	C	G	P	Replacement
Mask	32k-bit	HN46332	NMOS	4096x8	350	+5	0.25	24			•	
	64k-bit	HN48364		8192x8	350		0.225	24			•	
	128k-bit	HN43128	CMOS	16384x8 32768x4	6000		3m	28			•	
		HN613128*		16384x8	250		5μ/0.1	28			•	
	256k-bit	HN61256		32768x8 65536x4	3000		3m	28			•	
U.V. Erasable & Electrically	16k-bit	HN462716	NMOS	2048x8	450	+5	0.555	24	•	•		2716
		HN462716-1			350		0.555			•		2716-1
		HN462716-2			390					•		2716-2
	32k-bit	HN462532	NMOS	4096x8	450	+5	0.858	24	•	•		TMS2532
		HN462532-2			390					•		
		HN462532L			450		0.543			•		TMS25L32
		HN462732	NMOS	4096x8	450	+5	0.788	24	•	•		2732
		HN462732-2			390					•		
		HN482732A-20**	NMOS	4096x8	200	+5	–	24		•		2732A-2
		HN482732A-25**			250					•		2732A
		HN482732A-30**			300					•		2732A-3
	64k-bit	HN482764*	NMOS	8192x8	250	+5	0.555	28	•	•		2764
		HN482764-3*			300				•	•		2764-3
		HN482764-4			450				•	•		
Electrically Erasable	16k-bit	HN48016*	NMOS	2048x8	350	+5	0.3	24			•	

* Preliminary
** Under development
*** The package codes of P, G, and C are applied to the package materials as follows.
P: Plastic DIP, G: Cerdip, C: Side-brazed Ceramic DIP

■ OUTLINE



TTL HD74/HD74S/HD74LS Series

■ **PERFORMANCE (per gate)**

Performance	HD74 Series	HD74S Series	HD74LS Series
Propagation Delay Time	10ns	3ns	10ns
Power Dissipation	10mW	20mW	2mW
Speed-Power Product	100pJ	60pJ	20pJ

■ **MAIN CHARACTERISTICS (Ta = −20~+75°C)**

Parameter \ Series	HD74 Series		HD74S Series		HD74LS Series	
	min.	max.	min.	max.	min.	max.
V_{OL} (I_{OL} max)	–	0.4V	–	0.5V	–	0.5V
V_{OH} (I_{OH} = −400μA)	2.4V	–	2.7V	–	2.7V	–
V_{IL}	–	0.8V	–	0.8V	–	0.8V
V_{IH}	2V	–	2V	–	2V	–
I_{IL}	–	−1.6mA	–	−2mA	–	−0.4mA
I_{IH} (V_{IH} min)	–	40μA	–	50μA	–	20μA

■ **SELECTION GUIDE**

● **NAND/NOR/AND/OR GATES**

Function	HD74 Series	HD74S Series	HD74LS Series
Quad. 2-input Positive NAND Gates	00	00	00
Quad. 2-input Positive NAND Gates (with Open Collector Outputs)	01	–	01
Quad. 2-input Positive NOR Gates	02	02	02
Quad. 2-input Positive NAND Gates (with Open Collector Outputs)	03	03	03
Hex Inverters	04	04	04
Hex Inverters (with Open Collector Outputs)	05	05	05
Hex Inverter Buffers/Drivers (with Open Collector High-voltage Outputs)	06	–	–
Hex Buffers/Drivers (with Open Collector High-voltage Outputs)	07	–	–
Quad. 2-input Positive AND Gates	08	–	08
Quad. 2-input Positive AND Gates (with Open Collector Outputs)	09	–	09
Triple 3-input Positive NAND Gates	10	10	10
Triple 3-input Positive AND Gates	–	11	11
Triple 3-input Positive NAND Gates (with Open Collector Outputs)	12	12	12
Dual 4-input Schmitt NAND Gates	13	–	13
Hex Schmitt-trigger Inverters	14	–	14
Triple 3-input Positive AND Gates (with Open Collector Outputs)	–	15	15
Hex Inverter Buffers/Drivers (with Open Collector High-voltage Outputs)	16	–	–
Hex Buffers/Drivers (with Open Collector High-voltage Outputs)	17	–	–
Dual 4-input Positive NAND Gates	20	20	20
Dual 4-input Positive AND Gates	–	–	21
Dual 4-input Positive NAND Gates (with Open Collector Outputs)	22	22	22
Expandable Dual 4-input Positive NOR Gates (with Strobe)	23	–	–
Dual 4-input Positive NOR Gates	25	–	–
Quad. 2-input High-voltage Interface NAND Gates	26	–	26
Triple 3-input Positive NOR Gates	27	–	27
8-input Positive NAND Gate	30	–	30
Quad. 2-input Positive OR Gates	32	–	32
Quad. 2-input Positive NAND Buffers	37	–	37
Quad. 2-input Positive NAND Buffers (with Open Collector Outputs)	38	–	38
Dual 4-input Positive NAND Buffers	40	40	40
Quad. Bus Buffer Gates with 3-state Output (Inverting)	125	–	125A
Quad. Bus Buffer Gates with 3-state Output (Noninverting)	126	–	126A
Quad. 2-input Positive NAND Schmitt Triggers	132	–	132
13-input Positive NAND Gate	–	133	–
12-input Positive NAND Gate (with 3-state Out.)	–	134	–
Dual 4-input Positve NAND Line Drivers	–	140	–
Hex Bus Buffers/Drivers (with 3-state Outputs)	–	–	365A
Hex Bus Buffers/Drivers (with 3-state Outputs)	–	–	366A
Hex Bus Buffers/Drivers (with 3-state Outputs)	–	–	367A
Hex Bus Buffers/Drivers (with 3-state Outputs)	–	–	368A

(to be continued)

• AND-OR-INVERT GATES

Function	HD74 Series	HD74S Series	HD74LS Series
Expandable Dual 2-wide 2-input AND-OR-INVERT Gates	50	–	–
Dual 2-wide 2-input AND-OR-INVERT Gates	51	–	51
Expandable 4-wide 2-input AND-OR-INVERT Gate	53	–	–
4-wide 2-input AND-OR-INVERT Gate	54	–	54
2-wide 4-input AND-OR-INVERT Gate	–	–	55
4-2-3-2-input AND-OR-INVERT Gate	–	64	–
4-2-3-2-input AND-OR-INVERT Gate (with Open Collector Outputs)	–	65	–

• EXPANDER

Function	HD74 Series	HD74S Series	HD74LS Series
Dual 4-input Expanders	60	–	–

• FLIP FLOPS

Function	HD74 Series	HD74S Series	HD74LS Series
J-K Master-Slave Flip Flop (AND Inputs)	72	–	–
Dual J-K Flip Flops	73	–	73
Dual D-type Edge-triggered Flip Flops	74	74	74A
Dual J-K Flip Flops (with PR and CLR)	76	–	76
Dual J-K Flip Flops (with PR, Common CLR, and Common CK)	–	–	78
Dual J-K Flip Flops	107	–	107
Dual J-$\overline{K}$ Positive Edge-triggered Flip Flops (with PR and CLR)	–	–	109A
Dual J-K Negative-edge-triggered Flip Flops (with PR and CLR)	–	112	112
Dual J-K Negative-edge-triggered Flip Flops (with PR)	–	113	113
Dual J-K Negative-edge-triggered Flip Flops (with PR, Common CLR, and Common CK)	–	114	114
Monostable Multivibrator	121	–	–
Retriggerable Monostable Multivibrator	–	–	122
Dual Retriggerable Monostable Multivibrators	123	–	123
Hex D-type Flip Flops (with CLR)	174	174	174
Quad. D-type Flip Flops (with CLR)	175	175	175
Dual Monostable Multivibrators (with Schmitt Trigger)	221	–	221
Octal D-type Flip-Flops (with Common CK, and Single-Rail Outputs)	–	–	273

• COUNTERS

Function	HD74 Series	HD74S Series	HD74LS Series
Decade Counter	90A	–	90
Divide-by-Twelve Counter	92A	–	92
4-bit Binary Counter	93A	–	93
Presettable Decade Counter/Latch	176	–	–
4-bit Binary Counter/Latch	177	–	–
Synchronous Decade Counter	160	–	160
Synchronous 4-bit Binary Counter	161	–	161
Fully Synchronous Decade Counter	162	–	162
Fully Synchronous 4-bit Binary Counter	163	–	163
Synchronous Decade Decimal Rate Multiplier	167	–	–
Synchronous Decade Up/Down Counter	190	–	190

(to be continued)

Function	HD74 Series	HD74S Series	HD74LS Series
Synchronous 4-bit Binary Up/Down Counter	191	–	191
Synchronous Decade Up/Down Counter	192	–	192
Synchronous 4-bit Binary Up/Down Counter	193	–	193
Decade Counter	290	–	290
4-bit Binary Counter	293	–	293
Dual 4-bit Decade Counters	–	–	390
Dual 4-bit Binary Counters	–	–	393
Dual 4-bit Decade Counters	–	–	490
Synchronous Decade Up/Down Counter	–	–	668
Synchronous 4-bit Binary Up/Down Counter	–	–	669

• 4-BIT, 5-BIT SHIFT/STORAGE REGISTERS

Function	HD74 Series	HD74S Series	HD74LS Series
4-bit Right-shift, Left-shift Register	95A	–	95B
5-bit Shift Register (Dual Parallel-in, Parallel-out)	96	–	–
4-bit D-type Register (with 3-state Outputs)	173	–	–
4-bit Parallel-in, Parallel-out Bidirectional Shift Register	194	–	194A
4-bit Parallel-in, Parallel-out Shift Register (J-K Inputs for First Stage)	195	–	195A

• 8-BIT SHIFT REGISTERS

Function	HD74 Series	HD74S Series	HD74LS Series
8-bit Shift Register	91A	–	91
8-bit Parallel-out Shift Register	164	–	164
Parallel-load 8-bit Shift Register	166	–	166
8-bit Parallel-in, Parallel-out Bidirectional Shift Register	198	–	–
8-bit Parallel-in, Parallel-out Shift Register (J-K Inputs for First Stage)	199	–	–
8-bit Universal Shift/Storage Register	–	–	299

• ENCODERS

Function	HD74 Series	HD74S Series	HD74LS Series
10-line-to-4-line Priority Encoder	147	–	–
8-line-to-3-line Priority Encoder	148	–	148

• DECODERS/DEMULTIPLEXERS

Function	HD74 Series	HD74S Series	HD74LS Series
BCD-to-Decimal Decoder	42A	–	42
Excess 3-to-Decimal Decoder	43A	–	–
Excess 3-Gray-to-Decimal Decoder	44A	–	–
3-to-8-line Decoder	–	–	138
Dual 2-to-4-line Decoders/Demultiplexers	–	–	139
4-line-to-16-line Decoder/Demultiplexer	154	–	154
Dual 2-line-to-4-line Decoders/Demultiplexers	155	–	155
Dual 2-line-to-4-line Decoders/Demultiplexers (with Open Collector Outputs)	156	–	156
4-line-to-16-line Decoder/Demultiplexer (with Open Collector Outputs)	159	–	–

• DECODERS/LAMP DRIVERS/BUFFERS

Function	HD74 Series	HD74S Series	HD74LS Series
BCD-to-Decimal Decoder/Driver (with 30V Outputs)	45	–	–
BCD-to-Decimal Decoder/Driver (with 15V Outputs)	145	–	145
BCD-to-Seven Segment Decoder/Driver (with 30V Outputs)	46A	–	–
BCD-to-Seven Segment Decoder/Driver (with 15V Outputs)	47A	–	47
BCD-to-Seven Segment Decoder	–	–	48
BCD-to-Seven Segment Decoder	–	–	49
BCD-to-Decimal Decoder/Driver (with 60V Out.)	141	–	–
BCD-to-Seven Segment Decoder/Driver (with 15V Outputs)	–	–	247
BCD-to-Seven Segment Decoder/Driver	–	–	248
BCD-to-Seven Segment Decoder/Driver	–	–	249

• LATCHES

Function	HD74 Series	HD74S Series	HD74LS Series
Quad. Bistable Latches	75	–	75
4-bit Bistable Latch	–	–	77
Quad. S-$\bar{R}$ Latches	279	–	279
8-bit Addressable Latch	–	–	259
Octal D-type Latches (with 3-state Out., Common Enable)	–	–	373
Octal D-type Latches (with 3-state Out., Common Clock)	–	–	374
4-bit Bistable Latch	–	–	375

• RANDOM ACCESS MEMORIES (less than 256-bit)

Function	HD74 Series	HD74S Series	HD74LS Series
64-bit Random Access Memory (16W by 4b)	89	–	–
4-by-4 Register Files (with Open Collector Outputs)	–	–	170
4-by-4 Register Files (with 3-state Outputs)	–	–	670

• ARITHMETIC ELEMENTS

Function	HD74 Series	HD74S Series	HD74LS Series
4-bit Binary Full Adder	83A	–	83A
4-bit Magnitude Comparator	85	–	85
Quad. 2-input Exclusive-OR Gates	86	86	86
Quad. Exclusive-OR/NOR Gates	–	135	–
Quad. 2-input Exclusive-OR Gates (with Open Collector Outputs)	136	–	136
8-bit Odd/Even Parity Generator/Checker	180	–	–
4-bit Arithmetic Logic Unit/Function Generator	–	181	181
Look-Ahead Carry Generator (for ALU)	182	182	–
Dual Carry Save Full Adders	H183	–	–
Quad. 2-input Exclusive-NOR Gates (with Open Collector Outputs)	–	–	266
9-bit Odd/Even Parity Generator/Checker	–	280	280
4-bit Binary Full Adder (with Fast Carry)	283	–	283
Quad. 2-input Exclusive-OR Gates	–	–	386

• DATA SELECTORS/MULTIPLEXERS

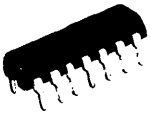
Function	HD74 Series	HD74S Series	HD74LS Series
16-bit Data Selector/Multiplexer	150	–	–
8-bit Data Selector/Multiplexer (with Strobe)	151A	151	151
8-bit Data Selector/Multiplexer	–	–	152
Dual 4-line-to-1-line Data Selectors/Multiplexers	153	–	153
Quad. 2-line-to-1-line Data Selectors/Multiplexers	157	157	157
Quad. 2-line-to-1-line Data Selectors/Multiplexers	–	158	158
8-bit Data Selector/Multiplexer (with Stobe and 3-state Outputs)	251	251	251
Dual 4-line-to-1-line Data Selectors/Multiplexers (with 3-state Outputs)	–	–	253
Quad. 2-line-to-1-line Data Selectors/Multiplexers (with 3-state Outputs)	–	257	257
Quad. 2-line-to-1-line Data Selectors/Multiplexers (with 3-state Outputs)	–	258	258
Quad. 2-input Multiplexers (with Storage)	–	–	298

• MICROPROCESSOR SUPPORT FUNCTIONS

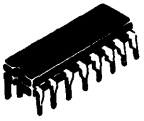
Function	HD74 Series	HD74S Series	HD74LS Series
Octal Buffers/Line Drivers/Line Receivers (Inverted 3-state Outputs)	–	–	240
Octal Buffers/Line Drivers/Line Receivers (Noninverted 3-state Outputs)	–	–	241
Quad. Bus Transceivers (Inverted 3-state Outputs)	–	–	242
Quad. Bus Transceivers (Noninverted 3-state Outputs)	–	–	243
Octal Buffers/Line Drivers/Line Receivers (Inverted 3-state Outputs)	–	–	244
Octal Bus Transceivers (Noninverted 3-state Outputs)	–	–	245
Octal Bus Transceivers (Inverted 3-satte Outputs)	–	–	640
Octal Bus Transceivers (Noninverted Open Collector Outputs)	–	–	641
Octal Bus Transceivers (Inverted Open Collector Outputs)	–	–	642
Octal Bus Transceivers (Noninverted 3-state Outputs)	–	–	645

■ OUTLINE

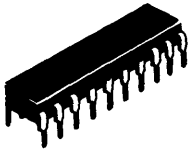
DP-14



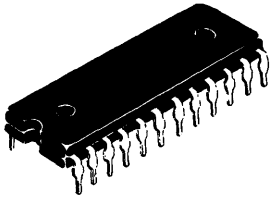
DP-16



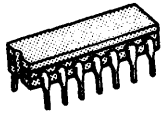
DP-20



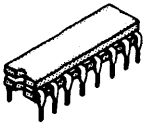
DP-24



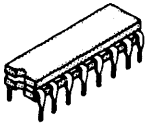
DG-14



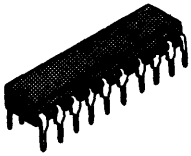
DG-16



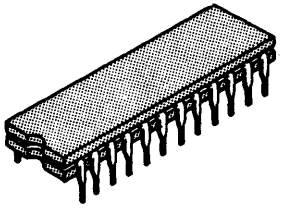
DG-16A



DG-20



DG-24



Advanced Low Power Schottky TTL HD74ALS Series

ALS Series attracting the users' attention as the standard logic have also added to our logic family. At present, only the following types are being introduced, but our company is making every effort to develop not only SSI but also MSI, to meet your needs.

■ **PERFORMANCE (per gate)**

Performance	LS Series	ALS Series
Propagation delay	10ns	4ns
Power dissipation	2mW	1mW
Speed-power product	20pJ	4pJ

■ **MAIN CHARACTERISTICS** (Ta = −20~+75°C)

Parameter	min	max
V_{OL} (I_{OL} = 8mA)	–	0.5V
V_{OH} (I_{OH} = −400μA)	2.7V	–
V_{IL}	–	0.8V
V_{IH}	2.0V	–
I_{IL}	–	−0.4mA
I_{IH} (V_{IH} min)	–	20μA

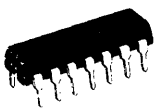
■ **LINE-UP**

Type No.	Function
HD74ALS00	Quad. 2-Input NAND Gates
HD74ALS01	Quad. 2-Input NAND Gates (o/c)
HD74ALS03	Quad. 2-Input NAND Gates (o/c)
HD74ALS04	Hex. Inverters
HD74ALS05	Hex. Inverters (o/c)
HD74ALS08	Quad. 2-Input AND Gates
HD74ALS09	Quad. 2-Input AND Gates (o/c)
HD74ALS20	Dual 4-Input NAND Gates
HD74ALS21	Dual 4-Input AND Gates
HD74ALS22	Dual 4-Input NAND Gates (o/c)
HD74ALS74	Dual D-Type Flip-Flops
HD74ALS109	Dual J-K Flip-Flops
HD74ALS112*	Dual J-K Flip-Flops
HD74ALS113*	Dual J-K Flip-Flops
HD74ALS114*	Dual J-K Flip-Flops
HD74ALS175*	Quad. D-Type Flip-Flops

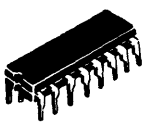
* Preliminaly

■ **OUTLINE**

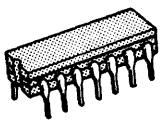
DP-14



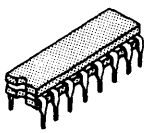
DP-16



DG-14



DG-16



CMOS Logic HD14000B/UB Series

■ FEATURES

- Low Current Drain .... 0.5nA typ./Package (V_{DD} = 5V)
- High Noise Margin 45% typ. of V_{DD}, 30% min. of V_{DD}
- Wide Supply Voltage Range ..... V_{DD} = 3~18V
- Wide Operating Temperature Range ..... −40~+85°C
- Capable of driving two low-power TTL loads, one low-power Schottky TTL load, or two HTL loads over the rated temperature range
- Industry-standardized (EIA/JEDEC) family specification
- Parameters specified at 5, 10, and 15V supply

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Rating	Unit
DC Supply Voltage	V_{DD}	−0.5~+18	V
Input Voltage (All inputs)	V_{in}	−0.5~V_{DD}+0.5	V
Output Voltage	V_{out}	−0.5~V_{DD}+0.5	V
Input Current (per Pin)	I_{in}	±10	mA
Operating Temperature	T_A	−40~+85	°C
Storage Temperature	T_{stg}	−65~+150	°C
Power Dissipation	P_D	300	mW

■ SELECTION GUIDE

• NAND Gates

Quad. 2-input NAND Gate	HD14011B
Quad. 2-input NAND Schmitt Trigger	HD14093B
Triple 3-input NAND Gate	HD14023B
Dual 4-input NAND Gate	HD14012B
8-input NAND Gate	HD14068B

• NOR Gates

Quad. 2-input NOR Gate	HD14001B
Triple 3-input NOR Gate	HD14025B
Dual 4-input NOR Gate	HD14002B
8-input NOR Gate	HD14078B

• AND Gates

Quad. 2-input AND Gate	HD14081B
Triple 3-input AND Gate	HD14073B
Dual 4-input AND Gate	HD14082B

• OR Gates

Quad. 2-input OR Gate	HD14071B
Triple 3-input OR Gate	HD14075B
Dual 4-input OR Gate	HD14072B

• Complex Gates

Quad. Exclusive-OR Gate	HD14070B
Quad. Exclusive-NOR Gate	HD14077B
Triple Gate (Dual 4-input NAND and 2-input NOR/OR or 8-input AND/NAND)	HD14501UB
Dual Expandable AND-OR-INVERT Gate	HD14506B
4-bit AND/OR Selector (Quad. 2 channel Data Selector or Quad. Exclusive-NOR Gate)	HD14519B
Dual 5-input Majority Logic Gate	HD14530B
Hex Gate (Quad. Inverter plus 2-input NOR plus 2-input NAND)	HD14572UB

• Inverters/Buffers/Level Translators

Dual Complementary Pair plus Inverter	HD14007UB
Hex Inverter/Buffer	HD14049UB
Hex Buffer	HD14050B
Hex Inverter	HD14069UB
Strobed Hex Inverter/Buffer	HD14502B
Hex 3-state Buffer	HD14503B
Hex Schmitt Trigger	HD14584B

• Decoders/Encoders

BCD-to-Decimal/Binary-to-Octal Decoder	HD14028B
4-bit Latch/4-to-16-line Decoder (high)	HD14514B
4-bit Latch/4-to-16-line Decoder (low)	HD14515B
8-bit Priority Encoder	HD14532B
Dual Binary-to-1-of-4 Decoder/Demultiplexer	HD14555B
Dual Binary-to-1-of-4 Decoder/Demultiplexer (Inverting)	HD14556B

• Display Decoders

BCD-to-Seven Segment Latch/Decoder/Driver	HD14511B
BCD-to-Seven Segment Latch/Decoder/Driver	HD14543B

• Multiplexers/Demultiplexers/Bilateral Switches

Quad. Analog Switch/Quad. Multiplexer	HD14016B
Quad. Analog Switch/Quad. Multiplexer	HD14066B
Triple 2-channel Analog Multiplexer/ Demultiplexer	HD14053B
Dual 4-channel Analog Multiplexer/ Demultiplexer	HD14052B
Dual 4-channel Analog Data Selector	HD14529B
Dual 4-channel Data Selector/Multiplexer	HD14539B
8-channel Analog Multiplexer/Demultiplexer	HD14051B
8-channel Data Selector	HD14512B
4-bit AND/OR Selector	HD14519B
BCD-to-Seven Segment Decoder	HD14558B

• Schmitt Triggers

Quad. 2-input NAND Schmitt Trigger	HD14093B
Dual Schmitt Trigger	HD14583B
Hex Schmitt Trigger	HD14584B

• Flip-Flops/Latches

Dual Type D Flip-Flop	HD14013B
Dual J-K Flip-Flop	HD14027B
Quad. Latch	HD14042B
Quad. NOR R-S Latch	HD14043B
Quad. NAND R-S Latch	HD14044B
Quad. D-Type Register	HD14076B
Quad. Type-D Flip-Flop	HD14175B
Dual 4-bit Latch	HD14508B
Hex Type-D Flip-Flop	HD14174B

(to be continued)

• Shift Registers	
4-bit Parallel-In, Parallel-Out Shift Register	HD14035B
4-bit Bidirectional Universal Shift Register	HD14194B
Dual 4-bit Static Shift Register	HD14015B
8-bit Static Shift Register	HD14014B
8-bit Static Shift Register	HD14021B
8-bit Universal Bus Register	HD14034B
18-bit Static Shift Register	HD14006B
1-of-64-bit Variable Length Shift Register	HD14557B
Dual 64-bit Static Shift Register	HD14517B
128-bit Static Shift Register	HD14562B

• Counters	
Seven-Stage Ripple Counter	HD14024B
Decade Counter/Divider	HD14017B
Presettable Divide-by-N Counter	HD14018B
Decade Counter (Asynchronous Clear)	HD14160B
Decade Counter (Synchronous Clear)	HD14162B
BCD Up/Down Counter	HD14510B
Programmable Divide-by-N 4-bit Counter (BCD)	HD14522B
12-bit Binary Counter	HD14040B
14-bit Binary Counter	HD14020B
Octal Counter/Divider	HD14022B
4-bit Binary Counter (Asynchronous Clear)	HD14161B
4-bit Binary Counter (Synchronous Clear)	HD14163B
Binary Up/Down Counter	HD14516B
Programmable Divide-by-N 4-bit Counter (Binary)	HD14526B
Dual BCD Up Counter	HD14518B
Dual Binary Up Counter	HD14520B
Dual Programmable BCD/Binary Counter	HD14569B
3-Digit BCD Counter	HD14553B
Real Time 5-Decade Counter	HD14534B
Industrial Time Base Generator	HD14566B

• Oscillators/Timers	
25-Stage Frequency Divider	HD14521B
Programmable Timer	HD14536B
Programmable Oscillator/Timer	HD14541B

• Phase-Locked Loops	
Phase-Locked Loop	HD14046B
Phase Comparator and Programmable Counter	HD14568B

• Multivibrators	
Dual Precision Retriggerable/Resettable Monostable Multivibrator	HD14538B

• Adders/Comparators	
4-bit Full Adder	HD14008B
Triple Serial Adder (Positive Logic)	HD14032B
Triple Serial Adder (Negative Logic)	HD14038B
NBCD Adder	HD14560B
9's Complementer	HD14561B
Look-Ahead Carry Block	HD14582B
4-bit Magnitude Comparator	HD14585B

• ALU Rate Multipliers	
BCD Rate Multiplier	HD14527B
2x2-bit Parallel Binary Multiplier	HD14554B
4-bit Arithmetic Logic Unit	HD14581B

• Parity Checkers	
12-bit Parity Tree	HD14531B

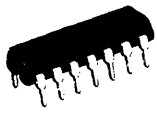
• Memories	
4x4 Multiport Register	HD14580B
64-bit Static Random Access Memory	HD14505B
256-bit Static Random Access Memory	HD14537B
256-bit Static Random Access Memory	HD14552B
1024-bit Read Only Memory	HD14524B

• A/D Converter/Logic Functions	
3½ Digit A/D Converter	HD14433B*
Microprocessor Based A/D Converter	HD14443B
Microprocessor Based A/D Converter	HD14447B
Successive Approximation Register	HD14549B
Successive Approximation Register	HD14559B

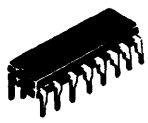
* Preliminary

■ OUTLINE

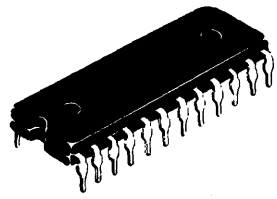
DP-14



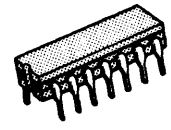
DP-16



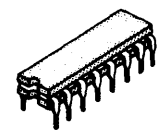
DP-24



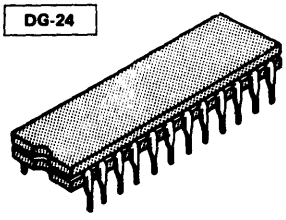
DG-14



DG-16



DG-24



Linear ICs

■ **Line Up**

Functions			Type No.	Package Code				Cross-reference
				P	PS	G	GS	
Operational Amplifiers	General Purpose		HA17741		DP-8	DG-14	DG-8	Fairchild μA741C
	High Speed		HA17715			DG-14		Fairchild μA715C
	Dual		HA17458		DP-8		DG-8	NS LM 1458
			HA17747	DP-14		DG-14		Fairchild μA747C
			HA17904		DP-8		DG-8	NS LM 2904
	Quad.		HA17301	DP-14		DG-14		Motorola MC3301
			HA17902	DP-14		DG-14		NS LM 2902
JFET Operational Amplifiers	Single		HA17080/A		DP-8		DG-8	Texas TL080/A
	Dual		HA17082/A		DP-8		DG-8	Texas TL082/A
			HA17083/A	DP-14		DG-14		Texas TL083/A
	Quad.		HA17084/A	DP-14		DG-14		Texas TL084/A
Voltage Comparators	Single		HA1813		DP-8			
	Universal		HA1812		DP-8		DG-8	
	Dual		HA17903		DP-8		DG-8	NS LM 2903
			HA1807			DG-14		
	Quad.		HA17901	DP-14		DG-14		NS LM 2901
Voltage Regulators	Variable	2~37V, 150mA	HA17723			DG-14		Fairchild μA723C
	Fixed	5V, 1A	HA17805	T-220AB				Fairchild μA7805C
		6V, 1A	HA17806	T-220AB				Fairchild μA7806C
		7V, 1A	HA17807	T-220AB				
		8V, 1A	HA17808	T-220AB				Fairchild μA7808C
		12V, 1A	HA17812	T-220AB				Fairchild μA7812C
		15V, 1A	HA17815	T-220AB				Fairchild μA7815C
		18V, 1A	HA17818	T-220AB				Fairchild μA7818C
		24V, 1A	HA17824	T-220AB				Fairchild μA7824C
		5V, 0.5A	HA178M 05	T-220AB				Fairchild μA78M05C
		6V, 0.5A	HA178M 06	T-220AB				Fairchild μA78M06C
		7V, 0.5A	HA178M 07	T-220AB				
		8V, 0.5A	HA178M 08	T-220AB				Fairchild μA78M08C
		12V, 0.5A	HA178M 12	T-220AB				Fairchild μA78M12C
		15V, 0.5A	HA178M 15	T-220AB				Fairchild μA78M15C
		18V, 0.5A	HA178M 18	T-220AB				Fairchild μA78M18C
		20V, 0.5A	HA178M 20	T-220AB				Fairchild μA78M20C
		24V, 0.5A	HA178M 24	T-220AB				Fairchild μA78M24C
	Switching Regulator Controller		HA17494	DP-16				Texas TL494
			HA17524	DP-16				Silicon General SG3524
A/D, D/A Converters	8-bit Double Integral Type A/D		HA16613A	DP-28				
	8-bit D/A		HA17008	DP-16				Analog Device DAC08
			HA17408	DP-16				AMD AM1408
	12-bit D/A		HA17012					AMD AM6012

(to be continued)

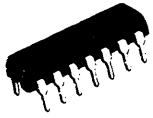
Linear ICs

Functions			Type No.	Package Code				Cross-reference
				P	PS	G	GS	
Other Function	Differential Video Amp.		HA17733			DG-14		Fairchild μA733C
	5-Transistor Arrays		HA1127			DG-14		RCA CA3045
	Precision Timers		HA17555		DP-8		DG-8	Signetics NE555
	Monostable Multivibrators		HA1607		DP-8			
	Micromotor Speed Controller		HA16503	DP-14				
	Light-measurement Amp. for Camera		HA16506	DP-14				
			HA16564	DP-14				
	Coin Sensor		HA16603	DP-16				
	Electric Leakage Breaker		HA16604	SP-8				
	Burner Controller		HA16605W	DP-20				
	8-channel Fluorescent Display Driver	Positive Supply	HA16617	DP-18				
		Negative Supply	HA16619	DP-18				

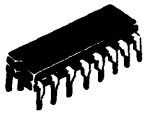
■ **OUTLINE**

DP-8

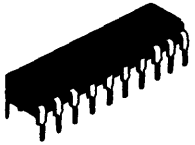
DP-14



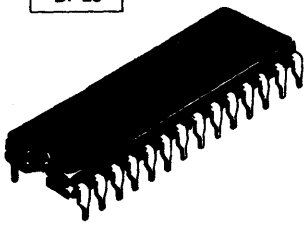
DP-16



DP-20

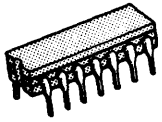


DP-28

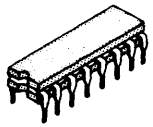


DG-8

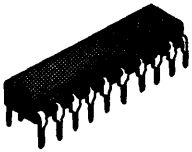
DG-14



DG-16

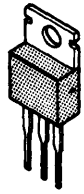


DG-20



SP-8

T-220AB



Interface Circuits

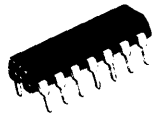
■ **Line Up**

Functions			Type	Package Code		Cross-Reference
				P	G	
Line Driver/ Receiver	Driver	Dual	HD75109	DP-14	DG-14	Texas SN75109
			HD75110	DP-14	DG-14	Texas SN75110
		Triple	HD2904		DG-16	
		Quad.	HD75188	DP-14	DG-14	Texas SN75188
	Receiver	Dual	HD75107A	DP-14	DG-14	Texas SN75107A
			HD75108A	DP-14	DG-14	Texas SN75108A
		Triple	HD2905		DG-16	
			HD2915		DG-16	
		Quad.	HD75154	DP-16	DG-16	Texas SN75154
			HD75189	DP-14	DG-14	Texas SN75189
Peripheral Driver	Dual NAND + NPN Transistor		HD75450A	DP-14	DG-14	Texas SN75450A
	Dual AND		HD75451A	DP-8	DG-8	Texas SN75451A
	Dual NAND		HD75452	DP-8	DG-8	Texas SN75452
	Dual OR		HD75453	DP-8	DG-8	Texas SN75453
	Dual NOR		HD75454	DP-8	DG-8	Texas SN75454
Memory Support	Core Memory	Dual Sense Amplifier	HD1902		DG-16	Texas SN7524
	IC Memory	Quad. TTL-MOS Clock Driver	HD2912		DG-16	
		Quad. TTL-MOS Clock Driver	HD2916		DG-16A	
		Quad. ECL-MOS Clock Driver	HD2922		DG-16	
		Quad. ECL-TTL Driver	HD2923		DG-16A	
Other	Printer Driver		HD2919	DP-16		

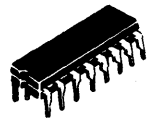
■ **OUTLINE**

DP-8

DP-14

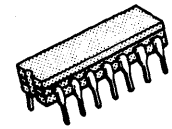


DP-16

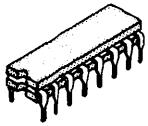


DG-8

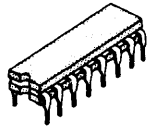
DG-14



DG-16



DG-16A



CROSS-REFERENCE

		Function	Hitachi	Motorola	Fujitsu	Fairchild	AMI	NEC	Intel	Toshiba	Mitsubishi	Matsushita	Others
8-bit Single-chip	MCU	2kB ROM 128B RAM	HD6801S0 HD6801S5	MC6801 MC6801-1	MB86801	F6801	S6801						Zilog; Z8
	MCU	4kB ROM 128B RAM	HD6801V0 HD6801V5					μPD7801G	8051				
	MPU	128B RAM	HD6803 HD6803-1	MC6803 MC6803-1	MB86803								
	MCU	1.1kB ROM 64B RAM	HD6805S1	MC6805P2	MB8883	F3870	S6805	μPD8048	8048	TMP8048	M5L8048		
		2kB ROM 96B RAM	HD6805U1	MC6805U2	MB8881	F3872		μPD8049	8049	TMP8049	M5L8049		Mostek; MK3870/12
		4kB ROM 96B RAM	HD6805V1										Mostek; MK3870/22
		4kB ROM 96B RAM	HD6805W0	MC6805R2				μPD8022	8022				Mostek; MK3870/42
		4kB ROM 96B RAM	HD6805X0	MC6805P2									
	MCU	4kB ROM 128B RAM	HD6301V0 HD63A01V0 HD63B01V0					μCOM87L					
	MPU	128B RAM	HD6303 HD63A03 HD63B03										
	MPU	128B RAM	HD6333 HD63A33 HD63B33										
	MCU	4kB ROM 96B RAM	HD63L05										
	MCU	EPROM+ 128B RAM	HD68P01S0										
			HD68P01V05 HD68P01V07										Mostek; MK38P73/02
	MCU	EPROM+ 96B RAM	HD68P05V05 HD68P05V07										Mostek; MK38P70/02
8-bit Multi-chip	MPU		HD6800 HD68A00 HD68B00	MC6800 MC68A00 MC68B00	MB8861	F6800	S6800	μPD8080A	8080A	TMP9080A	M5L8080A		
	MPU	Internal Clock & RAM	HD6802 HD6802W	MC6802	MB8870	F6802	S6802	μPD8085A	8054A	TMP8085A	M5L8085A	MN6802	
	MPU	High-End	HD6809 HD68A09 HD68B09 HD6809E HD68A09E HD68B09E	MC6809 MC68A09 MC68B09 MC6809E MC68A09E MC68B09E	MBL6809	F6809	S6809	μPD780					Zilog; Z80
	PIA		HD6821 HD68A21 HD68B21	MC6821 MC68A21 MC68B21	MB8874	F6821	S6821	μPD8255	8255	TMP9555P	M5L8255AP		
	PTM		HD6840 HD68A40 HD68B40	MC6840 MC68A40 MC68B40	MB8873			μPD8253	8253		M5L8253P		
	FDC		HD6843 HD68A43	MC6843	MB8876 MB8877	F6843			8271	T3444A	M5W1791-01P		WD;1771
	DMAC		HD6844 HD68A44 HD68B44	MC6844 MC68A44 MC68B44	M8865	F6844		μPD8257	8257	TMP9517C	M5L8257P		
	CRTC		HD6845S HD68A45S HD68B45S	MC6845★1 MC68A45★1 MC68B45★1		F6845		μPD3301	8275				
	COMBO		HD6846	MC6846	MB8872	F6846	S6846	μPD8155	8155	TMP8155	M5L8155P	MN6846	
	ACIA		HD6850 HD68A50	MC6850 MC68A50	MB8863	F6850	S6850	μPD8251	8251	TMP9551P	M5L8251AP		
	SSDA		HD6852 HD68A52	MC6852 MC68A52	MB8864	F6852	S6852	μPD8251	8251	TMP9551P	M5L8251AP		
	ADU		HD46508 HD46508-1 HD46508A HD46508A-1								M58990P		
	RTC		HD146818	MC146818									
16-bit Multi-chip	MPU		HD68000-4 HD68000-6 HD68000-8 HD68000-10	MC68000L4 MC68000L6 MC68000L8 MC68000L10									
	DMAC		HD68450	MC68450									

HITACHI LTD. SEMICONDUCTOR AND INTEGRATED CIRCUITS DIVISION SALES OFFICE

HEADQUARTERS

Hitachi, Ltd.
Nippon Bldg., 6-2, 2-chrome
Ohtemachi, Chiyoda-ku, Tokyo, 100, Japan
Tel: 270-2111
Telex: J22395, J22432
Fax: 03-246-0427

US SALES OFFICES

Hitachi America Ltd.
Electronic Devices
Sales and Service Division
1800 Bering Drive, San Jose, CA 95112
Tel: 408/292-6404
Telex: 17-1581
Twx: 910-338-2103
Fax: 408-2922133

EASTERN REGION
594 Marrett Rd., Suite 22
Lexington, MA 02173
617/861-1642

SOUTHERN REGION
6200 Savoy Dr., Suite 704
Houston, TX 77036
713-974-0534

NORTH CENTRAL REGION
500 Park Blvd., Suite 415
Itasca, IL 60143
312/773-4864

NORTHWEST REGION
1800 Bering Dr.
San Jose, CA 95112
408/292-6404

SOUTHWEST REGION
9700 Reseda Blvd., Suite 208
Northridge, CA 91324
213/701-6606

EUROPEAN SALES OFFICES (EU)

Hitachi Electronic Components Europe GmbH
Hans-Pinselstr. 3 8013 Haar bei München
Tel: 089/46140
Telex: 05-22593 (HITEC-D)
Fax: 089463151

Branch Offices:

Hitachi Electronic Components Europe GmbH
Veraufsbüro Düsseldorf
Königsallee 6, Düsseldolf 1
Tel: 0211-8905-0
Telex: 8584536 (HIEC-D)

Hitachi Electronic Components Europe GmbH
Verkaufsbüro Stuttgart
Fabrikstr 17 7024 Filerstadt 4
Tel: 0711/772011
Telex: 7255267 (HIES)

Hitachi Electronic Components Europe GmbH
Bureau de Representation en France
9, Boulevard Ney 75018 Paris
Tel: 201-/25-00
Telex: 211-835F

EUROPEAN SALES OFFICES (UK)

Hitachi Electronic Components (UK) Ltd.
221-225 Station Road
Harrow, Middlesex, HAL 2XL England
Tel: 01-861-1414
Telex: 936293 (HITEC-G)
Fax: 01-863-6646

Branch Office:
Hitachi Electronic Components (UK) Ltd.
Box 1062, 163 11 Spanga, Stockholm, Sweden
Tel: 08-751-0035
Telex: 14106

ASIAN SALES OFFICES

Hitachi Semiconductor (Hong Kong) Ltd.
Room 706 ~ 707, 7/F., Wing On Plaza,
Salisbury Road, Tsimshatsu, Kowloon, Hong Kong
Telephone: 3 ~ 7219218 ~ 9
Telex: 40815 HISAL HX
Fax: 852-0-251012

Branch Offices:
Hitachi Semiconductor (Hong Kong) Ltd.
Taiwan Branch Office
No. 73 7th F1-1, Fu Shing N. RD.
Hsing Nan Bldg. Taipei, Taiwan
Tel: 02-773-2162-3
Telex: 23222 HISEKTWN

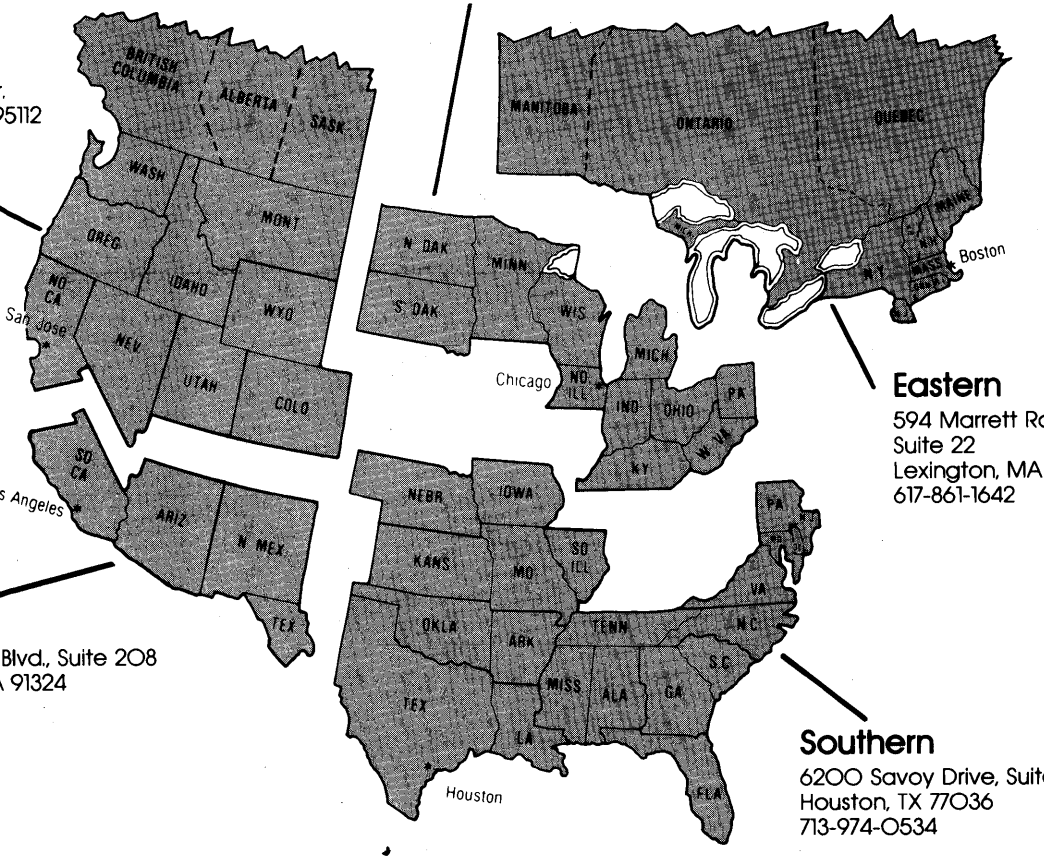
Hitachi Semiconductor (Hong Kong) Ltd.
Singapore Branch Office
60 Benoi Road, Singapore 2262
Tel: 268-9658
Telex: Rs33904 HEDS

REGIONAL OFFICES

For further information, contact your Regional Sales Office:

North Central
500 Park Blvd.
Suite 415
Itasca, IL 60143
312-773-4864

Northwest
1800 Bering Dr.
San Jose, CA 95112
408-292-6404



Eastern
594 Marrett Road
Suite 22
Lexington, MA 021
617-861-1642

Southwest
9700 Reseda Blvd., Suite 208
Northridge, CA 91324
213-701-6606

Southern
6200 Savoy Drive, Suite 70
Houston, TX 77036
713-974-0534

HITACHI

A World Leader in Technology

Hitachi America, Ltd., Semiconductor and IC Sales and Service Division
1800 Bering Drive, San Jose, CA 95112 408-292-6404

Printed in U.S.A.