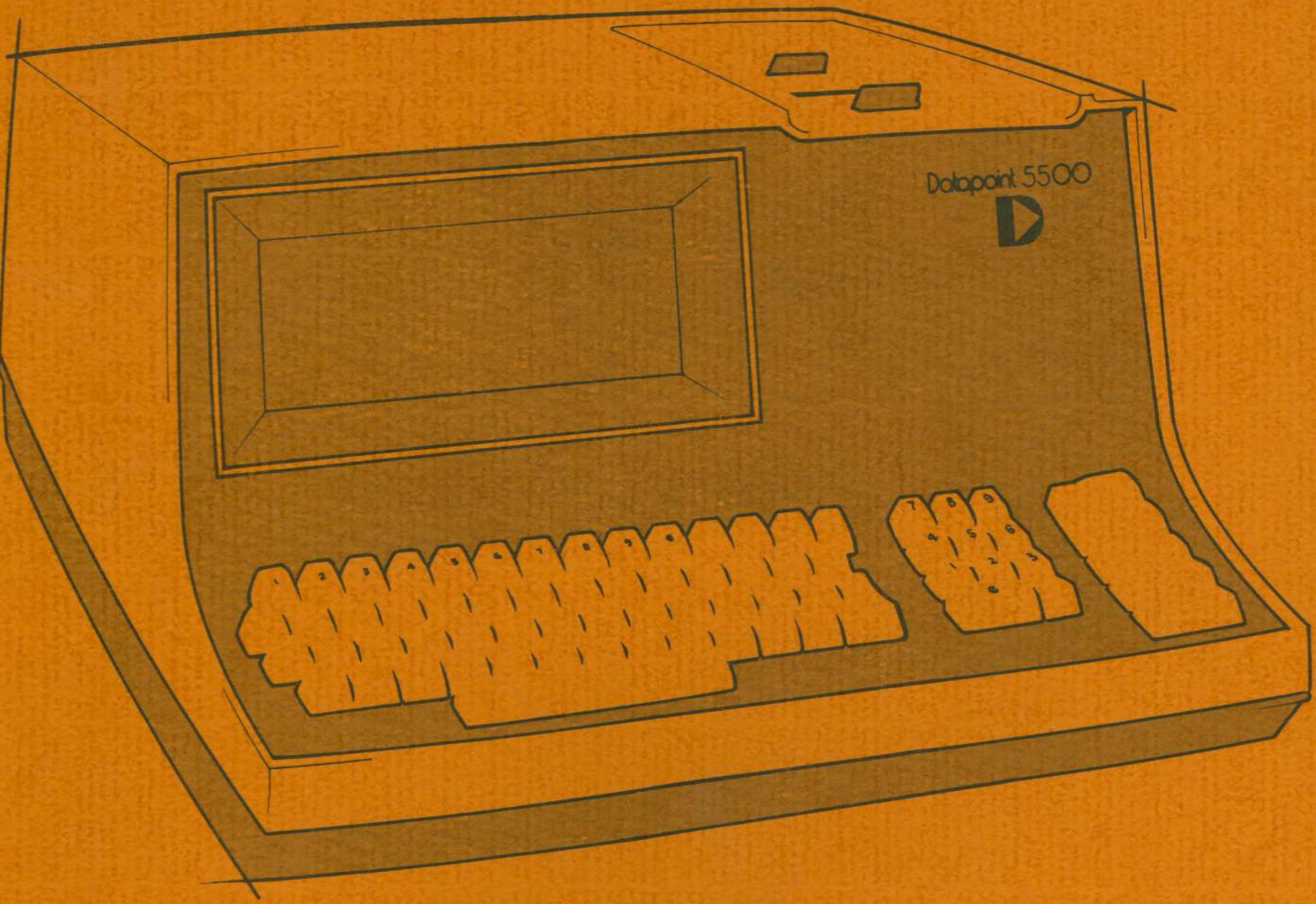
5501/5502 Reference Manual



DATAPOINT CORPORATION

The leader in dispersed data processing ™

5501/5502 Reference Manual

DATAPOINT CORPORATION

DATAPOINT CORPORATION

DATAPOINT 5501/5502

REFERENCE MANUAL

The computer-oriented user will find this manual useful for evaluation of the Datapoint 5500 system capabilities and limitations. However, only the hardware considerations are covered in this manual. The full utility of the Datapoint 5500 system cannot be appreciated until the available software support for the machine has been reviewed.

There is a complete family of software packages available for the Datapoint 5500 system including high-level languages, operating systems, source code and text editors, communications programs, utility programs, etc. Reference should be made to the latest issue of the Datapoint 5500 system Software Catalog for more complete information.

Table of Contents

1. SUMMARY

The Datapoint 5500 is a low cost versatile business oriented data processing system. The Models 5501 and 5502 processors provide the basic processor functions for this system and both include a keyboard, display, dual cassette decks, and 12,288 bytes of user program memory space in addition to the resident system memory. User program memory space may be expanded within the basic machine in increments of 12,288 bytes (with Model 5510 Memory Expansion Units) up to 49,152 bytes. The models 5501 and 5502 have identical specifications, except that the 5501 specifies nominally 115 v.a.c. power and the 5502 specifies nominally 230 v.a.c. power. Both units operate on 50 or 60 Hz power.

The instruction set for these processors contains all instructions used in the Datapoint 1100 and 2200 systems providing complete upward program and input-output compatability. In addition 5501/5502 processors provide:

- higher operating speed
- double precision arithmetic
- string arithmetic, moves, logic, etc.
- multiple-byte I/O transfers
- indexing and basing
- state saving and restoring instructions
- privileged instructions
- segmented and protected memory
- Memory and I/O parity
- additional registers

2. KEYBOARD

The integral keyboard provides a basic 55 key alphanumeric key group, an 11 key numeric group and five system control keys.

The keyboard provides a unique multi-key roll-over characteristic providing maximum ease of typing. Transfer of characters from the keyboard is under control of the processor. An audible click providing an acoustical feedback to the operator is available under processor control.

A programmable audio "beep" is also provided when it is desired to gain the operator's attention.

The 11-key matrix may be optionally supplied with control-key coding rather than numeric key coding and with keytops engraved to customer specifications.

The control keys are special function keys which exert control over the processor. Their names and associated functions are as follows:

RUN — Momentary contact switch, which when depressed, causes the processor to begin execution of the instruction located at the address in memory currently addressed by the program counter.

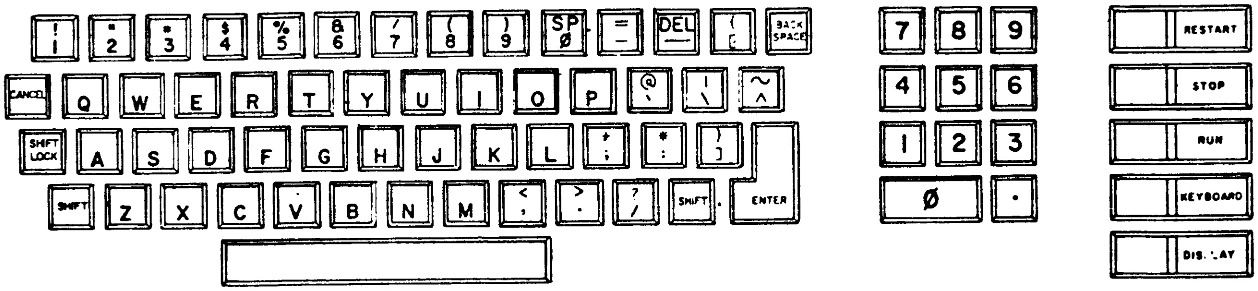
STOP — Momentary contact switch which, when depressed, causes instruction execution to halt at the completion of the current instruction. Care should be taken when using this switch, because any cassette tape operation which may be in progress will be aborted.

KEYBOARD — Momentary contact switch which sets a status bit that may be tested at any time by the processor.

DISPLAY — Momentary contact switch with a function similar to that of KEYBOARD switch. Either one or both of these switches may be depressed.

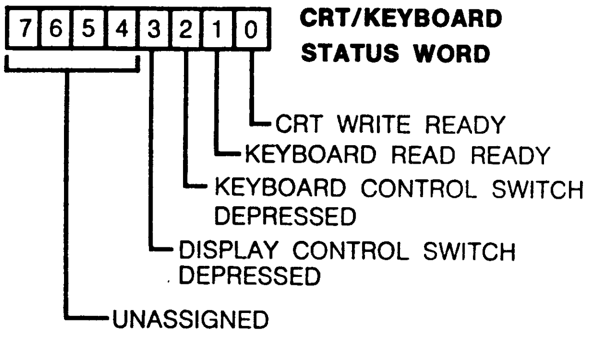
RESTART — Momentary contact switch which causes the processor to halt, rewind the system or program tape mounted on Deck 1 (rear deck), load and execute the first record found on tape. To protect against accidential restart, the RESTART function is inhibited unless the RUN key is depressed simultaneously.

KEYBOARD LAYOUT



2.1 KEYBOARD OPERATION

The keyboard is addressed by the processor by loading the A-register with 341_8 and executing an EX ADR command. (The CRT display also uses this address. Data transfers to the processor are from the keyboard and transfers from the processor are to the display.) Following the address sequence the c.r.t./keyboard status word can be loaded into the A-register by executing an INPUT instruction. Bit 1 of the A-register may be tested by the program to determine if a character is ready for transfer from the keyboard. Bits 2 and 3 will indicate if either the KEYBOARD or DISPLAY control switch is pressed.

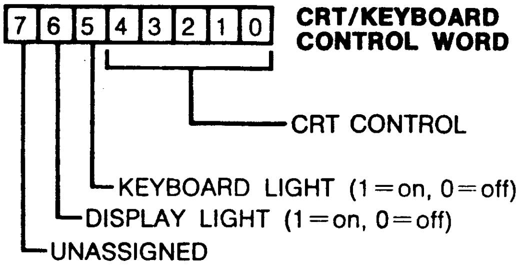


The External Commands associated with the operation of the keyboard are as follows:

a. EX BEEP. This command produces a 1500 Hertz tone for a duration of about 100 msec. The tone could be used as an error or ready signal to the keyboard operator.

b. EX CLICK. This command produces an audible click which could be used to acknowledge receipt of a valid character when a key is depressed.

c. EX COM1 (Command 1). Presents a control word contained in the A-register to the keyboard. Bit 5 of the control word controls the KEYBOARD switch light and bit 6 controls the DISPLAY switch light as follows:


7
6
5
4
3
2
1
0
CRT/KEYBOARD CONTROL WORD
CRT CONTROL
KEYBOARD LIGHT (1=on, 0=off)
DISPLAY LIGHT (1=on, 0=off)
UNASSIGNED

KEYBOARD CODING (ASCII)

A-101	0-060
B-102	1-061
C-103	2-062
D-104	3-063
E-105	4-064
F-106	5-065
G-107	6-066
H-110	7-067
I -111	8-070
J -112	9-071
	Space-040
K-113	
L-114	!-041
M-115	″-042
N-116	#-043
O-117	$-044
P-120	%-045
Q-121	&-046
R-122	′-047
S-123	(-050
T-124	)-051
U-125	*-052
V-126	+-053
W-127	,-054
X-130	- -055
Y-131	.-056
Z-132	/-057
a-141	:-072
b-142	;-073
c-143	<-074
d-144	=-075
e-145	>-076
f-146	?-077
g-147	[-133
h-150	~-176
i -151	]-135
j -152	∧-136
	— -137
k-153	
l -154	@-100
m-155	{-173
n-156	\-134
o-157	′-140
p-160	| -174
q-161	}-175
r-162	Enter-015
s-163	Cancel-030
t-164	Backspace-010
u-165	Del-177
v-166	
w-167	
x-170	
y-171	
z-172	

SPECIAL NUMBER PAD OPTION

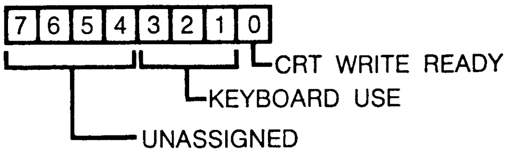
[.]-016
[0]-020
[1]-021
[2]-022
[3]-023
[4]-024
[5]-025
[6]-026
[7]-027
[8]-030
[9]-031

3. DISPLAY

The 5501/5502 display provides extended character generation flexibility and maximum character transfer rates. The display system includes: CRT Display of 12 lines of 80 characters, power line screen refresh rate, 960 cells of random access memory holding the screen image, a program loadable random access character generation memory capable of producing 128 individual 5 by 7 dot matrix characters, a group of registers utilized to position the cursor, and automatic cursor increment provisions. The maximum character transfer rate to the CRT is determined by processor input/output speed.

3.1 DISPLAY OPERATION

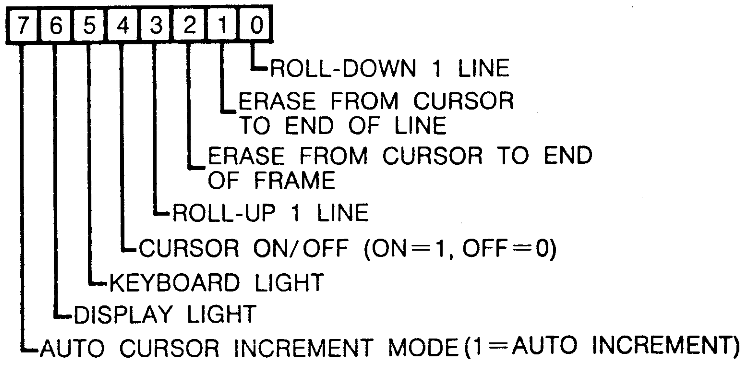
The CRT is addressed by the processor by loading the A-register with 341_8 and executing an EX ADR command. (Note that the keyboard also uses this address, see section 2 above.) Following the address sequence, the CRT/Keyboard status word can be loaded into the A-register by executing an INPUT instruction. The CRT status assignment is as follows:



Bit 0 of the status word indicates that the CRT is ready to accept data or commands if it is set to a logical 1. (Note that this status Bit will indicate a logical one if the cursor is positioned to an invalid screen position.) Bits 1, 2 and 3 are used for keyboard status.

Control of the CRT is accomplished through the use of the following external commands:

a) EX COM1 (Command 1) transfers a control word contained in the A-register to the CRT. The applicable bit assignments and their functions are as follows:


7 6 5 4 3 2 1 0
ROLL-DOWN 1 LINE
ERASE FROM CURSOR TO END OF LINE
ERASE FROM CURSOR TO END OF FRAME
ROLL-UP 1 LINE
CURSOR ON/OFF (ON=1, OFF=0)
KEYBOARD LIGHT
DISPLAY LIGHT
AUTO CURSOR INCREMENT MODE(1=AUTO INCREMENT)

A0: Each execution of EX COM1 with this bit set to 1 causes the roll-down operation to occur. All displayed characters (not the cursor) are moved down one line. The bottom line on the screen is lost and the top line is filled with the pattern in position 40 octal of the character generation memory. The Write Ready status bit goes false until the roll-down operation is complete; another EX COM1 must not be issued during this time.

A1: Each execution of EX COM1 with this bit set to 1 causes erasure from (including) the current cursor position to the end of the line. This function writes 40 octal into these locations of the screen image memory; the character displayed in the erased positions is determined by the pattern in position 40 octal of the character generation memory. The Write Ready status bit goes false until this operation is complete; another EX COM1 must not be issued during this time.

A2: Each execution of EX COM1 with this bit set to 1 causes erasure from (including) the current cursor position to the end of the frame. This function writes 40 octal into the screen image memory; the character displayed in the erased position is determined by the pattern in position 40 octal of the character generation memory. The Write Ready status bit goes false until this operation is complete; another EX COM1 must not be issued during this time.

A3: Each execution of EX COM1 with this bit set to 1 causes the roll-up operation to occur. All displayed characters (not the cursor) are moved up one line. The top line on the screen is lost and the bottom line is filled with the pattern in position 40 octal of the character generation memory. The Write Ready status bit goes false until the roll-up operation is complete; another EX COM1 must not be issued during this time.

A4: The cursor image may be turned on or off through the control word. The cursor position is the same in either case. The cursor image is automatically turned off whenever the processor is in the HALT state, and will be turned on again when RUN is depressed if the cursor was on prior to the HALT.

A5,A6: Keyboard & Display Light—(See section 2)

A7: When this bit is set to 1, the automatic cursor increment feature is in effect. In auto cursor increment mode, the cursor moves one character to the right after each EX WRITE command. The vertical position of the cursor does not change. If the last character (horizontal position 79) is written the cursor will increment off the screen and the CRT write ready status bit will stay true until the cursor is repositioned back onto the screen.

b) EX COM2 (Command 2) positions the cursor to the horizontal character slot designated by the contents of the A-register. Character positions 0-79 (decimal) or 0-117 octal are valid.

c) EX COM3 (Command 3) positions the cursor to the line designated by the contents of the A-register. Line numbers 0-11 (decimal) or 0-13 (octal) are valid.

d) EX COM4 (Command 4) places the character generator memory in the load mode and sets the load pointer to the contents of the A-register. Character positions 0-127 (decimal) or 0-177 (octal) are valid.

e) EX WRITE transfers the character in the A-register to the screen image memory at the position indicated by the cursor position. The cursor need not be on for this transfer to occur. If the auto-cursor increment feature is enabled the cursor position will be incremented after the transfer. When the character generation memory has been set to the load mode, the above transfer is inhibited (as is the automatic cursor increment) and EX WRITE transfers data from the A-register to the character generation memory. Execution of an EX WRITE (to either the screen image memory or the character generation memory) causes the Write Ready status bit to go false for up to 20 microseconds. Unless a delay of at least this duration is guaranteed by the program, the Write Ready status bit should be checked before execution of an EX WRITE, EX COM1, EX COM2, EX COM3 or EX COM4 after a previous EX WRITE.

Five successive byte transfers are required to load a complete 5 by 7 character dot pattern. The loading format is illustrated by the following diagram which illustrates the letter "A" loaded into memory:

A—REGISTER DATA

A6		x	x	x	
A5	x				x
A4	x				x
A3	x	x	x	x	x
A2	x				x
A1	x				x
A0	x				x
	1	2	3	4	5

TRANSFER
NUMBER
(EX WRITE)

For example, the procedure for loading the character location 101 with an "A" as illustrated would consist of the following character transfers:

```
        .
        .
        .
        LA      0101        Set load pointer to
        EX      COM4        Location 101
        LB      077
        CALL    DWRITE      Load column 1
        LB      0110
        CALL    DWRITE      Load column 2
        CALL    DWRITE      Load column 3
        CALL    DWRITE      Load column 4
        LB      077
        CALL    DWRITE      Load column 5
        .
        .
        .
```

The DWRITE subroutine below is used here instead of an EX WRITE instruction to guarantee the 17 microsecond delay required between executions of EX WRITE instructions:

```
            .
            .
DWRITE      LAB
            EX      WRITE
DWRITW      IN
            SRC
            JFC     DWRITW
            RET
```

After all five columns of a character have been loaded, the character load pointer is automatically incremented to the following character. In the case of the above example the load pointer will be incremented to location 102. Note that it is only necessary to issue additional COM4's when non-sequential character locations are being loaded. The display logic card is removed from the load mode by the execution of an EX COM1.

As mentioned previously, the Write Ready status bit goes false during the roll-up, roll-down, erase-to-end-of-line and erase-to-end-of-frame operations. The maximum periods during which Write Ready will be false for each of these operations is tabulated below for 60 Hz and 50 Hz primary power frequency:

OPERATION	50 HZ	60 HZ
Roll Up	21.1 msec	17.8 msec
Roll Down	21.1 msec	17.8 msec
Erase to End of Line	21.1 msec	17.8 msec
Erase to End of Frame	35 msec	31.7 msec

4. CASSETTE TAPES

The 5501/5502 processors contain two cassette tape recording devices for storage of programs and data. Since the hardware RESTART uses the rear deck (number one), programs will typically be on it while data areas will be the front deck (number two). However, once the machine is initially loaded, either deck may be used for both purposes.

Data on the Tape is organized by record (of any length). Records are written and read at 350 eight-bit characters per second with a tape speed of approximately 7.5 inches per second.

4.1 CASSETTE OPERATIONS

Data is recorded or read in bit serial fashion on one track. Each eight bit character is framed by three sync bits on either side of the character:

-	010	xxxxxxxx	010	xxxxxxxx	010	-
	SYNC CODE	CHARACTER 1	SYNC CODE	CHARACTER 2	SYNC CODE	

The appearance of the correct sync code indicates that the character is valid. Any other sync code causes special action to be taken on data reads. Note that the sync codes are valid for tape motion in either direction so the tape may be read backwards although in the reverse direction the data bits will appear reversed (bit 0 will be bit 7, 1 will be 6, etc.)

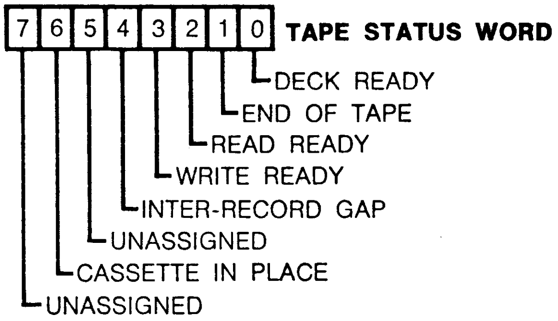
A record is a group of successive valid characters. An interrecord gap is indicated by the failure of the sync code to be zero one zero and the character position on all mark code (ones):

-	010	xxxxxxxx	010	xxxxxxxx	111	xxx	-
	SYNC CODE	VALID CHARACTER	SYNC CODE	INVALID CHARACTER	SYNC CODE	INTER RECORD GAP	

Only valid characters will be presented as data from the tape unit.

4.2 CASSETTE STATUS WORDS

The cassette tape unit is addressed by the processor by loading the A-register with 360_8 and executing the EX ADR instruction. Following this sequence, the tape unit status can be loaded into the A-register by executing an INPUT instruction. The bit assignments are as follows:



DECK READY — Deck ready will be set whenever the tape unit is ready to accept another command. (Only the TSTOP command should be issued if this bit is false). When Deck Ready is true the tape will be stopped, a cassette in the selected deck, and not wound to the clear leader at either end, and the head engaged. This bit should be checked after selecting a deck.

END OF TAPE — End of Tape indicates that the cassette has run onto leader (in either direction).

READ READY — Read Ready indicates that the selected deck has read another character.

WRITE READY — Write Ready indicates that the selected deck is ready to write another character.

INTER-RECORD GAP — Inter-Record Gap indicates the selected deck has come across an inter-record gap (invalid sync code).

CASSETTE IN PLACE — Cassette in Place indicates that a cassette is physically in place in the selected deck.

4.3 CASSETTE CONTROL

When the cassette tape unit is addressed the following instructions will control the action of the tape:

a. EX TSTOP causes any motion of either deck to be stopped and any read or write operations to be terminated. When everything has settled, the ready status bit will come true and operations may be resumed.

b. EX DECK1 causes deck one (rear) to be the currently selected deck. Before commanding a deck selection, care should be taken that the currently selected deck has completed all operations.

c. EX DECK2 causes deck two (front) to be the currently selected deck. Note the precaution in (b).

d. EX RBK causes the currently selected deck to be set in forward motion and, after 70 msec, for the read circuitry to be enabled. The read ready status bit will come true upon appearance on the tape of the first valid character. Upon appearance of an invalid sync code, the inter-record gap status bit comes true and tape motion is automatically stopped. Note that will happen only after at least one valid character has been found. Once the read ready status bit comes true, the character must be taken within 2.8 milliseconds or it will be overwritten with the next one. The tape read hardware double-buffers incoming characters to allow the 2.8 msec character availability.

e. EX BSP is similar to EX RBK except that tape motion is in the reverse direction so the data bits will be reversed.

f. EX SF is similar to EX RBK except the tape is not stopped upon appearance of an inter-record gap, and if allowed to continue will start to read the next record on the tape. In this case, the read ready status bit will come true again after the first character of the next record is read. Only an EX TSTOP will stop the motion initiated by EX SF.

g. EX SB is similar to EX SF except that tape motion is in the reverse direction and the data bits are reversed.

h. EX WBK causes the currently selected deck to be set in forward motion and all status bits except the write ready to go false. A character must then be presented within 2.8 milliseconds (the first character will be accepted at once due to the buffering in the tape hardware and then there will be a pause while the tape comes up to speed), at which time the write ready will go false until the writing circuitry is ready to accept another character. An end of record is signaled to the hardware by withholding a character for a period of time longer than 2.8 milliseconds specified above. When this is done, the write ready will go false, an interrecord gap will be written, the tape motion will cease and the deck ready status bit will come true again.

i. EX REWIND causes the tape to be rewound to the beginning on the selected deck. Worst case rewind time is approximately 40 seconds.

j. PUNCH TABS on the Cassette Cartridge are used for "write protect" and "automatic restart". The punch tab on the left (as you face the terminal) inhibits the ability to write on tape, when punched. When the tab on the right is punched on the rear deck, it causes an automatic restart whenever a programmed halt or power-up occurs. A manual halt (STOP key) will not initiate restart.

SUMMARY OF CASSETTE PHYSICAL SPECIFICATIONS

Density	47 characters/inch
Speed	7.5 ips
Recording Rate	350 c.p.s.
Capacity	115,000 characters (typical)
Start/Stop Time (Inter-Record Gap)	305 msec.
Start/Stop Distance (Inter-Record Gap)	2.2 inches
Rewind Speed	90 ips
Rewind Time (max 300 ft.)	40 sec.
Character Transfer Time	2.8 msec.

5. PROCESSOR

The processor in the 5501 or 5502 is comprised of two sets of eight 8-bit program accessible registers; and two sets of 4 control flags, up to 65,536 bytes of memory (49,152 of user program memory); a 16-bit program counter; an 8-bit instruction register; an 8-bit base register; a 16-level push-down stack; a special 4-bit instruction modification register; and a 16-word memory sector table.

5.1 PROCESSOR REGISTERS

The eight programmable registers are named A,B,C, D,E,H,L, and X. The flag flip-flops are named C (carry), Z (zero), S (sign), and P (parity). There are two sets of these registers and access to them depends upon the mode the processor is in. Upon restart or whenever the Alpha mode instruction is executed all Alpha mode registers and flags are accessible by the program. Whenever a Beta mode instruction is executed, the Beta mode registers and flags are accessible. No other registers or functions within the machine are affected by the processor mode.

The P register is the "location counter" for the program and contains the address of the next instruction to be executed. This register is stored in the pushdown stack upon the execution of a "CALL" instruction and is loaded with the effective address upon execution of a "JUMP", "CALL" or "RETURN" instruction. The P register is 16 bits in length and is capable of addressing up to 64K of memory.

The I register is the register which holds the "operation code" of the instruction currently being executed. The contents of I are gated through a decoding network to determine what operation interr.al or external, is to be performed. It is 8 bits long.

Additional special purpose registers are discussed below.

5.2 COMPARISON WITH DATAPOINT 2200 SYSTEM

5.2.1 Input/Output

Besides simply executing I/O instructions faster than the 2200 system (input instructions are twice as fast and output instructions are approximately the same speed).

the 5500 system I/O has parity checking while maintaining control over compatibility with 2200 devices.

5.2.2 Input parity checking

A ninth wire has been added to the input and output data paths of the I/O buses (there are several unused wires in the 2200 I/O cable). A second INPUT instruction (PIN) has been added which will cause an interrupt if there are not an odd number of ones out of the nine bits in the input bus when the data is strobed into the processor. Note that if a non-existent device is addressed and then a status check made, a parity fault will occur because the status will be all nine zeros, which is an even number (zero) of ones. Also note that using the old INPUT instruction will never cause a parity fault interrupt, allowing all 2200 programs to execute properly on the 5500 system (see Section 5.2.4).

5.2.3 Output parity checking

In addition to the output bus parity bit, there is another input wire to the processor called the Output Parity Fault Alert. If this wire is low during the parity fault check window (about 40 nanoseconds wide occuring 4 to 6 us after the trailing edge of any output strobe), the output parity fault interrupt will occur.
A 5500 system I/O device can check for an even number of ones out of the nine output bits at the leading edge of the output strobe. If there are an even number of ones, the device can hold the Output Parity Fault Alert line low until the leading edge of the next I/O strobe, thus causing the output parity fault interrupt.

5.2.4. Compatibility with 2200 system peripherals

5500 system peripherals may not be compatible with the 2200 because of the use of output parity checking. 2200 peripherals can be made to work on the 5500 system if the PIN (parity checking input) instruction is not used. The three additional wires used in the 5500 system I/O bus are currently not used in the 2200 system I/O.

5.3 MEMORY

In addition to having more memory capability than the 2200 system, the 5500 memory system features parity checking and advanced memory space handling.

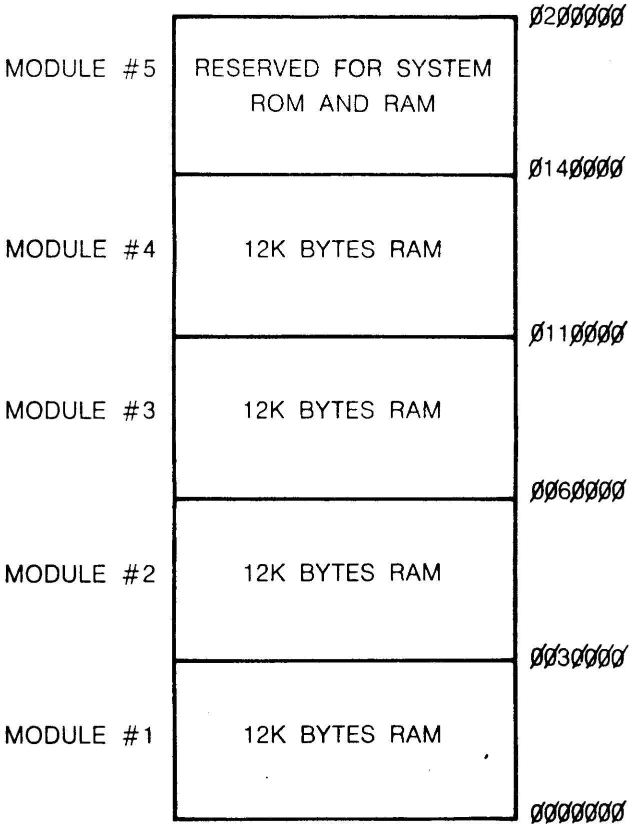
5.3.1 Parity checking

Each byte in the memory system has a ninth bit which is used for parity checking. Even parity is written in every location automatically when the machine is powered up and in the given location whenever a data byte is written (the words are written such that there are always an even number of ones out of the total number of nine bits). Whenever a data byte is read, a check for even parity is made and a special interrupt invoked if the check fails. This interrupt supplies the address of the failing memory location for diagnostic purposes. Note that if a non-existent memory location is accessed, a parity fault will not occur because all zeros (even number of ones) will be read. In addition to the RAM, the 5501/5502 contains a ROM (read-only memory) which is used for power initialization, RESTART, debugging, memory testing, and other system functions. The parity bit for ROM is generated artificially.

5.3.2 Physical layout

The 5501/5502 contains provisions for five memory boards. The first four boards contain 12K bytes of RAM each for a total user RAM capacity of 48K bytes. The fifth board is reserved for the ROM and RAM which is used exclusively for system control software.

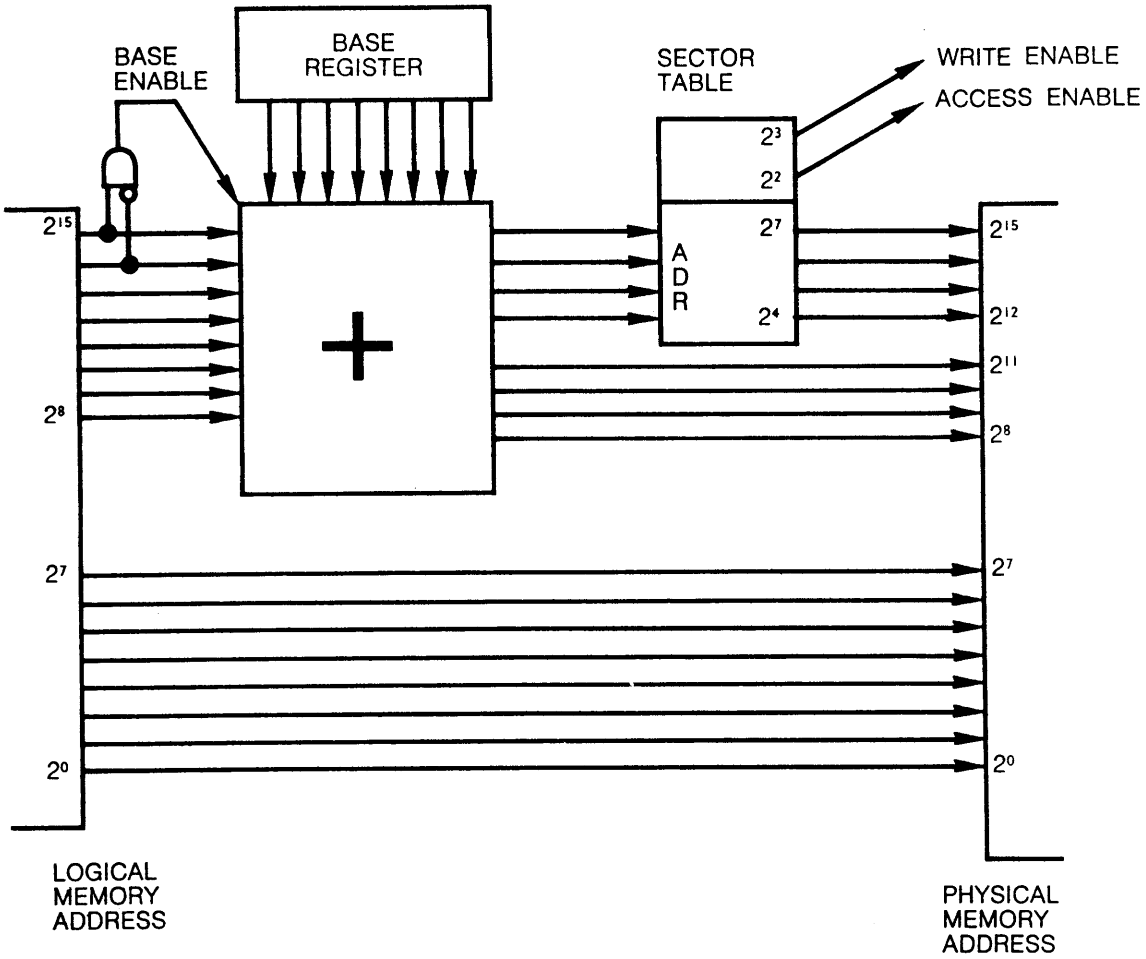
MEMORY LAYOUT:



5.3.3 Address generation

The above diagram is a picture of the physical memory layout. This memory is referenced by what is called a *physical memory address*. Board 1 is physical locations Ø through Ø27777 (octal), board 2 is Ø3ØØØØ through Ø57777, and so forth with the ROM and RAM residing in the area above Ø137777.

User programs use what is called a *logical memory address*. This is the 16-bit value created by the program and is translated to the proper *physical memory* address by a mechanism in the processor. The translation mechanism utilizes a base register and a memory sector table as depicted by the following diagram.



If the logical memory address is between Ø1ØØØØØ and Ø137777, its upper eight bits are added (two's complement) to the eight bit base register. Otherwise, the upper eight bits of the logical memory address are unchanged by the adder. The new 16-bit value consisting of the lower eight bits from the logical memory address and the eight bits from the adder is called the *based logical memory address.* Note that the base register may be negative (two's complement) for creating based logical memory addresses lower than Ø1ØØØØØ.

The upper four bits of the based logical memory address form an address for the 16-word 6-bit sector table. This table divides the 64K based logical memory space into sixteen 4K byte sectors, each of which may be translated to any physical 4K memory section and may be protected from being accessed if the USER mode flag is set or from being written into regardless of the state of the USER mode flag (these protections are indicated by two of the bits from the sector table). (Note that many people in the computer industry refer to the sector table as a page table. However, the reference has been changed here to avoid confusion with the term "page" used elsewhere to denote a 256 byte section of logical memory space starting at an address of 0 modulo 256.)

The mapping of based logical memory space to physical memory space is achieved by having four of the output bits from the sector table used for the upper four bits of the physical memory address and the lower twelve bits of the based logical memory address used for the lower twelve bits of the physical memory address. The 16th entry in the sector table is always set to point to the last 4K section of physical memory (0170000 through 0177777) with USER access not enabled to insure proper access to the information above 0167777 when the machine is initialized.

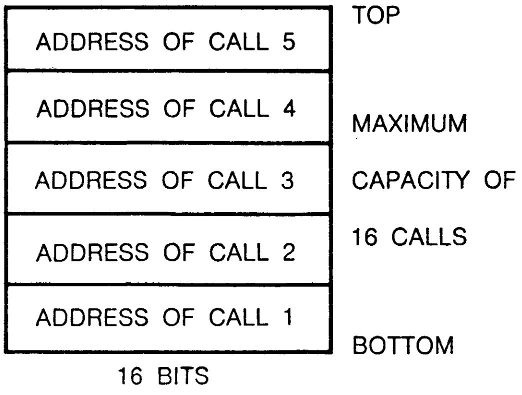
With the address generation mechanism described above, two major efficiencies can be realized. The first is ease of reentrant coding for multiple user tasks. The program can load into the base register the base address (in multiples of 256 bytes) of his non-reentrant data area minus Ø1ØØØØØ and then all logical memory references made between Ø1ØØØØØ and Ø1ØØØØØ plus the length of his data area will automatically be translated into the proper based logical memory location. The second major efficiency is afforded by the sector table. Besides providing the ability to implement a completely protected monitor, it provides ease in running several independent partitions in memory at once.

5.4 PUSHDOWN STACK

A feature of this machine is the incorporation into the processor's structure of a pushdown stack. This is useful for subroutine calling, saving the value of register pairs, calculating an address and then jumping to it without having to overstore a JUMP instruction, making an abortive exit from a subroutine (returning control to a location other than the one after the CALL instruction), and saving the state of the machine (if there is at least one free stack location).

Information may be transferred between either the P-counter and the stack or any register pair and the stack. The stack is actually a separate scratch pad memory of sixteen 16-bit words which is addressed by a four-bit up/down counter. Whenever a CALL or PUSH instruction is executed, the P-counter or indicated register pair is written into the stack word pointed out by the counter which is then incremented. The pointer end-arounds to 0 if it is incremented past 15. Whenever a RETURN or POP instruction is executed, the stack pointer is first decremented (ending around to 15 if it is decremented below 0) and then the P-counter or indicated register pair is loaded from the pointed location. Note that the above description implies that the maximum subroutine nesting depth is sixteen and will be less if data is also pushed onto the stack. That is, the seventeenth CALL or PUSH will overstore the value written in the first if no RETURN or POP instructions intervene.

PUSHDOWN STACK



5.5 CONTROL FLIP-FLOPS

Also contained in the basic processor is eight control (flag) flip-flops (four in Alpha mode and four in Beta mode) which reflect the state of the arithmetic logic unit and which can be tested through the execution of a conditional jump, call or return instruction. The flip-flop mnemonics with their associated functions are as follows:

C Carry Flip-flop. Set when an arithmetic operation results in either a carry (add) or borrow (subtract). *The Carry Flip-flop also reflects the state of the most significant bit in the accumulator after completion of a shift right instruction. Likewise, it reflects the state of the accumulator least significant bit after completion of a shift left instruction. Cleared after all logical operations.

Z Zero Flip-flop. Set when the result of an arithmetic or logical operation is equal to zero.*

S Sign Flip-flop. This flip-flop reflects the state of bit 7 in the accumulator after an arithmetic or logical operation.*

P Parity Flip-flop. Indicates the parity of the accumulator after any arithmetic or logical operation. This is entirely separate from the I/O or memory parity system referred to elsewhere. If this flip-flop is set (true), the accumulator contains an odd number of one bits; if it is reset (false), the accumulator contains an even number of one bits.*

*In the event of a compare instruction the contents of the accumulator are not changed; however, the control flip-flops reflect the equivalent of a subtract instruction.

5.6 SYSTEM ROM FUNCTIONS

The primary reason for using the ROM is simplification of the processor as it no longer must perform the logic of reading the bootstrap cassette from the rear deck. However, once a ROM is included, it is easy to implement even more sophisticated system control programs. The ROM contains four major routines that the user should be familiar with.

The first major ROM routine is executed whenever the 5501/5502 is supplied with power. This routine writes zeroes in all of memory to cause the proper setting of the parity bits and then halts the machine. Halting the processor causes a jump to the restart routine if the auto-restart tab is punched out on the cassette in the rear deck. Otherwise, the processor will stay halted until further operator action takes place.

The second major routine is the restart routine which is invoked by either the above action or the depression of the RESTART key. Actually, the process of manually restarting the machine has been changed from the 2200 system slightly due to human engineering considerations. On the 5501/5502, he RESTART key by itself has no effect. To RESTART the machine, operator must depress both the RUN and RESTART keys (thus the operator must also depress the RUN key (thus preventing accidental restarts because of depression of the wrong key). Also, auto-restart is suppressed if the STOP key is depressed (as opposed to a programmed halt or power-on halt), causing this key to bring the machine to quiescense regardless of the auto-restart tab in the rear deck.

The restart routine first checks to see if the KEYBOARD or DISPLAY key is depressed. If so, the debugging tool described in the following paragraph is entered. Otherwise, the routine initializes the sector table to make a one for one translation from based logical space to physical space and for no protection set (except for the 16th entry described above). Interrupts are disabled, the processor set to ALPHA mode, and the USER mode flag is cleared. The routine then waits for a cassette to be present in the rear deck. When it is present, it is rewound and the first block of information read into memory-starting at logical location zero (remember that at this point in time logical addresses are equivalent to physical addresses). After the first block has been loaded, execution is started at logical location zero.

The third major ROM routine is a debugging facility which can be operated without disturbing the user's program state. When RESTART is depressed with either the KEYBOARD or DISPLAY keys depressed, the entire state of the CPU (except for two stack locations) is saved and the debugging routine is entered. This routine is similar to software debugging tools released for the Datapoint 2200. It allows the operator to examine memory contents, CPU states, and I/O states from the keyboard and display replacing the crude method of most computers where information at such a level is usually accessed using console switches and indicator lights.

The debugging tool also allows the operator to initiate the fourth major ROM routine which is a generalized memory test. Having a memory test in ROM allows all RAM storage to be checked without relying upon the correct operation of some RAM storage.

5.7 INTERRUPT HANDLING

There are nine different interrupt events possible in the 5501/5502. All except the power-up interrupt use the System Call mechanism (see instruction description) to the memory location explained below. The System Call mechanism pushes the current value of the P-counter onto the stack, turns off USER mode, and forces execution to continue at the indicated vector location. Note that one of the interrupts is actually the System Call (SC) instruction and that the other interrupts use the same mechanism but jump to different locations.

The following describe the interrupt vector entry point locations. Note that all of these are into ROM locations.

Ø17ØØØØ MEMORY PARITY FAULT

This is caused by a memory read resulting in a nine bit word with an odd number of ones. Before the P-counter was pushed onto the stack by the System Call mechanism, the based logical memory address of the faulty memory cell was pushed onto the stack.

Ø17ØØØ3 INPUT PARITY FAULT

This is caused by a PIN instruction (see instruction explanation) resulting in a nine bit word from the I/O input bus with an odd number of ones. The PC valve pushed onto the stack points to the PIN instruction.

Ø17ØØØ6 OUTPUT PARITY FAULT

This is caused by the Output Parity Fault Line in the I/O bus being low during the parity fault check window (about 40 nanoseconds occurring 4 to 6 us after the trailing edge of any output strobe). The Output Parity Fault Line can be held low by 5500 System I/O devices if they see an odd number of ones out of the nine bits of the output bus.

System I/O devices if they see an odd number of ones out of the nine bits of the output bus. The PC valve pushed onto the stack points to the output instruction.

Ø17ØØ11 WRITE PROTECT VIOLATION

This is caused by a memory write operation being attempted on a sector of memory for which the write enable bit (A3 in the sector table entry) has not been set.

Systems software supplied with the 5500 system provides routines for handling each of these interrupts. Modification to these routines are possible by the user and systems software documentation should be consulted for further information.

Ø17ØØ14 ACCESS PROTECT VIOLATION

This is caused by the USER mode flag being set and a memory operation being performed on a sector of memory for which the access enable bit (A2 in the page sector table entry) has not been set.

Ø17ØØ17 PRIVILEGED INSTRUCTION VIOLATION

This is caused by the execution of an I/O instruction (except BEEP and CLICK) or an instruction capable of changing the sector table or base register while the USER mode flag is set. The PC valve pushed onto the stack points to the I/O instruction which caused the interrupt.

Ø17ØØ22 ONE MILLISECOND CLOCK

This is caused every 1000 microseconds. These interrupts can be inhibited with the DI instruction as in the 2200 system (and are inhibited with RESTART or POWER UP).

Ø17ØØ25 USER SYSTEM CALL

This is caused by the execution of an SC instruction.

Ø17ØØ3Ø BREAK POINT

This is caused by the execution of a BP instruction.

Ø17ØØ33 RESTART

This is caused by the depression of the RESTART key.

Ø17ØØ36 POWER UP

This is caused by the application of power to the 5501/5502. This interrupt is not invoked via the System Call mechanism. When power is appiled to the 5501/5502, the one millisecond clock interrupts are disabled, the USER mode flag is cleared, the ALPHA set of registers is selected, and the base register is set to zero.

5.8 PROCESSOR INSTRUCTIONS

The 5501/5502 processor instructions have been divided into six categories for convenience of presentation.

- Category one: All instructions contained in 1100 and 2200 system processors.
- Category two: 2200 system instructions which have been enhanced with additional register referencing capability.
- Category three: Multi-byte (string) instructions.
- Category four: Instructions for saving and restoring the state of the processor.

- Category five: Address manipulation instructions.
- Category six: Operating system control instructions.

5.8.1 Comparison to 2200 System Instructions

The 5501/5502 has a number of instructions not in 2200 system processors. Before these instructions can be described, however, the new data paths in the processor must be described. A new discrete register (not part of the register stack containing the general purpose registers) has been added. It is a working register called the *implicit register.*

Many 2200 instructions reference the A register implicitly (e.g., use it for an accumulator or load it from the I/O bus). The register that is implicitly referenced in the 5501/5502 in these cases is still the A register unless an instruction is executed which changes the implicitly referenced register for the following instruction *only*. There are seven instructions (one byte long) which allow the implicit register to be loaded with one through seven (implying registers B, C, D, E, H, L, or X). Once this is done, interrupts are inhibited until the following instruction is completed. If the following instruction would reference the A register implicitly in the 2200, the 5501/5502 will reference the register implied by the implicit register instead. Notice the use of the word "implied" as reference will be made to the implied register in later descriptions.

The instructions which set the implicit register will not be described separately since they are used only to augment the function code (opcode) of the instruction that follows. In some cases the value of the implicit register will not determine a register reference but will modify an operation action instead. The implicit register is al ;o used for a loop counter in many of the multi-byte ins:.uctions. Since the implicit register is only four bits wide, the multi-byte instructions that do use it for a loop counter are limited to executing the loop sixteen times (usually meaning that fields are limited to sixteen bytes in width). However, some of the multi-byte instructions use a general purpose register for the loop counter enabling them to loop 256 times. Note, however, that the one millisecond interrupt can occur only during the fetch of a new instruction if interrupts are enabled at all. This means that for some of the longer multi-byte instructions, interrupts can be disabled for as long as 850 microseconds. This would be troublesome if one was using the one millisecond clock for short-term time critical work. The full 256 byte capability was included, however, in the event that one might find it useful if time critical work was not being performed.

Two additional general purpose registers have been added to the 5501/5502 processors. By general purpose, it is meant that there is one for each mode (Alpha and Beta) and that they reside in the register stack along with the rest of the general purpose registers. These new registers actually exist in 2200 system processors (register 7), but are not useful in those machines because the only instruction which works is the load immediate! (A 076 <v> in the 2200 processor will load the register 7 with <v> instead of storing <v> into the memory location pointed to by HL). In the 5501/5502, this register 7 in the general purpose register stack is called the X register.

The X register is not quite as generally accessable as the rest of the registers, due to the fact that register number 7 is used to specify memory in many instructions. However, the X register can be loaded immediately as well as be accessed via the implicit register mechanism and also by several instructions which use the X register's contents as the upper eight bits of an address. The X register is generally used in the 5500 system to indicate a working storage page in memory. (Here the word "page" is used to denote a 256 byte section of logical memory space.) The use of the X register enables several of the instructions which provide a fixed memory address in the instruction to be one byte shorter by not having to specify the upper eight bits of the address (using the contents of the X register instead). Experience in programming the 2200 system has shown that one working storage page is generally quite adequate to hold most of the items accessed most often by a given program and that these items are accessed often enough to make the X register concept useful both in terms of saving memory and increasing speed.

Additional programming conventions developed with the 2200 system have been reflected in the new instruction set. The BC and DE registers are often used as pairs to form a sixteen bit value (B or D being the MSB and C or E being the LSB). Several of the new instructions treat these pairs specifically as sixteen bit values.

5.8.2 Presentation Format

A description of each 5501/5502 instruction is given below. In order to simplify the presentation, the following symbols and abbreviations are used:

A
B
C
D
E
H
L
X } 8 bit general purpose registers

M memory location designated by contents of the designated register pair

P Program counter

⟨op⟩ one of the eight ALU operations (AD AC SU SB ND XR OR CP)

⟨rs⟩ a source general register (ABCDEHLX) (s=0 to 7)

⟨rd⟩ a destination general register (ABCDEHLX) (d=0 to 7)

⟨r⟩ a general register (ABCDEHLX)

⟨rp⟩ one of the pairs of registers (BC DE HL)

⟨v⟩ an immediate value at P+1

⟨vv⟩ a two-byte immediate value at P+1, P+2

[r] a register select opcode. No byte is necessary for selection of the A register. Otherwise: B-111 C-062 D-113 E-174 H-115 L-176 X-117.

[rp] a register pair select opcode. No byte is necessary for selection of HL. Otherwise: BC-062 DE-174 XA-022.

⟨cf⟩ control flags (CZSP) (c = 0 to 3)

5.8.3 Category 1 – 2200 System Instructions

Load Immediate
Mnemonic: L ⟨rd⟩ ⟨v⟩ – example LB 3
Opcode: Od6
Flags: Unchanged
Timing: 2.0
Length: 2 bytes
Operation: Transfers the contents of the memory location immediately following the first byte of the instruction to the register specified by bits 3-5 of the first byte of the instruction.

Load
Mnemonic: L⟨rd⟩M, L⟨rd⟩⟨rs⟩, LM⟨rs⟩ – examples LBE, LMA
Opcode: 3ds (d or s=7 for M)
Flags: Unchanged
Timing: 1.2 (2.6 if LM⟨rs⟩)
Length: 1 byte
Operation: Transfers the operand from the source specified by bits 0-2 of the instruction word to the destination specified by bits 3-5 of the instruction word.

Add Immediate
Mnemonic: AD ⟨v⟩
Opcode: ØØ4
Flags: C,Z,S, and P indicate the status of the A register at completion.
Timing: 2.0
Length: 2 bytes
Operation: Adds to the contents of the A register the contents of the memory location immediately following the instruction, and retains the sum in the A register. Sets the C Flip-flop if overflow occurs, otherwise resets C.

Add
Mnemonic: AD⟨rs⟩, ADM
Opcode: 2Øs
Timing: 1.4 (2.6 if ADM)
Length: 1 byte
Operation: Identical with Add Immediate except that source is a register or separately addressed memory.

Add With Carry Immediate

Mnemonic: AC ⟨v⟩
Opcode: Ø14
Flags: C,Z,S and P indicate the status of the A register at completion.
Timing: 2.0
Length: 2 bytes
Operation: Adds the C bit and the contents of the location immediately following the instruction to the contents of the A register, and retains the sum in the A register. If overflow occurs, the C Flip-flop is set, otherwise C is reset.

Add With Carry
Mnemonic: AC⟨rs⟩, ACM
Opcode: 21s
Timing: 1.4 (2.6 if ACM)
Length: 1 byte
Operation: This instruction is identical to ADD WITH CARRY IMMEDIATE with the exception of operand source.

Subtract Immediate
Mnemonic: SU ⟨v⟩
Opcode: Ø24
Flags: C,Z,S and P indicate the status of the A register at completion.
Timing: 2.0
Length: 2 bytes
Operation: Subtracts the contents of the memory location immediately following the instruction from the contents of the A register, and retains the difference in the A register. The C Flip-flop is set if underflow occurs.

Subtract
Mnemonic: SU⟨rs⟩, SUM
Opcode: 22s

Timing: 1.4 (2.6 if SUM)
Length: 1 byte
Operation: This instruction is identical to Subtract Immediate with the exception of operand source.

Subtract With Borrow Immediate
Mnemonic: SB ⟨v⟩
Opcode: Ø34
Flags: C,Z,S and P indicate the status of the A register at completion.
Timing: 2.0
Length: 2 bytes
Operation: Subtracts the contents of the memory location immediately following the instruction and the C bit, from the contents of the A register. Sets the C bit if underflow occurs, otherwise resets C.

Subtract With Borrow
Mnemonic: SB ⟨rs⟩, SBM
Opcode: 23s
Timing: 1.4 (2.6 if SBM)
Length: 1 byte
Operation: This instruction is identical to Subtract With Borrow Immediate with the exception of operand source.

AND Immediate
Mnemonic: ND ⟨v⟩
Opcode: Ø44
Flags: Z,S, and P will indicate the status of the A register on completion. The C flag will be reset.
Timing: 2.0
Length: 2 bytes
Operation: Forms the logical product of the contents of the A register with the contents of the memory location immediately following the instruction, and places the results in the A register.

AND
Mnemonic: ND ⟨rs⟩, NDM
Opcode: 24s
Timing: 1.4 (2.6 if NDM)
Length: 1 byte
Operation: This instruction is identical to And Immediate with the exception of operand source.

OR Immediate
Mnemonic: OR ⟨v⟩
Opcode: Ø64
Flags: Z,S and P will indicate the status of the A register on completion. The C flag will be reset.
Timing: 2.0
Length: 2 bytes
Operation: Forms the logical sum of the contents of the A register and the contents of the memory location immediately following the instruction, and places the result in the A register.

OR
Mnemonic: OR ⟨rs⟩, ORM
Opcode: 26s
Timing: 1.4 (2.6 if ORM)
Length: 1 byte
Operation: This instruction is identical to OR Immediate with the exception of operand source.

Exclusive-OR Immediate
Mnemonic: XR ⟨v⟩
Opcode: Ø54
Flags: Z,S and P will indicate the status of the A register on completion. The C flag will be reset.
Timing: 2.0
Length: 2 bytes
Operation: The logical difference of the contents of the A register and the contents of the memory location immediately following the instruction is formed, and the result replaces the contents of the A register.

Exclusive-OR
Mnemonic: XR ⟨rs⟩, XRM
Opcode: 25s
Timing: 1.4 (2.6 if XRM)
Length: 1 byte
Operation: This instruction is identical to Exclusive-OR Immediate with the exception of operand source.

Compare Immediate
Mnemonic: CP ⟨v⟩
Opcode: Ø74
Flags: Same as SU instruction
Timing: 1.8
Length: 2 bytes
Operation: Compares the contents of the A register with the contents of the memory location immediately following the instruction. The flag flip-flops assume the same state as they would for a Subtract instruction.

Compare
Mnemonic: CP ⟨rs⟩, CPM
Opcode: 27s
Timing: 1.2 (2.4 if CPM)
Length: 1 byte
Operation: This instruction is identical to Compare Immediate with the exception of operand source.

Unconditional Jump
Mnemonic: JMP
Opcode: 1Ø4
Flags: Unchanged
Timing: 2.8
Length: 3 bytes
Operation: An unconditional transfer of control. The contents of P+1 represent the least significant portion of the new address, while the contents of P+2 represent the most significant portion.

Jump If Condition True

Mnemonic: JT⟨cf⟩
Opcode: 1⟨c+4⟩Ø
Flags: Unchanged
Timing: 3.0 (1.4 if no jump)
Length: 3 bytes
Operation: Examines the designated flip-flop. If set, transfers control to the address designated by the contents of the two memory locations immediately following the instruction. If the selected flip-flop is reset, executes the next sequentially available instruction.

Jump If Condition False

Mnemonic: JF⟨cf⟩
Opcode: 1⟨c⟩Ø
Flags: Unchanged
Timing: 3.0 (1.4 if no jump)
Length: 3 bytes
Operation: Examines the designated flip-flop. If reset, transfers control to the new address designated by the contents of the two memory locations immediately following the instruction. If the selected flip-flop is set, executes the next sequentially available instruction.

Subroutine Call

Mnemonic: CALL
Opcode: 1Ø6
Flags: Unchanged
Timing: 3.0
Length: 3 bytes
Operation: Transfers the address of the next sequentially available instruction to the Pushdown Stack, and transfers control to the new address specified by the contents of the two memory locations immediately following the Opcode.

Subroutine Call If Condition True

Mnemonic: CT⟨cf⟩
Opcode: 1⟨c+4⟩2
Flags: Unchanged
Timing: 3.2 (1.6 if no call)
Length: 3 bytes
Operation: Examines the designated flip-flop. If set, transfers the address of the next sequentially available instruction to the pushdown stack, and transfers control to the new address of the two memory locations immediately following the Opcode. If the selected flip-flop is reset, executes the next sequentially available instruction.

Subroutine Call If Condition False

Mnemonic: CF⟨cf⟩
Opcode: 1⟨c⟩2
Flags: Unchanged
Timing: 3.2 (1.6 if no call)
Length: 3 bytes
Operation: Examines the designated flip-flop. If reset, transfers the address of the next sequentially available instruction to the pushdown stack, and transfers control to the new address of the two memory locations immediately following the Opcode If the selected flip-flop is set, executes the next sequentially available instruction.

Subroutine Return

Mnemonic: RET
Opcode: ØØ7
Flags: Unchanged
Timing: 1.8
Length: 1 byte
Operation: Transfer control to the address specified by the most recent entry in the Pushdown Stack and deletes the most recent entry from the Stack.

Subroutine Return If Condition True

Mnemonic: RT⟨cf⟩
Opcode: Ø⟨c+4⟩3
Flags: Unchanged
Timing: 1.8 (1.0 if no return)
Length: 1 byte
Operation: Examines the designated flip-flop. If set, transfers control to the address specified by the most recent entry in the pushdown stack. Deletes the most recent entry in the stack. If the selected flip-flop is reset, executes the next sequentially available instruction.

Subroutine Return If Condition False

Mnemonic: RF⟨cf⟩
Opcode: Ø⟨c⟩3
Flags: Unchanged
Timing: 1.8 (1.0 if no return)
Length: 1 byte
Operation: Examines the designated flip-flop. If reset, transfers control to the address specified by the most recent entry in the stack and deletes that entry. If the selected flip-flop is set, executes the next sequentially available instruction

Shift Right Circular

Mnemonic: SRC
Opcode: Ø12
Flags: Z,S, and P unchanged. The C flag contains the most significant bit of the A register after the operation is complete.
Timing: 1.4
Length: 1 byte
Operation: Shifts the contents of the A register right in a circular fashion. Shifts the least significant bit into the most significant bit position. Upon completion of the operation, the Carry Flip-flop is equal to the most significant bit.

Shift Left Circular

Mnemonic: SLC
Opcode: ØØ2
Flags: Z,S, and P unchanged. The C flag contains the least significant bit of the A register after the operation is complete.
Timing: 1.4
Length: 1 byte

Operation: Shifts the contents of the A register left in a circular fashion. Shifts the most significant bit into the least significant bit position. Upon completion of the operation, the Carry Flip-flop is equal to the least significant bit.

Halt
Mnemonic: HALT
Opcodes: ØØØ, ØØ1, 377
Flags: Unchanged
Timing: Indeterminate
Length: 1 byte
Operation: The computer halts. When the RUN button on the console is depressed, operation resumes at P+1. This instruction causes a privileged instruction interrupt if the USR mode flag is set.

Input
Mnemonic: INPUT
Opcode: 1Ø1
Flags: Unchanged
Timing: 5.0
Length: 1 byte
Operation: Transfers the contents of the I/O Bus to the A register. This instruction causes a privileged instruction interrupt if the USER mode flag is set.

Pop
Mnemonic: POP
Opcode: Ø6Ø
Flags: Unchanged
Timing: 2.2
Length: 1 byte
Operation: Transfers the most recent entry of the stack into the H&L registers. H = MSP, L = LSP and deletes the most recent entry in the stack.

Push
Mnemonic: PUSH
Opcode: Ø7Ø
Flags: Unchanged
Timing: 2.0
Length: 1 byte
Operation: Transfers the contents of the H&L registers onto the PUSHDOWN stack. H = MSP, L = LSP.

Enable Interrupts
Mnemonic: EI
Opcode: Ø5Ø
Flags: Unchanged
Timing: 1.0
Length: 1 byte
Operation: Following the next instruction, will allow the one millisecond clock interrupts to occur until a Disable Interrupt instruction is executed.

Disable Interrupts
Mnemonic: DI
Opcode: Ø4Ø
Flags: Unchanged
Timing: 1.2
Length: 1 byte
Operation: Prevents the one millisecond clock interrupts from occurring until an Enable Interrupt instruction is executed. This instruction causes a privileged instruction interrupt if the USER mode flag is set.

Select Alpha Mode
Mnemonic: ALPHA
Opcode: Ø3Ø
Timing: 1.0
Length: 1 byte
Operation: Selects the Alpha Mode registers and control Flip-flops.

Select Beta Mode
Mnemonic: BETA
Opcode: Ø2Ø
Timing: 1.2
Length: 1 byte
Operation: Selects the Beta Mode registers and control Flip-flops. This instruction causes a privileged instruction interrupt if the USER mode flag is set.

External Commands
Mnemonic: EX ⟨exp⟩ (See Table)
Opcode: (See Table)
Flags: Unchanged
Timing: 9.2
Length: 1 byte
Operation: These instructions perform the functions necessary for control of the I/O system and external devices. Many of these functions are specifically related to operation of particular devices. The device oriented commands for the Keyboard, CRT Display, Cassette Tapes, and Communications Interface are explained in the sections covering these devices. All of these instructions (except BEEP and CLICK) will cause a privileged instruction interrupt if the USER mode flag is set.

EXTERNAL COMMANDS

EX (exp)

COMMAND NUMBER	(exp)	OCTAL CODE	COMMAND	DESCRIPTION	DEVICE ADDRESS
1	ADR	121	Address	Selects device specified by A-register	ALL
2	STATUS	123	Sense Status	Connects selected device status to input lines	
3	DATA	125	Sense Data	Connects selected device data to input lines	
4	WRITE	127	Write Strobe	Signals selected device that output data word is on output lines	
5	COM1	131	Command 1	Outputs a control function to selected device	
6	COM2	133	Command 2	Outputs a control function to selected device	
7	COM3	135	Command 3	Outputs a control function to selected device	
8	COM4	137	Command 4	Outputs a control function to selected device	ALL
9	– –	141	(Unassigned)	– –	–
10	– –	143	(Unassigned)	– –	–
11	– –	145	(Unassigned)	– –	–
12	– –	147	(Unassigned)	– –	–
13	BEEP	151	Beep	Activates tone producing mechanism	ANY
14	CLICK	153	Click	Activates audible click producing mechanism	ANY
15	DECK1	155	Select Deck 1	Connects deck 1 to I/O bus	360
16	DECK2	157	Select Deck 2	Connects deck 2 to I/O bus	
17	RBK	161	Read Block	Enables read circuitry and sets tape in forward motion	
18	WBK	163	Write Block	Enables write circuitry and sets tape in forward motion	360
19	– –	165	(Unassigned)	– –	–
20	BSP	167	Backspace One Block	Backs up the selected tape one record	360
21	SF	171	Slew Forward	Sets selected tape deck in forward motion	
22	SB	173	Slew Backward	Sets selected tape deck in backward motion	
23	REWIND	175	Rewind	Rewinds the selected deck to beginning of tape	
24	TSTOP	177	Stop Tape	Halts motion of the selected tape deck	360

5.8.4 Category 2 – Augmented Category 1 Instructions

L⟨rd⟩M using BC, DE, or XA for the address

Mnemonic: L⟨rd⟩M⟨rp⟩ – example: LEM BC uses BC

Opcode: [rp|3d7

Timing: 3 4

Length: 2 bytes

Operation: Identical to the L⟨rd⟩M instruction except that the specified register pair instead of HL is used for the memory address.

LM⟨rs⟩ using BC, DE, or XA for the address

Mnemonic: LM⟨rs⟩⟨rp⟩ – example: LMB DE uses DE

Opcode: [rp|37s

Timing: 3.4

Length: 2 bytes

Operation: Identical to the LM⟨rd⟩ instruction except that the specified register pair instead of HL is used for the memory address.

Arithmetic and logical operations to other than the A register

Mnemonic: ⟨op⟩⟨rs⟩⟨rd⟩ – examples:
ADAB adds A to B
ADMC adds (HL) to C
⟨op⟩ ⟨v⟩, ⟨rd⟩ – example: ADC 2ØC adds 20 to C
SRC⟨r⟩ – example: SRCB shifts B right
SLC⟨r⟩ – example: SLCD shifts D left

Opcodes: [r] 2ps
[r] Øp4 vvv
[r] Ø12
[r] ØØ2

Timing: Add 1.0 to equivalent category 1 instruction timing

Length: 2 or 3 bytes

Operation: Identical to the category 1 arithmetic operations except that the specified register instead of the A register is used for the accumulator.

Shift Right Extended

Mnemonic: SRE or SRE⟨r⟩ – example SREB
Opcode: Ø32 or [r] Ø32
Flags: Z,S, and P unchanged. C as described below.
Timing: 1.4 (2.4 if to other than A)
Length: 1 or 2 bytes
Operation: The accumulator is shifted right one place with the left hand bit being replaced by the Carry and The Carry being replaced by the right hand bit.

I/O using other than the A register

Mnemonic: IN⟨rd⟩ – example: INB loads into B
EX⟨rs⟩ ⟨exp⟩ – example: EXC COM1 outputs C
Opcodes: [r] 1Ø1
[r] 121, [r] 123, etc. (I/O strobes)
Timing: Add 1.0 to equivalent category 1 instruction timing
Length: 2 or 3 bytes
Operation: Identical to the 22ØØ I/O operations except the specified register instead of the A register is used.

Parity checking input

Mnemonic: PIN or PIN⟨rd⟩
Opcode: 1Ø3 or [r] 1Ø3
Flags: Unchanged
Timing: 5.4
Length: 1 or 2 bytes
Operation: Identical to the INPUT instruction except that if the nine bits of the 55ØØ system input bus contain an odd number of ones, an interrupt will occur.

PUSH using BC, DE, or XA

Mnemonic: PUSH⟨rp⟩ – example: PUSH DE
Opcode: [rp] Ø7Ø
Timing: 2.8
Length: 2 bytes
Operation: Pushes the specified register pair onto the stack.

PUSH Immediate

Mnemonic: PUSH ⟨vv⟩
Opcode: Ø51
Timing: 2.8
Length: 3 bytes
Operation: Pushes the contents of the operand onto the stack.

POP using BC, DE, or XA

Mnemonic: POP⟨rp⟩ – example: POP BC
Opcode: [rp] Ø6Ø
Timing: 3.Ø
Length: 2 bytes
Operation: Pops the stack into the specified register pair.

5.8.5 Category 3 – Multi-byte (string) operations

Block Transfer or Block Transfer Reverse

Mnemonic: BT or BTR
Opcode: Ø21 or 111 Ø21
Length: 1 or 2 bytes
Operation: The block transfer instructions move the number of bytes specified in the C register from the field pointed to by HL to the field pointed to by DE while adding the contents of the A register to each byte transferred. BT causes the pointers to be incremented after each transfer while BTR causes the pointers to be decremented after each transfer. If the B register is not zero, the transfer will stop if a character which is equal to the 2's complement of the B register is transferred (stops after the matching character is moved)

Entry: HL=location of first byte
DE=location of first destination byte
C =number of bytes to move (C=1 to 255; Ø for 256)
B = 2's complement of terminating character if not Ø.
A =8-bit value added to each byte as it is moved (for de-zoning and zoning decimal numbers)

Exit: HL=⟩ location past last source byte
DE=⟩ location past last destination byte
A =entry value
B =entry value
C =zero or count after terminator character found
Condition flags are all altered

Stack: 2 entries used
Timing: 4.8+C*3.0 for BT
4.8+C*3.2 for BTR

Caution: Since BT and BTR instructions can take up to 820us to execute, care must be exercised in their use if time critical interrupt driven programs are to be simultaneously executed.

Binary Field Add with Carry or Subtract with Borrow

Mnemonic: BFAC or BFSB
Opcode: Ø11 or Ø31

Length: 1 byte

Operation: These instructions take the field pointed to by HL and either add it to or subtract if from the field pointed to by DE, leaving the result in the field pointed to by DE. The fields may be 1 through 16 bytes in length.

Entry: HL=location of right hand byte of the operand field
DE=location of right hand byte of the accumulator field
C =the field width (1 through 16; Ø or 16 implies 16)
Carry=carry or borrow into the operation

Exit: HL=location of left of the left hand byte of the operand field
DE=location of left of the left hand byte of the accumulator field
C = indeterminate
Carry=carry or borrow out of the operation (all the condition flags are altered)

Algorithm:
1. Load the implicit register from C.
2. Get the byte pointed to by HL.
3. Add with carry it to or subtract with borrow it from the byte pointed to by DE and store the result where DE points.
4. Decrement HL and DE by one.
5. Decrement the implicit register by one.
6. Go to step 2 if the implicit register is not now zero.

Stack: 2 entries used
Timing: 4.8+C*(2.8)

Block Compare

Mnemonic: BCP
Opcode: Ø41
Length: 1 byte

Operation: This instruction matches two strings of bytes from left to right until either a mismatch is found or the specified maximum number of bytes has been scanned.

Entry: HL=location of left hand byte of the subtracting field
DE=location of left hand byte of the subtracted from field
C =the maximum number of bytes to scan (1 thru 255; Ø implies 256)

Exit: IF A MISMATCH WAS FOUND:
HL=location of left hand byte of the subtracting field
DE=location of left hand byte of the subtracted byte
C =entry value minus number of bytes that matched
Condition flags all reflect the result of the subtract instruction that found the two bytes differing.
IF ALL BYTES MATCHED:
HL=location after the last byte in the subtracting field
DE=location after the last byte in the subtracted from field
C =zero
Condition flags are all altered (Zero true)

Algorithm:
1. Get the byte pointed to by HL.
2. Subtract from it the byte pointed to by DE
3. Increment DE and HL.
4. Stop if the Zero condition is false.
5. Decrement C.
6. Go to step 1 if C not equal to zero.
7. Exit with the Zero condition true.

Stack: 2 entries used
Timing: 7.0+N*(2.8) if N bytes matched before mismatch
5.2+C*(2.8) if all C bytes matched.
Caution: BCP can take up to 722μs to execute

Decimal Field Add with Carry

Mnemonic: DFAC
Opcode: 111 Ø41
Length: 2 bytes

Operation: This instruction takes the field of zoned BCD digits pointed to by HL and adds it to the field of zoned decimal digits pointed to by DE, leaving the result in the field pointed to by DE. The zone bits of the result field are set to the zone bits in the B register. The fields may be 1 through 16 bytes in length.

Entry: Same as for the BFAC instruction except
Exit: Same as for the BFAC instruction.
B=output zoning (right 4 bits must be Ø; le
4 bits must be other than ØØØØ.) except A register is destroyed. B=entry value

Algorithm:
1. Load the implicit register from C.
2. Get the byte pointed to by HL.
3. Add with carry it to the byte pointed to by DE.
4. Strip away the zone bits.
5. Clear the Carry and go to step 7 if the result is less than 1Ø.
6. Subtract 1Ø from the result and set the Carry.
7. Set the zoning bits.
8. Store the result where DE points.
9. Decrement HL and DE by one.
10. Decrement the implicit register by one.
11. Go to step 2 if the implicit register is not now zero.

Stack: 2 entries used.
Timing: 5.4+C*4.6 if a carry occurred on every digit.
5.4+C*4.4 if no carries occurred.

Note: The binary values for the zoned BCD digits with xxxx not equal to ØØØ are as follows (the digits are not packed, i.e., only one digit per byte):

Ø: xxxxØØØØ
1: xxxxØØØ1
2: xxxxØØ1Ø
3: xxxxØØ11
4: xxxxØ1ØØ
5: xxxxØ1Ø1
6: xxxxØ11Ø
7: xxxxØ111
8: xxxx1ØØØ
9: xxxx1ØØ1

Decimal Field Subtract with Borrow
Mnemonic: DFSB
Opcode: Ø62 Ø41
Length: 1 byte
Operation: This instruction takes the field of zoned BCD digits pointed to by HL and subtracts it from the field of zoned BCD digits pointed to by DE, leaving the result in the field pointed to by DE. The zone bits of the two fields must be identical. The zone bits of the result field are set to the zone bits in the B register. The fields may be 1 through 16 bytes in length.
Entry: same as for the DFAC instruction.
Exit: same as for the DFAC instruction.
Algorithm:
1. Load the implicit register from C.
2. Get the byte pointed to by HL.
3. Subtract with borrow it from the byte pointed to by DE.
4. Go to step 6 and clear the Carry if the byte result is not negative.
5. Add 1Ø to the result and set the Carry.
6. Set the zone bits to those in the B register.
7. Store the result where DE points.
8. Decrement HL and DE by one.
9. Decrement the implicit register by one.
10. Go to step 2 if the implicit register is not now zero.

Stack: 2 entries used
Timing: 5.4 + c*4.0 if a borrow occurred on every digit
5.4 + C*3.6 if no borrows occurred.

Binary Field Shift Left
Mnemonic: BFSL
Opcode: Ø75
Length: 1 byte
Operation: This instruction shifts a field of bytes in memory left one bit position as if all of the bytes made up one continuous word.
Entry: HL = location of right hand byte of the field
C = the field width (1 through 16; Ø or 16 implies 16)
Carry = bit shift in on the right
Exit: HL = location left of the left hand byte of the field
C = indeterminate
A = indeterminate
Carry = bit shifted out on the left (all other flags are indeterminate)
Stack: 2 entries used
Timing: 3.8 + C*2.2

Binary Field Shift Right
Mnemonic: BFSR
Opcode: 111 Ø75
Length: 2 bytes
Operation: This instruction is similar to BFSL except the shift is in the opposite direction.
Entry: HL = location of left hand byte of the field.
C = field width (1 through 16; Ø or 16 implies 16)
Carry = bit shifted in on the left
Exit: HL = location right of the right hand byte of the field
C = indeterminate
Carry = bit shifted out from the right (all other condition flags are altered)
Stack: 2 entries used
Timing: 4.2 + C*2.0

Multiple Input
Mnemonic: MIN
Opcode: Ø61
Length: 1 byte
Operation: This instruction moves the number of bytes specified in the C register from a buffered input device to the field pointed to by HL. The number of bytes moved is the number in the C register modulo 16. To make transferring up to 256 bytes easy yet interruptable, the full eight bit value of the C register is retained during loop counting and exit is made with the C register containing its entry value minus the number of bytes transferred, HL containing its entry value plus the number of bytes transferred, and the condition code reflecting the eight bit result of the last decrementation of the C register. Thus the interruptable loop for transferring the number of bytes indicated by the eight bit value in the C register yet not inhibiting interrupts more than 155 microseconds would appear as follows:

```
LOOP  LA    DEVADR
      DI
      EX    ADR
      EX    DATA
      EI
      MIN
      JFZ   LOOP
```

Note that the device must be re-addressed for each execution of the MIN instruction since an interrupt could cause some other device to be addressed. The MIN instruction causes an input strobe to be executed every 8.4 microseconds. This execution operates without regard to any status bits of any kind. There is no existing 2200 system I/O device capable of using this instruction and it is included for use with 5500 system I/O devices with faster buffers allowing them to be used at data rates equivalent to DMA channels. The MIN instruction has all of the advantages of a non-I/O device interrupting system (lower software overhead in high throughput situations, superior control over the occurrence of events allowing provability of correctness in the program logic and repeatability of event occurrance, and simpler hardware using lower speeds and noise

filtered buses) and yet achieves DMA throughput rates.
Entry: HL = location of first destination byte
C = number of bytes to move (this number is take modulo 16 and if it is Ø modulo 16 then 16 bytes will be moved)
Exit: HL = location of entry value plus number of bytes moved
C = entry value minus number of bytes moved
Algorithm: 1. Execute an INPUT strobe
2. Store the bytes where HL points
3. Increment HL
4. Load the implicit register from C
5. Decrement C using the ALU
6. Decrement the implicit register
7. Exit if the implicit register is zero
8. Decrement the P-counter
9. Re-fetch the instruction without allowing interrupts
Stack: 1 entry used
Timing: 8.4 microseconds per byte transferred
Note: To input a block of 256 bytes using the loop described under 'Operation' above would take 2550 microseconds if no interrupts occurred (an average of 10 us per byte).

Multiple Output
Mnemonic: MOUT
Opcode: Ø71
Length: 1 byte
Operation: This instruction is similar to the MIN instruction except for timing and the direction of information flow. MOUT moves the number of bytes specified in the C register from the field pointed to by HL to a buffered output device. A byte is written using the EX WRITE strobe every 8.8 us and interrupts can be inhibited for a maximum of 161 microseconds. As with MIN, there is no existing 2200 system I/O device capable of being used with the MOUT instruction.
Timing: 8.8 microseconds per byte transferred.
Note: To output a block of 256 bytes using a loop similar to the one described under 'Operation' in MOUT instruction description would appear where a MIN instruction appears in that example) would take 2650 microseconds if no interrupts occurred (an average of 10.4 us per byte).

5.8.6 Category 4 – Processor state save and restore Instructions

Stack Store
Mnemonic: STKS
Opcode: Ø65
Length: 1 byte
Operation: The Stack Store instruction POPs a specified number of stack entries and stores them (LSB followed by MSB) in the field pointed to by HL.
Entry: HL = first location in the storage area
C = the number of entries to be stored (1 through 16; Ø or 16 implies 16)
Exit: HL and C indeterminate
Condition flags unchanged
Timing: 1.6 + C*2.4

Stack Load
Mnemonic: STKL
Opcode: 111 Ø65
Length: 2 bytes
Operation: The Stack Load instruction PUSHes onto the stack the specified number of entries from the field pointed to by HL. Upon entry, HL points to the right hand byte and the entries are loaded in reverse order to allow restoring the stack from locations stored using the STSK instruction.
Entry: HL = last location in the storage area
C = the number of entries to be PUSHed (1 through 16; Ø or 16 implies 16)
Exit: HL = indeterminate
C = indeterminate
Condition flags unchanged.
Timing: 4.4 + C*2.2

Register Store
Mnemonic: REGS
Opcode: Ø55
Length: 1 byte
Operation: The Register Store instruction stores all of the registers for the currently selected mode (Alpha or Beta) in the field pointed to by the top entry of the stack. This entry points to the right hand byte of the field and the registers are stored in reverse order moving to the left (X L H E D C B A from right to left). When the instruction terminates, the top entry of the stack points to the left of the left hand byte in the field. For example, if entry is made with the top entry of the stack pointing to location Ø2ØØ7 (octal), the registers are stored as follows:

Ø2ØØØ: A
Ø2ØØ1: B
Ø2ØØ2: C
Ø2ØØ3: D
Ø2ØØ4: E
Ø2ØØ5: H
Ø2ØØ6: L
Ø2ØØ7: X

In the above example, the top entry of the stack will be Ø1777 when the instruction terminates. The contents of neither the registers or the condition flags for the given mode are altered by this instruction.
Timing: 13.2

Register Load
Mnemonic: REGL
Opcode: 111 Ø55
Length: 2 bytes

Operation: The Register Load instruction loads all of the registers for a given mode (Alpha or Beta) from the field pointed to by HL. Upon entry, HL points to the right hand byte of the field. The registers are loaded in reverse order moving to the left in the field. In this manner, the registers can be reloaded from values stored by the REGS instruction. In the example given for the REGS instruction, if the REGS instruction were entered with HL = Ø2ØØ7, the registers shown would be loaded from the locations shown. The condition flags are not altered by this instruction.
Timing: 12.0

Condition Code Save
Mnemonic: CCS ⟨r⟩
Opcode: [r] Ø42
Length: 2 bytes
Operation: This instruction loads register ⟨r⟩ with a value such that if the value is added to itself using the AD operation the condition flags will all be restored to their state before the CCS instruction was executed. The logic equations for the value loaded into ⟨r⟩ are as follows:

A7 = Carry
A6 = Sign
A5 = A4 = A3 = A2 = Ø
A1 = Not Zero and Not Sign
AØ = Not Zero and Not Parity

This instruction does not alter the state of any of the condition flags.
Timing: 2.4 to 3.0 depending upon the condition flag states.

5.8.7 Category 5 — Address Manipulation Instructions

Increment Register Pair

Mnemonics	Opcodes	Timing
INCP HL	Ø15	2.8
INCP HL,2	117 Ø15	2.8
INCP HL,A	Ø17	3.0
INCP BC	Ø62 Ø15	2.6
INCP BC,2	113 Ø15	2.8
INCP BC,A	Ø62 Ø17	2.8
INCP DE	174 Ø15	2.6
INCP DE,2	115 Ø15	2.8
INCP DE,A	174 Ø17	2.8

Operation: These instructions increment the indicated register pair by either one, two, or the contents of the A-register. The increment value is added to the LSB register and then the carry is added to the MSB register. All conditions are indeterminate. The A-register is not changed.

Decrement Register Pair

Mnemonics	Opcodes	Timing
DECP HL	Ø35	2.8
DECP HL,2	117 Ø35	2.8
DECP HL,A	Ø37	3.0
DECP BC	Ø62 Ø35	2.6
DECP BC,2	113 Ø35	2.8
DECP BC,A	Ø62 Ø37	2.8
DECP DE	174 Ø35	2.6
DECP DE,2	115 Ø35	2.8
DECP DE,A	174 Ø37	2.8

Operation: These instructions decrement the indicated register pair by either one, two, or the contents of the A-register. The decrement value is subtracted from the LSB register and then the borrow is subtracted from the MSB register. All condition flags are indeterminate. The A-register is not changed.
condition flags are indeterminate.

Double Load

Mnemonics	Opcodes	Timing
DL DE,HL	Ø47	3.4
DL BC,HL	111 Ø47	5.2
DL BC,BC	Ø62 Ø47	4.6
DL BC,DE	113 Ø47	5.0
DL DE,BC	174 Ø47	4.6
DL DE,DE	115 Ø47	5.0
DL HL,BC	176 Ø47	4.6
DL HL,DE	117 Ø47	5.0
DL HL,HL	Ø57	4.0

Operation: These instructions load the register pair specified by the first operand from the memory location pointed to by the register pair specified by the second operand. The LSB register (C, E, or L) is loaded from the specified memory location and the MSB register (B, D, or H) is loaded from the next higher memory location. Note that indirect addressing can be accomplished by loading a register pair from the locations that the pair specify (DL HL,HL for example).

Double Store

Mnemonics	Opcodes	Timing
DS DE,HL	Ø27	3.4
DS BC,HL	111 Ø27	5.2
DS BC,DE	113 Ø27	5.0
DS DE,BC	174 Ø27	4.6
DS HL,BC	176 Ø27	4.6
DS HL,DE	117 Ø27	5.0

Operation: These instructions store the register pair specified by the first operand into the memory locations pointed to by the register pair specified by the second operand. The LSB register (C, E, or L) is stored in the specified memory location and the MSB register (B, D, or H) is stored in the next higher memory location.

Paged Load

Mnemonics	Opcodes	Timing
PL A,⟨loc⟩	1Ø5 LSP	3.0
PL B,⟨loc⟩	114 LSP	3.0
PL C,⟨loc⟩	124 LSP	3.0
PL D,⟨loc⟩	134 LSP	3.0
PL E,⟨loc⟩	144 LSP	3.0
PL H,⟨loc⟩	154 LSP	3.0
PL L,⟨loc⟩	164 LSP	3.0

Operation: These instructions load the specified register from the memory location specified by the LSP given in the instruction and the MSP given in the X-register

Paged Store

Mnemonics	Opcodes	Timing
PS A,⟨loc⟩	1Ø7 LSP	3.0
PS B,⟨loc⟩	116 LSP	3.0
PS C,⟨loc⟩	126 LSP	3.0
PS D,⟨loc⟩	136 LSP	3.0
PS E,⟨loc⟩	146 LSP	3.0
PS H,⟨loc⟩	156 LSP	3.0
PS L,⟨loc⟩	166 LSP	3.0

Operation: These instructions store the specified register in the memory location specified by the LSP given in the instruction and the MSP given in the X-register.

Double Paged Load

Mnemonics	Opcodes	Timing
DPL BC,⟨loc⟩	111 124 LSP	4.8
DPL DE,⟨loc⟩	113 144 LSP	4.8
DPL HL,⟨loc⟩	115 164 LSP	4.8

Operation: These instructions load the specified register pair from the memory locations specified by the LSP given in the instruction and the MSP given in the X-register. The C, E, or L register is loaded from the specified memory location and the B, D, or H register is loaded from the next higher location.

Double Paged Store

Mnemonics	Opcodes	Timing
DPS BC,⟨loc⟩	111 126 LSP	4.8
DPS DE,⟨loc⟩	113 146 LSP	4.8
DPS HL,⟨loc⟩	115 166 LSP	4.8

Operation: These instructions store the specified register pair in the locations specified by the LSP given in the instruction and the MSP given in the X-register. The C, E, or L register is stored in the specified location and the B, D, or H register is stored in the next higher location.

Increment Index and Decrement Index

Mnemonics	Opcodes	Timing
INCI ⟨disp⟩,⟨index⟩	ØØ5 LSP [i]	7.6
DECI ⟨disp⟩,⟨index⟩	Ø25 LSP [i]	7.8
INCI *⟨disp⟩,⟨index⟩	111 ØØ5 LSP MSP [i]	9.6
DECI *⟨disp⟩,⟨index⟩	111 Ø25 LSP MSP [i]	9.8

Operation: The processor has a construct called an index which is a 16-bit value kept in memory. The concept is similar to index registers except that all the values are kept in the page of memory pointed to by the X-register. The index is specified by a single byte in the instructions (shown as [i] above) which points to the memory location containing the LSB of the index value, the MSB being in the next higher memory location ([i] specifies the LSP of the index address while the X-register specifies the MSP of the index address). The instruction also contains a displacement that is either one or two bytes in length (depending upon the opcode). These instructions either increment or decrement the value of the index by the displacement The carry condition flag reflects the carry or borrow from the incrementation or decrementation. The rest of the condition flags are indeterminate

Load from Index Incremented or Decremented

Mnemonics	Opcodes	Timing
LFII BC,⟨disp⟩,⟨index⟩	Ø62 ØØ5 LSP [i]	7.4
LFID BC,⟨disp⟩,⟨index⟩	Ø62 Ø25 LSP [i]	7.6
LFII BC,*⟨disp⟩,⟨index⟩	113 ØØ5 LSP MSP [i]	8.4
LFID BC,*⟨disp⟩,⟨index⟩	113 025 LSP MSP [i]	8.6
LFII DE,⟨disp⟩,⟨index⟩	174 ØØ5 LSP [i]	7.4
LFID DE,⟨disp⟩,⟨index⟩	174 Ø25 LSP [i]	7.6
LFII DE,*⟨disp⟩,⟨index⟩	115 ØØ5 LSP MSP [i]	8.4
LFID DE,*⟨disp⟩,⟨index⟩	115 Ø25 LSP MSP [i]	8.6
LFII HL,⟨disp⟩,⟨index⟩	176 ØØ5 LSP [i]	7.4
LFID HL,⟨disp⟩,⟨index⟩	176 Ø25 LSP [i]	7.6
LFII HL,*⟨disp⟩,⟨index⟩	117 ØØ5 LSP MSP [i]	8.4
LFID HL,*⟨disp⟩,⟨index⟩	117 Ø25 LSP MSP [i]	8.6

Operation: These instructions are similar to the INCI and DECI instructions except that they load the specified pair of registers with the result of adding or subtracting the displacement to or from the index value instead of updating the value of the index. The condition flags are similarly affected.

Stack: 1 entry used

Base Register Load

Mnemonic: BRL⟨r⟩
Opcode: [r]Ø72
Length: 2 bytes

Operation: This instruction loads the base register from the specified register. Note that the base register cannot be saved. For this reason, loading the base register will normally be a monitor function, allowing the monitor to keep within itself the value of the base register for user state storage purposes. This instruction will cause a privileged instruction interrupt if the USER mode flag is set.

Timing: 1.2

NOP Jump

Mnemonic: NOJ
Opcode: Ø45
Length: 3 bytes.

Operation: This instruction causes no operation to be performed. It is useful for overstoring jump instructions which might be executed while being overstored. The procedure to overstore a jump instruction would be to first overstore the opcode with an Ø45 (NOP Jump) and then update the address portion. Then the opcode could be overstored with the appropriate jump instruction. The primary use of this instruction is for overstoring the interrupt vector jump instructions for the interrupts which cannot be disabled (such as memory parity fault) and which might happen while the jump is being overstored.

Timing: 1.4

5.8.8 Category 6 – Operating System Control

System Call
Mnemonic: SC
Opcode: Ø67
Operation: This instruction causes the USER mode flag to be cleared, the last entry in the sector table to be set to the last 4K section of physical memory space with access protection, and a CALL to be performed to location Ø17ØØ25 (in the ROM). This is the mechanism via which the user would communicate with an operating system that used the USER mode.
Timing: 1.8

User Return
Mnemonic: UR
Opcode: 111 1Ø2
Operation: This instruction is identical to the RETURN instruction (opcode ØØ7) except that additionally the USER mode flag is set.
Timing: 2.Ø

Sector Table Load
Mnemonic: STL
Opcode: Ø77
Operation: This instruction loads the first 15 entries in the sector table. This table contains six bits for each entry. The right hand two bits are not used and should always be set to zero. The 2^2 bit is set for access enable. The 2^3 bit is set for write enable. The left hand four bits are used to map that entry to particular 4K section of physical memory space. This instruction will cause a privileged instruction interrupt if the user mode flag is set.

Entry:	HL = location first byte in table of 15 to load. C = number of entries to load (Ø to 15)
Exit:	A = last value loaded
Stack:	1 entry used
Timing:	27.4

Breakpoint
Mnemonic: BP
Opcode: Ø52
Timing: 1.8
Length: 1 byte
Operation: This instruction is similar to a system call (SC) instruction except the call is performed to location Ø17ØØ3Ø. This will cause entry into the system DEBUG routine if the vector location is not changed.

6. Input/Output

The 55ØØ/55Ø2 communicates with and exercises program control over external devices via the Input/Output System Bus. The keyboard, display and cassette tapes, while internal, are also peripherals that operate from this bus.

All external devices are connected to the Bus in parallel, "daisy chain" fashion.

Each external device is assigned (by jumpers in the peripheral device's controller) an Input/Output "address" which is unique to that device. At any time, one device is designated by the processor as currently addressed and communication between the processor and that device only is possible. All other devices on the Bus are logically, although not electrically, disconnected from the Bus.

Signals in the Bus may be divided into six groups. These are:

► Nine output lines designated AOUTØ(−) thru AOUT8(−). The eight AOUTØ(−) thru AOUT7(−) lines carry data and control information from the processor to the external device. AOUT8(−) is the parity bit for these lines. The (−) signs indicate that the data is logically inverted, i.e., low voltage equals a binary one.

► Nine input lines designated AINØ(−) thru AIN8(−). The eight AINØ(−) thru AIN7(−) lines carry data and status information from the external device to the processor. AIN8(−) is the parity bit for the other 8 lines.

► Nine output control and data strobes.

► The system clock line.

► The output data parity error flag line from the external devices to the processor.

► Twelve power and ground lines. Four of these are ground lines and serve as signal ground for the Bus and power ground for devices which obtain operating power from the Bus.

6.1 Input/Output Physical Connections

The Input/Output System Bus connector on the 55Ø1/55Ø2 is a 5Ø pin Amphenol Series 17 receptacl with female contacts, with provisions for screw lock assemblies.

Each external device must have two 5Ø pin Input/Output Bus connectors; one an Amphenol Series 17 female plug with male contacts labelled "I/O Bus In" and the other an Amphenol Series 17 male plug with female contacts (same as the 55Ø1/55Ø2) labelled "I/O Bus Out". Both of these connectors must have provisions for screw lock assemblies.

Datapoint Universal Input/Output cables have a male connector at one end and a female at the other.

Connection is made from the 55Ø1/55Ø2 I/O connector to the "I/O Bus In" connector of an external device via a Universal I/O cable. If more than one device is connected to the Bus, connection is made from the "I/O Bus Out" connector of the first device to the "I/O Bus In" connector of the second device with a Universal I/O Cable. The process is repeated for other external devices.

Every external device must connect each of the 5Ø pins (including spares) of its "I/O Bus In" connector to the corresponding pin of its "I/O Bus Out" connector in addition to connection to those lines required for the particular device. This is required for continuity of all signal, power and ground lines in the Bus.

The following table gives I/O Bus pin assignments:

I/O Bus Pin Assignments

Signal	*Pin Number*
AOUTØ(–)	44
AOUT1(–)	45
AOUT2(–)	46
AOUT3(–)	29
AOUT4(–)	3Ø
AOUT5(–)	31
AOUT6(–)	32
AOUT7(–)	33
AOUT8(–) (Parity Out)	34
INPUT(–)	12
EX ADR(–)	15
EX STATUS(–)	13
EX DATA(–)	14
EX WRITE(–)	19
EX COM1(–)	2Ø
EX COM2(–)	21
EX COM3(–)	22
EX COM4(–)	23
SYSTEM CLOCK	39
AINØ(–)	1
AIN1(–)	2
AIN2(–)	3
AIN3(–)	4
AIN4(–)	5
AIN5(–)	6
AIN6(–)	7
AIN7(–)	18
AIN8(–) (Parity In)	17
PERR(–) (Parity Error Flag)	16
GROUND	4Ø, 41, 42, 43
+5V	8, 9, 1Ø, 11
–5V	27
+12V	25
–12V	24
+24V	26
Spare (unused)	28, 35, 36, 37, 38, 47, 48, 49, 5Ø

(–) indicates inverted logic

6.2 Input/Output Electrical and Timing Requirements

This section describes interface circuits and timing requirements for operation of external devices on the 55Ø1/55Ø2 Input/Output System Bus

6.2.1 Output Line Circuits

Three output line driver circuits are used in the Datapoint 55ØØ These are all illustrated in Figure 6.1.

Figure 6.1-a is the driver used for the eight AOUTØ(-) thru AOUT7(-) data lines and the AOUT8(-) parity line.

Figure 6.1-b is the driver used for the system clock line and all strobe lines except the INPUT(-) strobe.

Figure 6.1-c is the driver used for the INPUT(-) strobe line

The external device must use the receiver shown in Figure 1-d for all of these lines. This receiver is a differential comparator whose reference voltage is +1 8 volts DC The input is filtered for transient noise immunity and is diode clamped to the reference voltage Resistive source impedance for the reference must be less than 50 ohms and should be bypassed as required for reliable operation.

Current flowing into the line input of the differential comparator device (excluding current thru the clamp diodes to the reference voltage) must be less than 50 microamperes.

The minimum voltage slew rate at the line input to the comparator device (after the resistor-capacitor rolloff network) in response to a change of state of the I/O Bus signal is .7 volt per microsecond. At this slew rate, the delay from the time the line input to the comparator device is equal to the reference voltage (changing in either direction) until the comparator output changes state must be less than .2 microseconds.

In addition, plus and minus .1 volt hysteresis must be incorporated in the receiver for any output strobes whose transitions (versus level) are used by the interface logic circuitry. This requirement prevents logic malfunctions due to spurious receiver responses (multiple transitions) to an output strobe signal.

6.2.2 Input Line Circuits

The eight AINØ(-) thru AIN7(-) lines carry data or status information from the selected external device to the processor. Input line AIN8(-) is the parity bit for the input data lines.

Each external device connects to each of these 9 lines by means of the 'tri-state' gate circuit shown in Figure 6.2-a. When the device is enabled for data or status input, the tri-state gates are enabled and the logic levels for the AIN(-) lines are:

1 = Ø volts
Ø = +5 volts

Unless enabled, the external device must maintain these gates in the 'off' (high impedance) state.

All external devices connect to the AIN(-) lines in parallel with similar circuits.

Figure 6.2-b shows the circuit used for the parity error return line PERR(-). It is an open collector gate circuit and the conditions for its output are given in Section 6.2.5.

The +5 volt pullup voltage for the 4.7k pullup resistors must be that provided by the 55Ø1/55Ø2 in the I/O Bus, so that even if the external device interface is powered independent of the processor and is turned off, the Bus will be operative. In addition, the tri-state gates used for the AIN(-) lines must be powered from the I/O Bus +5 volt line and must be maintained in the disabled (high impedance) state when power is removed from the external device. (Note the voltage variation range of the I/O Bus +5 volt line in 6.2.3).

The 4.7 pullup resistor and the tri-state gate must be present on all AIN(-) lines even if the device does not logically use the line.

Figure 6.2-c shows the line receiver used in the

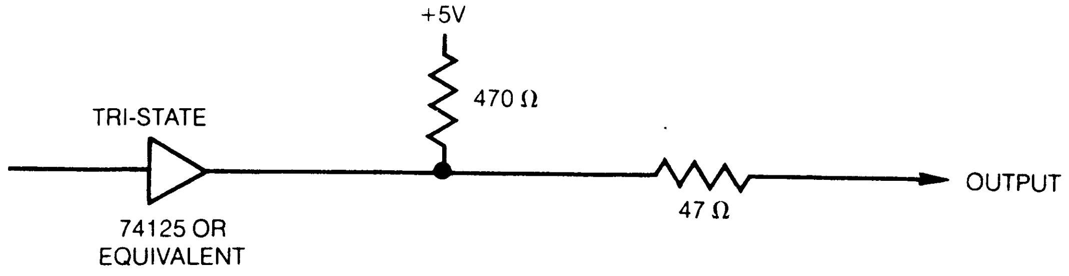


FIG 6.1-A AOUT(–) DRIVERS

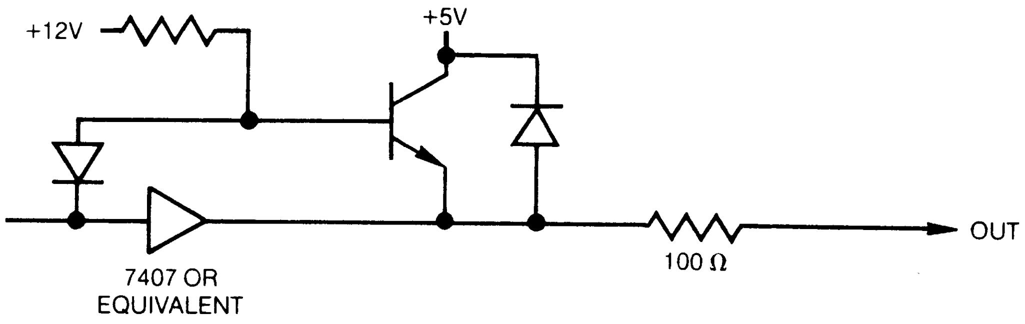


FIG 6.1-B STROBE (EXCEPT INPUT(–)) DRIVERS

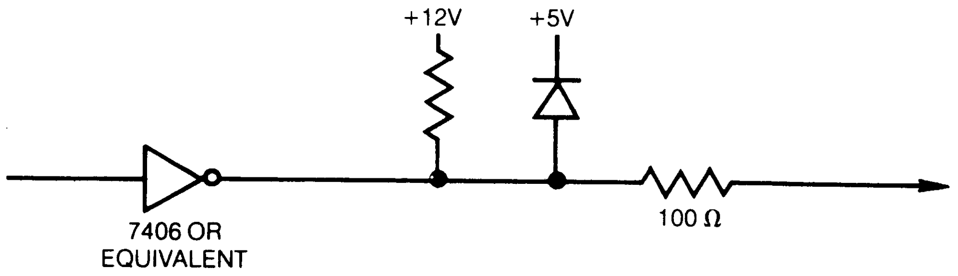


FIG 6.1-C INPUT(–) STROBE DRIVER

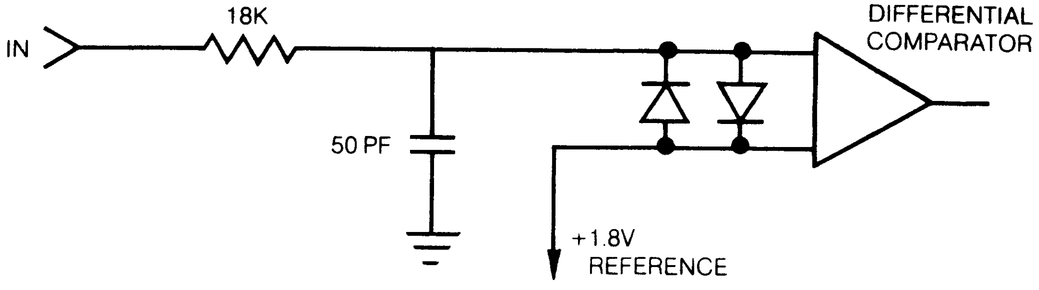


FIG 6.1-D EXTERNAL DEVICE RECEIVER

55Ø1/55Ø2 for the nine AIN(-) lines and Figure 6.2-d shows the circuit used for the PERR(-) line.

6.2.3 Power and Ground Lines

The Input/Output System Bus provides ground (common signal and power) and various supply voltages for operation of external devices on the Bus.

Each external device must connect all 12 power and ground lines (see table in 6.1) between its 'I/O Bus In' connector and 'I/O Bus Out' connector in addition to +5V and ground connections to its own circuitry.

Except as discussed in 6.2.2 above (+5 volts) current must not be drawn from these voltages by the external device.

The I/O Bus +5 volt line may vary in voltage from +4.8 volts to +7.5 volts. The external device must operate without damage or malfunction over this range.

Internal wiring, connectors and printed circuit board connections must be adequate to insure that no damage can occur to the external device if the I/O Bus +5 volt and ground circuits are shorted in the device. Maximum short circuit current for the I/O Bus +5 volt circuit is 7 Amperes.

6.2.4 Device Address

The processor addresses an external device by means of the EX ADR(-) strobe. The address of the device to be selected appears on the AOUTØ(-) thru AOUT7(-) lines. As in all output operations, AOUT8(-) provides odd parity information (i.e., the total number of logic 1's in the 9 AOUTØ(-) thru AOUT8(-) lines is odd).

All AOUT(-) lines are stable from 2.Ø microseconds before the leading (negative) edge of the EX ADR(-) strobe until 2.Ø microseconds after the trailing (positive) edge of the strobe.

Logic levels on the nine AOUT(-) lines are as follows:

Logic 1 = Ø volts
Logic Ø = +5 volts

The device whose address appears on the AOUTØ(-) thru AOUT7(-) lines must edge-trigger to the addressed state on the leading (negative) edge of the EX ADR(-) strobe if the 9 bit parity result is correct.

If the parity result is incorrect the device remains or becomes unaddressed and indicates an I/O Bus parity error by taking the PERR(-) line to Ø volts. (See 6.2.5), even if the presented address appears to be its own.

If the presented address is not its own but the parity result is correct, the device merely remains or becomes unaddressed; it does not take the PERR(-) line to Ø volts.

Once addressed, the device stays addressed until another EX ADR(-) strobe occurs and the AOUT(-) lines indicate an address other than its own.

The device must recognize output strobes (other than EX ADR(-)) or place data or status information on the AIN(-) lines only while addressed (see 6.2.5 and 6.2.6).

The device must be forced to the unaddressed state by initial application of +5 volt power from the I/O Bus and must also be set to the unaddressed state by initial application of its own logic supply voltage.

In addition, the device must insure that neither the tri-state drivers on the AIN(-) lines (see 6.2.6) nor the PERR(-) driver circuit (see 4.2.5) become erroneously enabled *for any period of time* during application or removal of the device logic supply voltage.

Although all eight AOUTØ(-) thru AOUT7(-) lines are used to carry address information, only 16 device addresses are used. To simplify address strapping in external devices, a four bit address code is employed. The address itself appears on the least significant 4 bits AOUTØ(-) thru AOUT3(-). (Note that these lines are inverted; logic 1 = Ø volts, logic Ø = +5 volts). The logical complement of these four bits appear on AOUT4(-) thru AOUT7(-). Thus, 'system address' 3 is represented as

3		Ø			3		
A7	A6	A5	A4	A3	A2	A1	AØ
1	1	Ø	Ø	Ø	Ø	1	1
3				3			
inverse device no.				device no.			

The complete (8 bit) octal address here is 3Ø3.

Only a four input gate is required to detect any of these addresses. Strapping must be arranged so that each gate input connects to one of two AOUT lines as follows:

Gate Input Number	To	Or
1	AOUTØ	AOUT4
2	AOUT1	AOUT5
3	AOUT2	AOUT6
4	AOUT3	AOUT7

Typical logic implementation of these functions is illustrated in Figure 6.3.

Address strapping will be provided by means of a plug with selective wiring or mechanical posts to which wires will be soldered. Standard pads on a printed circuit board will not be used for connection of wires.

6.2.5 Data and Control Output

All data or control information is transferred from the processor to external devices using one of the following strobes:

EX DATA(-)
EX STATUS(-)
EX WRITE(-)
EX COM1(-)
EX COM2(-)
EX COM3(-)
EX COM4(-)

Each of these is a 2.Ø microsecond negative pulse to the Ø volt level.

Logic levels on the AOUTØ(-) thru AOUT8(-) lines are as follows:

1 = Ø volts
Ø = +5 volts

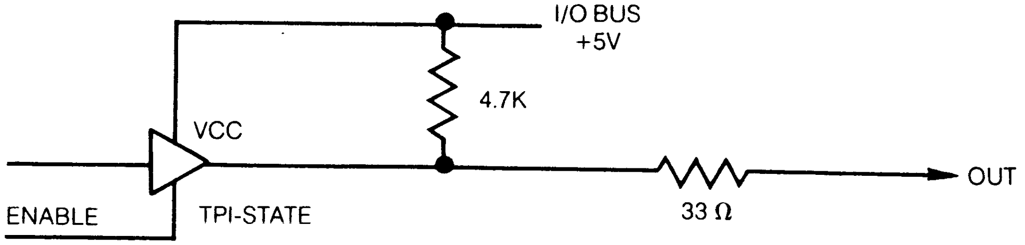


FIG 6.2-A EXTERNAL DEVICE STATUS/DATA DRIVER

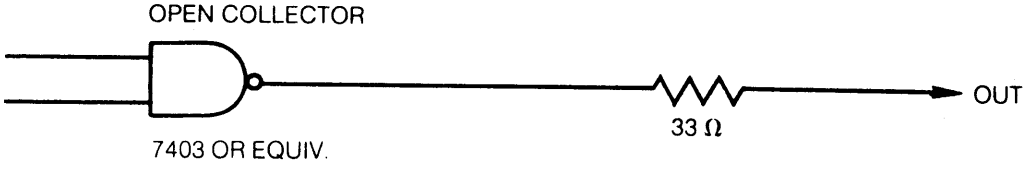


FIG 6.2-B EXTERNAL DEVICE PERR(−) DRIVER

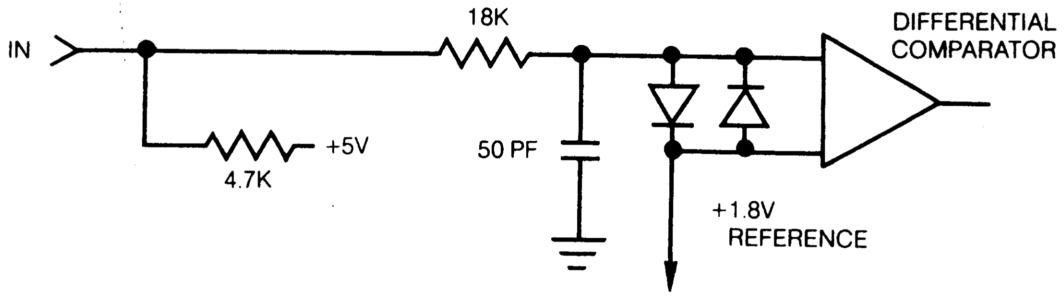


FIG 6.2-C STATUS/DATA LINE RECEIVERS

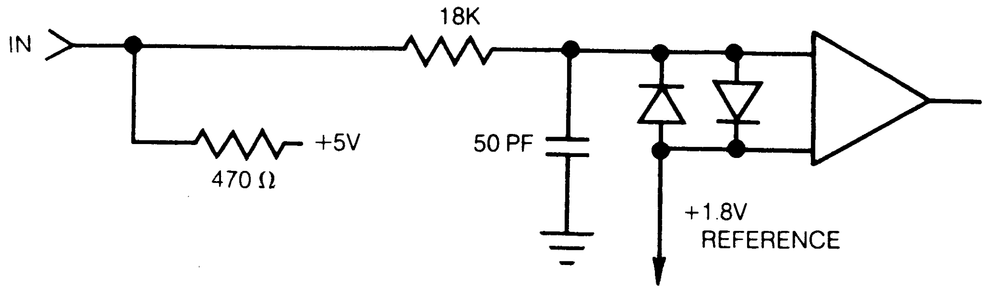


FIG 6.2-D PERR(−) RECEIVER

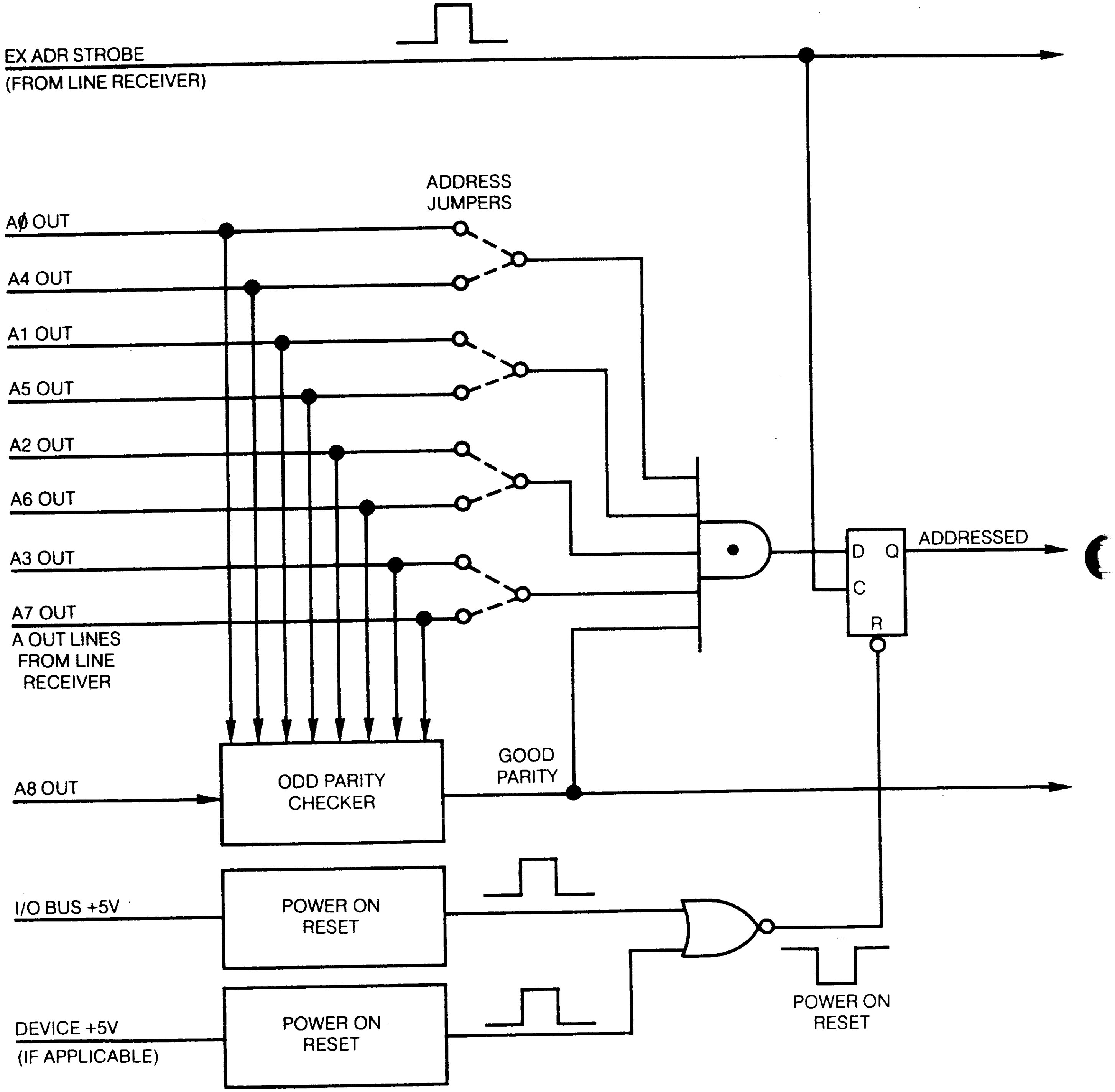


FIGURE 6.3 DEVICE ADDRESS LOGIC

Except for EX ADR(-), these strobes must not be recognized by the device unless it is addressed.

Odd parity is used on the AOUTØ(-) thru AOUT8(-) lines; i.e., the state of AOUT8(-) is such that the number of logic 1's (Ø volts) on these 9 lines is odd.

The processor is capable of two types of output operation: normal and multiple.

In normal output mode, the nine AOUT(-) lines are stable from 2.Ø microseconds before the leading (negative) edge of the strobe until 2.Ø microseconds following the trailing (positive) edge of the strobe.

In multiple output mode, up to 16 (program determined) consecutive EX WRITE output strobes can be executed using the timing shown in Figure 6.4-b. The AOUT(-) lines are valid 2.Ø microseconds before the leading (negative) edge of the 2.Ø microsecond strobe as above, but only remain valid until the trailing (positive) edge. The time from the leading edge of one strobe to the leading edge of the next is 8.Ø microseconds and new data is presented on the AOUT(-) lines with each strobe.

Devices required to utilize the multiple output operation must be able to accept data at the rate of one byte every 8 microseconds. Buffered devices required to utilize only the normal output operation must be able to accept data at the rate of one byte every 2Ø.Ø microseconds as long as buffer space is available and conditions prevail which allow data to be placed in the buffer.

In either normal or multiple output operations, the device must edge-load the contents of the AOUT(-) lines on the leading (negative) edge of the strobe.

If the external device is addressed when a strobe is received and the 9 bit parity result is incorrect, the device must set the PERR(-) line to Ø volts on the leading edge of the strobe. This requirement pertains to only those strobes used by the external device. If a strobe is not used at all, the device may ignore parity for the strobe. Note, however, that the device must check parity on all strobes used even if the data given with the strobe is not used. This is a validity check upon the existence of the strobe.

In addition to setting the PERR(-) line to Ø volts when incorrect parity is detected, the device must ignore the strobe if it would cause an irreversable action in the device. The definition of irreversability is device dependent and is made specific in each device's specification.

Once the PERR(-) line has been set to Ø volts, the device must maintain this state until another strobe (including EX ADR(-)) is received with correct parity.

Figure 6.5 shows a typical logic implementation of these functions.

PERR(-) must be initialized to the 'off' state upon initial application of I/O Bus +5 volts or the device logic supply voltage.

6.2.6 Status/Data Input

Both data (if applicable) and status are transmitted from the external device to the processor over the eight AINØ(-) thru AIN7(-) lines. Input line AIN8(-) is the parity bit for the other AIN(-) lines. The device must be configured for odd parity; i.e., the number of logic 1's (Ø volts) on these 9 lines must be odd.

The device is in status mode and will place status information on the AIN(-) lines immediately after being addressed or upon receipt of an EX STATUS(-) strobe. The device is placed in data mode and will place data on the AIN(-) lines immediately upon receipt of an EX DATA(-) strobe. If the device is in data mode, either an EX ADR(-) or EX STATUS(-) strobe returns it to status mode.

All data/status mode changes must be activated by the leading (negative) edge of the associated strobe.

The device must maintain all AIN(-) lines in the 'off' (high impedance) state while it is not addressed and for 1.5 microseconds (nominal) after becoming addressed. The 1.5 microseconds delay prevents the tri-state drivers from being enabled before the drivers of another device become disabled. Note that this delay applies only to to enabling the tri-state drivers; all other logic functions on the interface may respond to becoming addressed on the leading (negative) edge of the EX ADR(-) strobe.

An output-only device (such as a printer) which does not transmit data to the processor need not incorporate the two modes; rather, it may stay in status mode at all times.

See Figure 6.6 for typical logic implementation of these functions.

The processor is capable of two types of input operation: normal and multiple.

In normal input mode, a negative going 2.Ø microsecond INPUT(-) strobe is generated by the processor to indicate to the addressed device that the date or status information on the AIN(-) lines has been taken. The AINØ(-) thru AIN8(-) lines must be valid 3.5 microseconds before the leading edge of the INPUT(-) strobe. This time is with respect to the INPUT(-) strobe at the processor I/O Bus connector and does not include cable or device line receiver delay.

The data lines are sampled by the processor on the leading (negative) edge of the INPUT(-) strobe, so the device may change the AIN(-) lines immediately after detection of this edge.

In multiple input mode, up to 16 (program determined) INPUT(-) strobes may occur at 8.Ø microsecond intervals (8.Ø microseconds between leading edges). The device must present new valid data within 4.5 microseconds after the leading edge of the INPUT(-) strobe. This time is with respect to the INPUT(-) strobe at the I/O Bus connector and does not include cable or device line receiver delays.

Buffered input devices designed to work in normal input mode only must present new valid data within

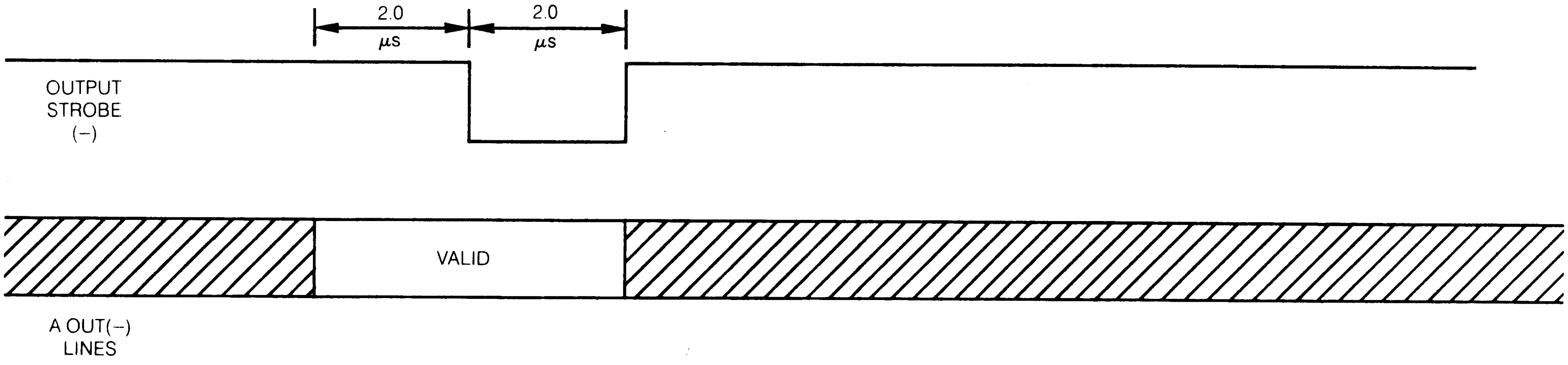
2Ø.Ø microseconds after the leading edge of the INPUT(-) strobe (at the I/O Bus connector) if data still resides in the buffer and conditions prevail which would not inhibit its availability.

All devices must be able to present valid status or data within 4.5 microseconds of the leading edge of any I/O Bus strobe (at the I/O Bus connector). This does not mean that new data must be available within this time but that the device must either give a status indicating that new data is not ready or, once having indicated that new data is ready, be able to produce valid data if a strobe is given which demands that data. This is the output to input strobe specification.

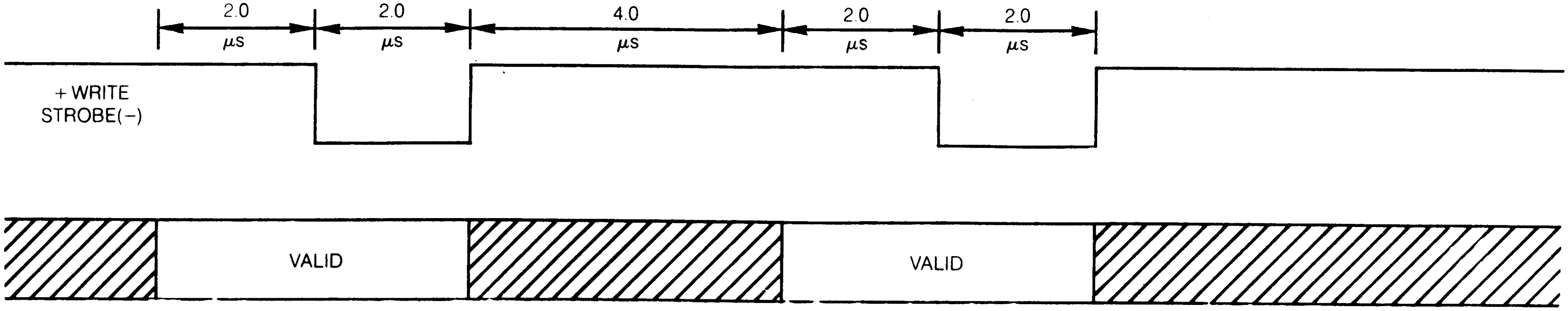
Figure 6.7-a shows normal input strobe timing. Figure 6.7-b shows multiple input strobe timing. Figure 6.7-c shows output to input strobe timing.

6.2.7 System Clock

The system clock signal is a 153.75 KHz square wave provided by the 55Ø1/55Ø2 as a convenient, accurate time base foruse by external devices. Its accuracy is plus or minus .Ø2%. It is not synchronized in any way to any other signals on the Bus.



4-a NORMAL OUTPUT



4-b MULTIPLE OUTPUT

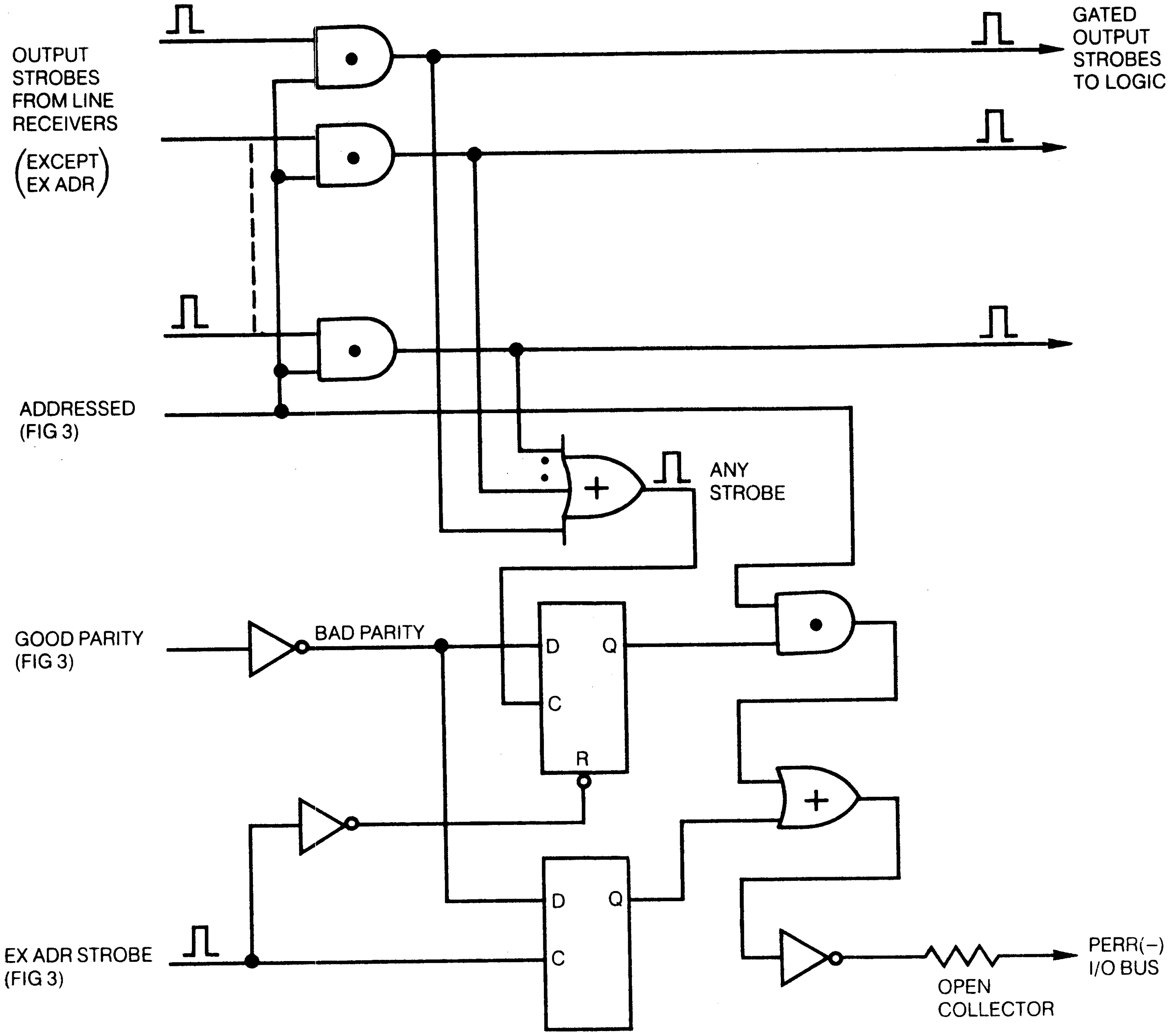


FIGURE 6.5 OUTPUT STROBE AND PARITY LOGIC

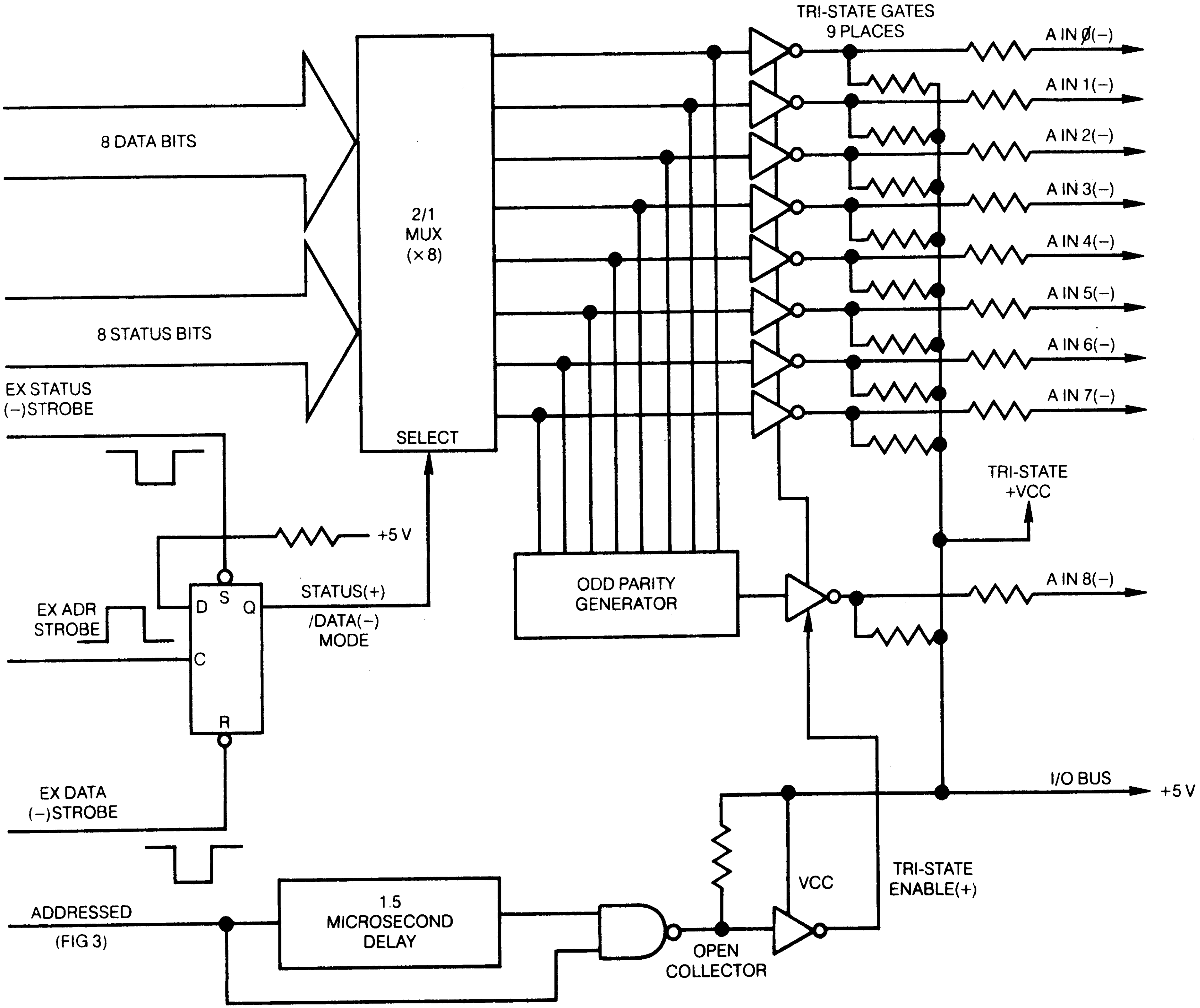


FIGURE 6.6 STATUS/DATA INPUT LOGIC

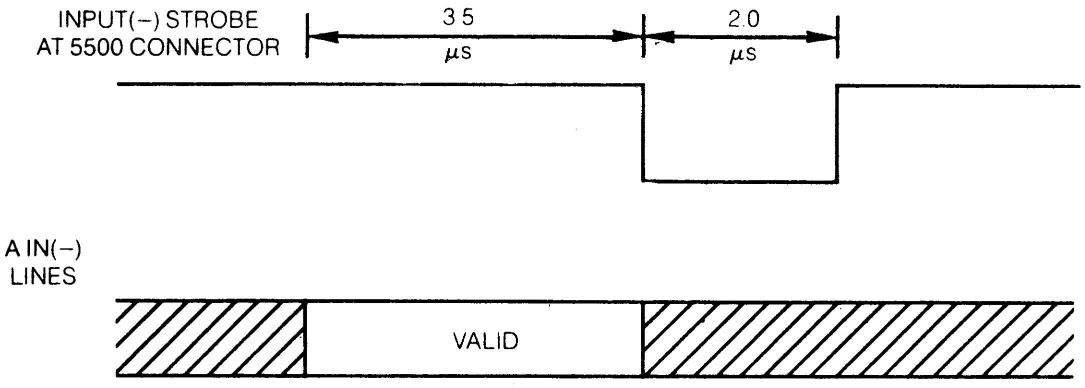


FIGURE 6.7-A NORMAL INPUT MODE TIMING

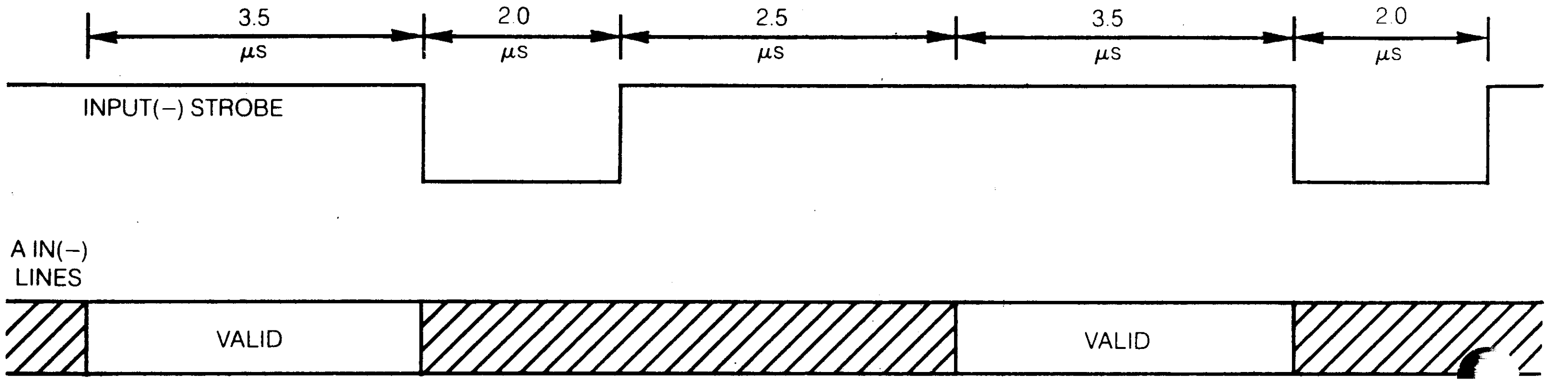


FIGURE 6.7-B MULTIPLE INPUT STROBE TIMING

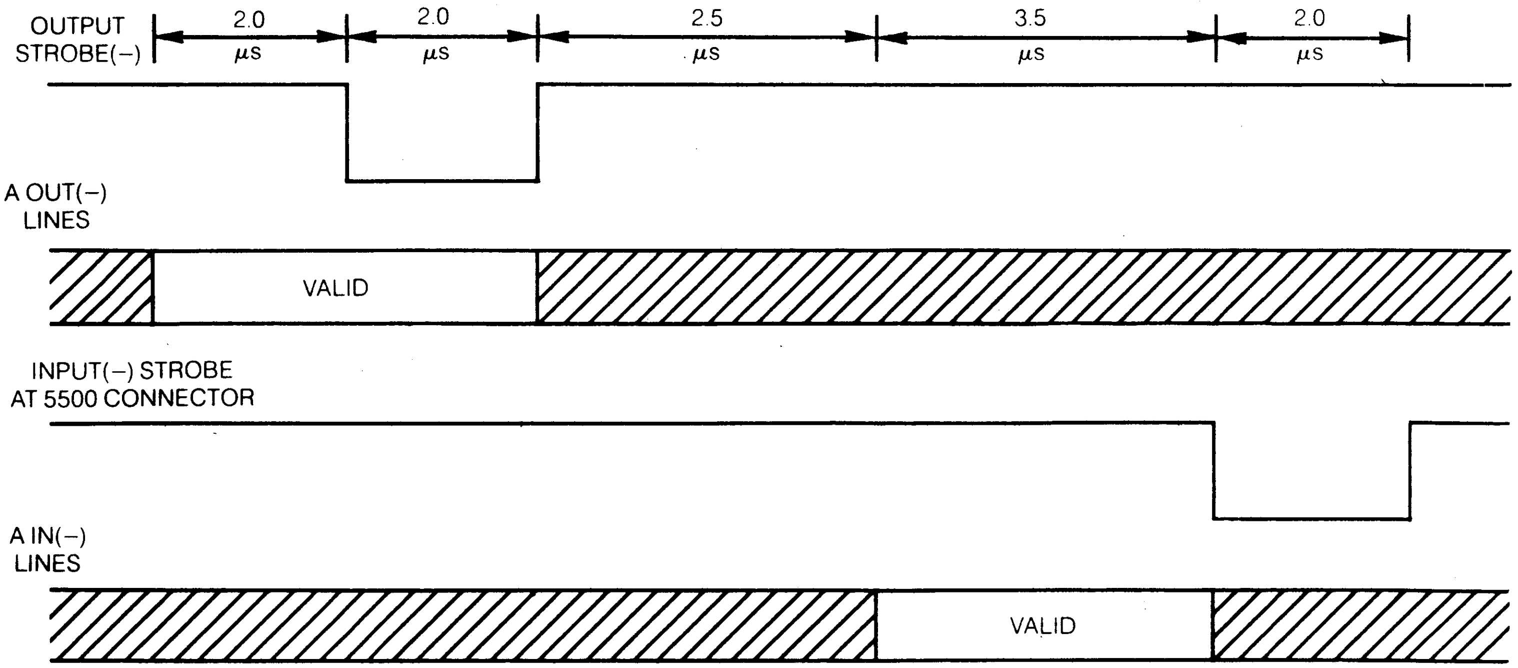


FIGURE 6.7-C MINIMUM OUTPUT
STROBE TO INPUT
STROBE TIMING

NOTES

DATAPOINT CORPORATION

The leader in dispersed data processing ™

DATAPOINT CORPORATION
9725 Datapoint Drive
San Antonio, Texas 78284
(512) 696-4520
Sales Offices in Principal Cities

MODEL CODE 60181-01