

**PRIMARY HARDWARE
MAINTENANCE MANUAL
Volume 1**

900-00020 Rev A



Valid Logic Systems Incorporated
1395 Charleston Road
Mountain View, CA 94043
(415) 940-4000

CPU BOARD USER'S MANUAL

VED-041582-2 Revision 5-14-82 (LCW)

02 Aug 84

Copyright 1984
Valid Logic Systems Incorporated

This document contains confidential proprietary information which is not to be disclosed to unauthorized persons without the written consent of an officer of Valid Logic Systems Incorporated.

The copyright notice appearing above is included to provide statutory protection in the event of unauthorized or unintentional public disclosure.

TABLE OF CONTENTS

Section	Page
LIST OF ILLUSTRATIONS	iii
LIST OF TABLES	v
1 INTRODUCTION	1-1
1.1 Purpose of Manual	1-1
1.2 General Description	1-1
2 M68000LB MICROPROCESSOR	2-1
2.1 General Features	2-1
2.2 M68000LB/CPU/Multibus Interface	2-2
3 LOCAL COMMANDS	3-1
3.1 Purpose	3-1
3.2 Local Commands/CPU Interface	3-1
3.3 Commands Generation	3-1
4 MEMORY	4-1
4.1 General	4-1
4.2 Segment and Page Map Interface	4-1
4.3 Segment and Page Map Strapping	4-4
4.4 Memory Management	4-4
5 CPU INTERRUPTS	5-1
5.1 Interrupts/CPU Interface	5-1
5.2 Interrupt Strapping	5-1
6 796 BUS	6-1
6.1 796 Bus Interface	6-1
6.2 Non-Standard Bits	6-4
6.3 796 Bus Description	6-4
6.3.1 Master-Slave Relationship	6-5
6.3.2 Bus Clock	6-5
6.3.3 Signal Interface	6-6
6.3.4 Data Transfers, Inhibit, and Interrupt Operations	6-10
6.3.4.1 Read Data	6-10
6.3.4.2 Write Data	6-11
6.3.4.3 Inhibit	6-11
6.3.4.4 Interrupts	6-11
6.3.5 Slave Interface	6-13
6.3.6 Address Decoding	6-13
6.3.7 Data Bus Drivers and Receivers	6-13

TABLE OF CONTENTS (Continued)

Section		Page
	6.3.8 Control Signal Logic	6-14
	6.3.9 Power Failures	6-14
7	796 BUS COMMANDS	7-1
	7.1 796 Bus Commands/CPU Interface	7-1
	7.2 796 Bus Commands Detailed Description	7-1
8	BUS ARBITER	8-1
	8.1 Bus Arbiter/CPU Interface	8-1
	8.2 Bus Arbiter Logic	8-2
	8.3 Bus Arbiter Strapping	8-5
9	CPU ADDRESSING	9-1
	9.1 Address Bus/CPU Interface	9-1
	9.2 Logical Address Space Structure	9-3
	9.2.1 User Mode	9-3
	9.2.2 Supervisor Mode	9-3
	9.3 Local Resources	9-5
10	REGISTERS	10-1
	10.1 Registers/CPU Interface	10-1
	10.2 Status Register	10-1
	10.3 Error Register	10-7
	10.4 Context Register	10-7
11	ERRORS AND TIMEOUT	11-1
	11.1 Errors and Timeout/CPU Interface	11-1
	11.2 Errors and Timeout Detailed Operation	11-3
12	RAM AND ROM MEMORIES	12-1
	12.1 Scratch Pad RAM/CPU Interface	12-1
	12.2 ROM/CPU Interface	12-1
13	INTERVAL TIMER	13-1
	13.1 Timer/CPU Interface	13-1
14	STATE GENERATION AND DTACK	14-1
	14.1 State Generation/CPU Interface	14-1
	14.2 State Signal Generation	14-3

TABLE OF CONTENTS (Continued)

Section		Page
15	FLOATING PCINT PROCESSOR	15-1
	15.1 Floating Point Processor/CPU Interface	15-1
16	SWITCHES	16-1
	16.1 Switches/CPU Interface	16-1
17	UARTS	17-1
	17.1 UARTS/CPU Interface	17-1
	17.2 UARTS Operation	17-1
	17.3 UARTS Strapping	17-3
18	CPU RESETS	18-1
	18.1 Resets/CPU Interface	18-1
	18.2 Resets Detailed Operation	18-1
	18.2.1 Power On Reset	18-3
	18.2.2 Reset From 68000	18-3
	18.2.3 Reset From Bus	18-3
19	CLOCK GENERATOR	19-1
	19.1 Clock Generator/CPU Interface	19-1
20	MAINTENANCE PORT	20-1
	20.1 Maintenance Port/CPU Interface	20-1

LIST OF ILLUSTRATIONS

Number	Title	Page
1-1	CPU Block Diagram	1-3
2-1	M68000LB Microprocessor/CPU Interface	2-3
3-1	Local Commands/CPU Interface	3-2
3-2	Local Commands Detailed Block Diagram	3-3
4-1	Segment and Page Map/CPU Interface	4-2
4-2	Segment Map Detailed Block Diagram	4-3
4-3	Segment and Page Map Strapping	4-5
4-4	Segment and Page Map Management	4-7

LIST OF ILLUSTRATIONS (Continued)

Number	Title	Page
5-1	Interrupts/CPU Interface	5-2
5-2	Interrupts Strapping	5-3
6-1	Multibus/CPU Interface	6-2
6-2	Detailed Multibus Interface	6-3
6-3	8/16-Bit Data Drivers/Data Transfers	6-12
7-1	796 Bus Commands/CPU Interface	7-2
7-2	796 Bus Commands Logic	7-3
8-1	Bus Arbiter/CPU Interface	8-3
8-2	Bus Arbiter Logic	8-4
9-1	Address Bus/CPU Interface	9-4
10-1	Status Register/CPU Interface	10-2
10-2	Status Register Fields	10-6
10-3	Error Register/CPU Interface	10-8
10-4	Error Register Fields	10-9
10-5	Context Register/CPU Interface	10-11
10-6	Context Register Fields	10-12
11-1	Errors and Timeout/CPU Interface	11-2
11-2	Errors and Timeout Logic	11-4
12-1	Scratch Pad RAM/CPU Interface	12-2
12-2	ROM/CPU Interface	12-3
13-1	Interval Timer/CPU Interface	13-2
14-1	State Generation/CPU Interface	14-2
15-1	Floating Point Processor/CPU Interface	15-2
16-1	Switches/CPU Interface	16-2
17-1	UARTS/CPU Interface	17-2
17-2	Host RS232 Interface Strapping	17-4
18-1	Resets/CPU Interface	18-2
18-2	(Example A). POWER ON RESET (INITIAL STATE)	18-5
18-2	(Example B). POWER ON RESET (RESET* TO RESET 68000)	18-6
18-2	(Example C). RESET FROM 68000 (RAW RESET*, FIRST CLOCK)	18-7
18-2	(Example D). RESET FROM 68000 (RAW RESET* , SECOND CLOCK)	18-8
18-2	(Example E). RESET FROM MULTIBUS (RESET*),	18-9
19-1	Clock Generator/CPU Interface	19-2
20-1	Maintenance Port/CPU Interface	20-4

LIST OF TABLES

Number	Title	Page
2-1	M68000LB Interface Signals	2-5
6-1	796 Bus Interface Signals (P1)	6-6
6-2	796 Bus Interface Signals (P2)	6-9
8-1	Bus Arbiter Interface Signals	8-1
9-1	CPU Addressing Features	9-1
10-1	Status Register Fields	10-3
10-2	Error Register Fields	10-7
10-3	Context Register Fields	10-10
20-1	Maintenance Port Interface Signals	20-1

SECTION 1 INTRODUCTION

1.1 Purpose of Manual

This manual provides a functional overview of the M68000LB CPU board. Included are descriptions of all functions contained on the CPU board as well as a block diagram of the overall board. The description of each function consists of explaining its interface with other functional areas within the CPU and a detailed description of the operation of that particular function. The overall block diagram shows the complete interconnection between all CPU functions described in this manual.

1.2 General Description

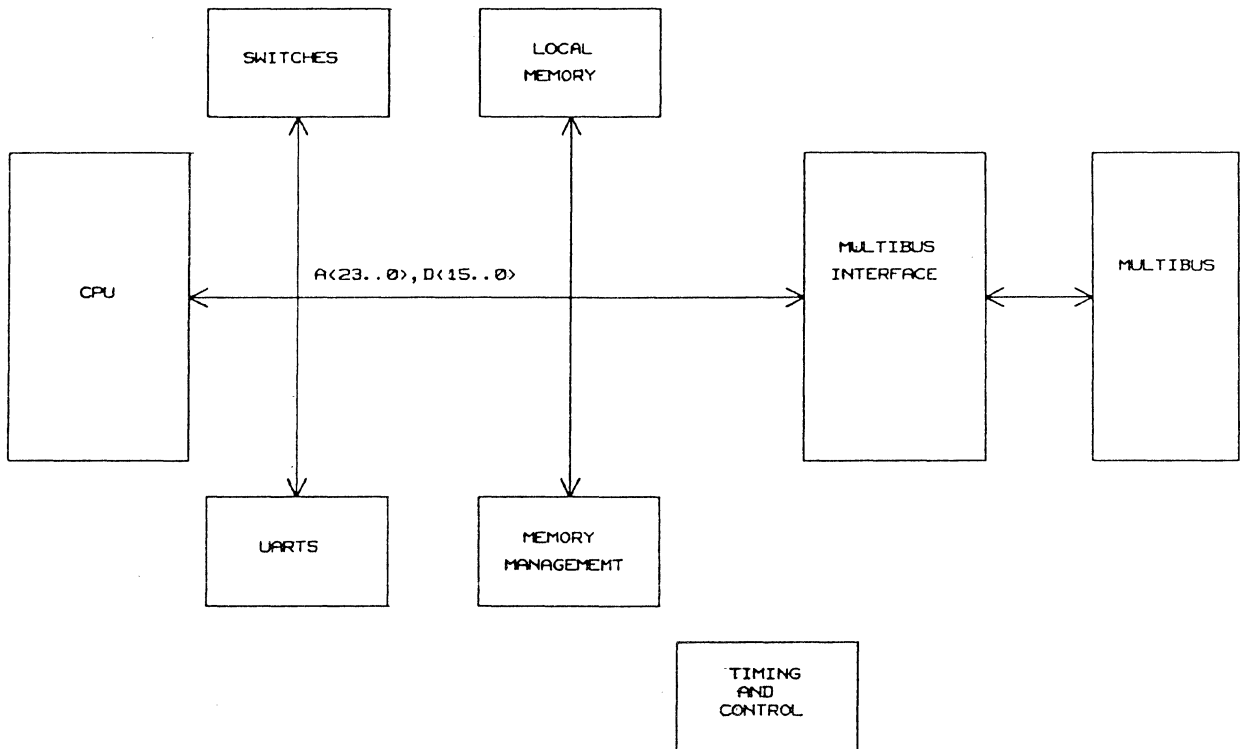
The CPU board provides a complete interface between the Motorola M68000LB Microprocessor running at an 8MHz rate and the IEEE-796 bus (Intel Multibus) referred to as the 796 bus in the remainder of this document. At the same time it provides the functions need in the CPU section of a high performance, large-memory, 796 bus based computer system. The board addresses the full 16 Mbyte 796 bus address space through a memory map which offers both relocation and protection for multiple simultaneous users. The board also provides fast serial bus arbitration allowing existence in a system with a large number of other bus masters. The CPU board consists of the following features:

R M68000LB is a registered trademark of Motorola, Inc.

R Multibus is a registered trademark of Intel.

- o Microprocessor running at 8MHz
- o Complete interface to a 16Mbyte 796 bus
- o Seven interrupt priority levels
- o Memory management utilizing segmentation and paging
- o Onboard status, error, and context registers
- o Eight software readable switches for configuration selection
- o Sockets for up to 16Kbyte of local EPROM
- o 2Kbyte of local scratch-pad RAM
- o Three programmable 16-bit timers
- o Two programmable UARTS, one with full modem control
- o Full MACSBUG compatibility, including transparent mode
- o Optional IEEE format floating point processor.
- o Maintenance port for testing and diagnostic operations.

The CPU board consists of the functional areas shown in Figure 1-1. Each of these areas are described in the sections that follow. Interfaces to and from the CPU board include the UARTS ports, 796 bus, and maintenance connector. Certain strapping options are also provided on the board as explained in each functional description as applicable.



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1982

TITLE:
CPU BOARD

ABBREV:

EXPR:

VERSION:

SECTION 2

M68000LB MICROPROCESSOR

2.1 General Features

The M68000LB microprocessor is a high-performance, programmable, 16-bit processor containing a 32-bit architecture with the features listed below.

- o 8MHz operation
- o 16Mbyte direct addressing range
- o Eight 32-bit data registers
- o Seven 32-bit address registers
- o 56 instruction types
- o Five data types
- o 14 addressing modes
- o CPU driven at 8MHz

In addition, it incorporates such features as multi-level, vectored interrupts, privilege states, illegal instruction policing and bus-cycle abort.

The detailed operation of the M68000LB is discussed in publications supplied by the manufacturer of the device. Consult these as required for detailed information as to the specific features and operation of the M68000LB.

2.2 M68000LB/CPU/Multibus Interface

The interface of the CPU to the 796 bus as shown in Figure 2-1 are listed as follows:

- o Address Latches
- o State generation and data acknowledge (DTACK)
- o Local commands
- o UARTS
- o Timer and FPU
- o RAM and ROM
- o SMAP and PMAP
- o CPU interrupts
- o Errors and timeout
- o 796 bus interface
- o Clock generator
- o Test and maintenance interface
- o Context and status registers

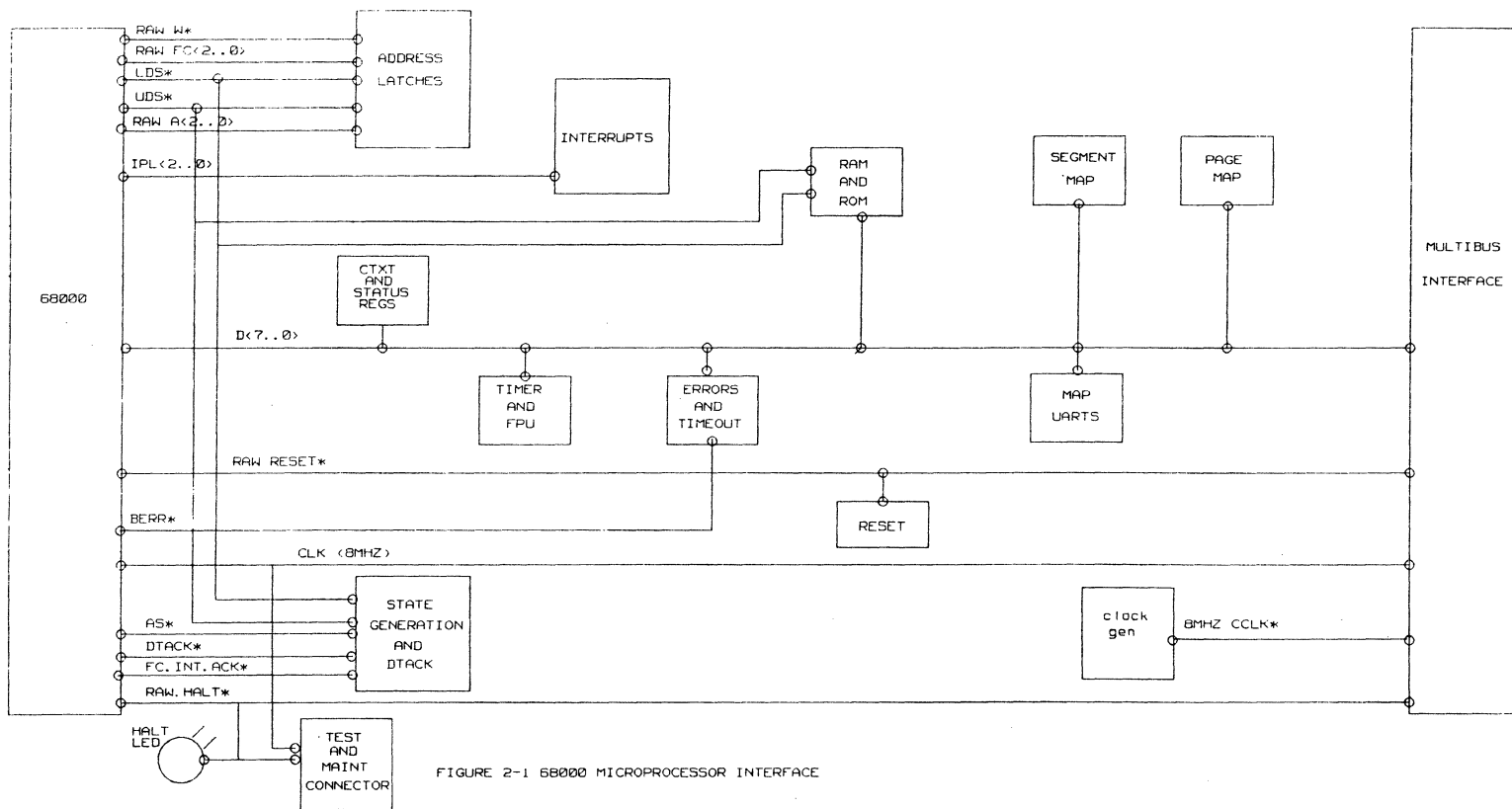


FIGURE 2-1 68000 MICROPROCESSOR INTERFACE

THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT © VALID 1982

TITLE:

ABBREV:

EXPR:

VERSION:

Table 2-1 lists the name, category, and function of each signal to and from the M68000LB.

The M68000LB supplies address bits (RAW.A<23..1>), raw write (RAW.W*), function code (RAW.FC<2..0>), and upper and lower data strobes (LDS* and UDS*) to the address latches. The latches in turn provide addressing, read/write commands, function codes, and upper and lower data byte selection as described in the addressing section of this manual.

LDS* and UDS* data strobes are supplied to the RAM and ROM memories and the state generation and DTACK logic. The strobes provide upper and lower byte selection for the operation of the RAM scratch pad memory. The strobes also generate data strobe outputs from the state generation logic to the CPU board. The function codes, all true, plus AS* from the M68000LB generate FC.INT.ACK* from the state generation logic to the M68000LB. Function code bits FC<2..1> are also supplied to the errors and timeout logic as inputs to the S.MAP ERR decoder.

The M68000LB data bus provides an interface among the various functions on the CPU board. These are the context and status registers, timer, FPU, RAM and ROM, errors and timeout, segment and page maps, UARTS, and 796 bus interface. Data is exchanged between these functions and the 796 bus as controlled by the program (user or supervisor mode) and in turn as selected by the address bus, read/write selection, function code, and upper and lower data strobes.

Table 2-1. M68000LB Interface Signals

SIGNAL	CATEGORY	FUNCTION
RAW A<23..1>	Address Data Lines	Provides address for bus operations during all cycles except interrupt modes (refer to CPU Interrupts, Section 5).
D<15..0>	Data Lines	Bi-directional data line for word or byte length data transfers (refer to CPU Interrupts, Section 5).
AS* (Address Strobe)	Asynchronous Bus Control	Indicates a valid address is on the M68000LB address bus.
RAW W* (Read/Write*)	Asynchronous Bus Control	Defines the data bus transfer as a read or write cycle. Also operates in conjunction with the UDS* and LDS* signals.
UDS* (Upper Data Strobe)	Asynchronous Bus Control	Controls upper byte of data lines D<15...8> depending on condition of RAW W*.
LDS* (Lower Data Strobe)	Asynchronous Bus Control	Controls lower byte of data lines (D<7..0>) depending on condition of RAW W*.

Table 2-1. M6800LB Interface Signals (Continued)

SIGNAL	CATEGORY	FUNCTION
DTACK (Data Transfer Acknowledge)	Asynchronous Bus Control	Indicates data transfer is completed.
RAW FC<2..0> (Function Code)	Processor Status	Indicate the state (user or supervisor) and the cycle type currently being executed. Function code outputs are valid only when AS* is active.
IPL<2..0>)* (Interrupt Priority Level)	Interrupt Control	Indicate the encoded priority level of the device requesting an interrupt (refer to CPU Interrupts, Section 5).
FC.INT.ACK.* (Valid Function Code)	Peripheral Control	Indicates M6800LB should use automatic vectoring for an interrupt.

Table 2-1. M68000LB Interface Signals (Continued)

SIGNAL	CATEGORY	FUNCTION
RAW RESET* (Reset)	System Control	Bidirectional signal resets the M68000LB in response to an external reset signal. An externally generated reset (from CPU) causes all external devices to be reset without affecting the state of the processor. A total system reset as a result of a HALT or RESET, causes M68000LB and all other devices to be reset.
RAW HALT* (Halt)	System Control	Bi-directional signal. When driven by M68000LB, indicates to external devices that processor has stopped. When driven by external device, causes processor to stop at completion of current bus cycle.

Table 2-1. M68000LB Interface Signals (Continued)

SIGNAL	CATEGORY	FUNCTION
BERR* (Bus Error)	System Control	Informs processor of problem with current cycle being executed. Works with RAW HALT* to determine if exception processing should be performed or current bus cycle retried.
CLK		Provides 8MHz clock signal to operate the M68000LB.

The 8MHz clock is generated by circuits on the CPU board and supplied to the 796 bus (see Clock Generator, Section 19) or can be received from the 796 bus from another source. Interrupts can originate either on the CPU board or the 796 bus. These result in a three bit code used by the M68000LB to select one of seven strappable interrupts. A bus error (BERR*) is applied to the M68000LB after a timeout, an address bus error from the command logic, or a segment or page error. Reset (RAW RESET*) is a bi-directional line between the 796 bus and the M68000LB. A reset from either source is provided to the CPU reset logic. The RAW.HALT line is a bi-directional signal used to halt M68000LB operations. The line illuminates a red light emitting diode (HALT LED) when activated. The reset and halt lines are also provided to the test and maintenance connector for fault isolation purposes.

SECTION 3 LOCAL COMMANDS

3.1 Purpose

The purpose of the local commands is to generate command strobes for the CPU. The commands logic is odd-byte addressed and provides on board resources, map, and 796 bus access. The local commands interface within the CPU is shown in Figure 3-1. A detailed diagram of the commands logic is shown in Figure 3-2.

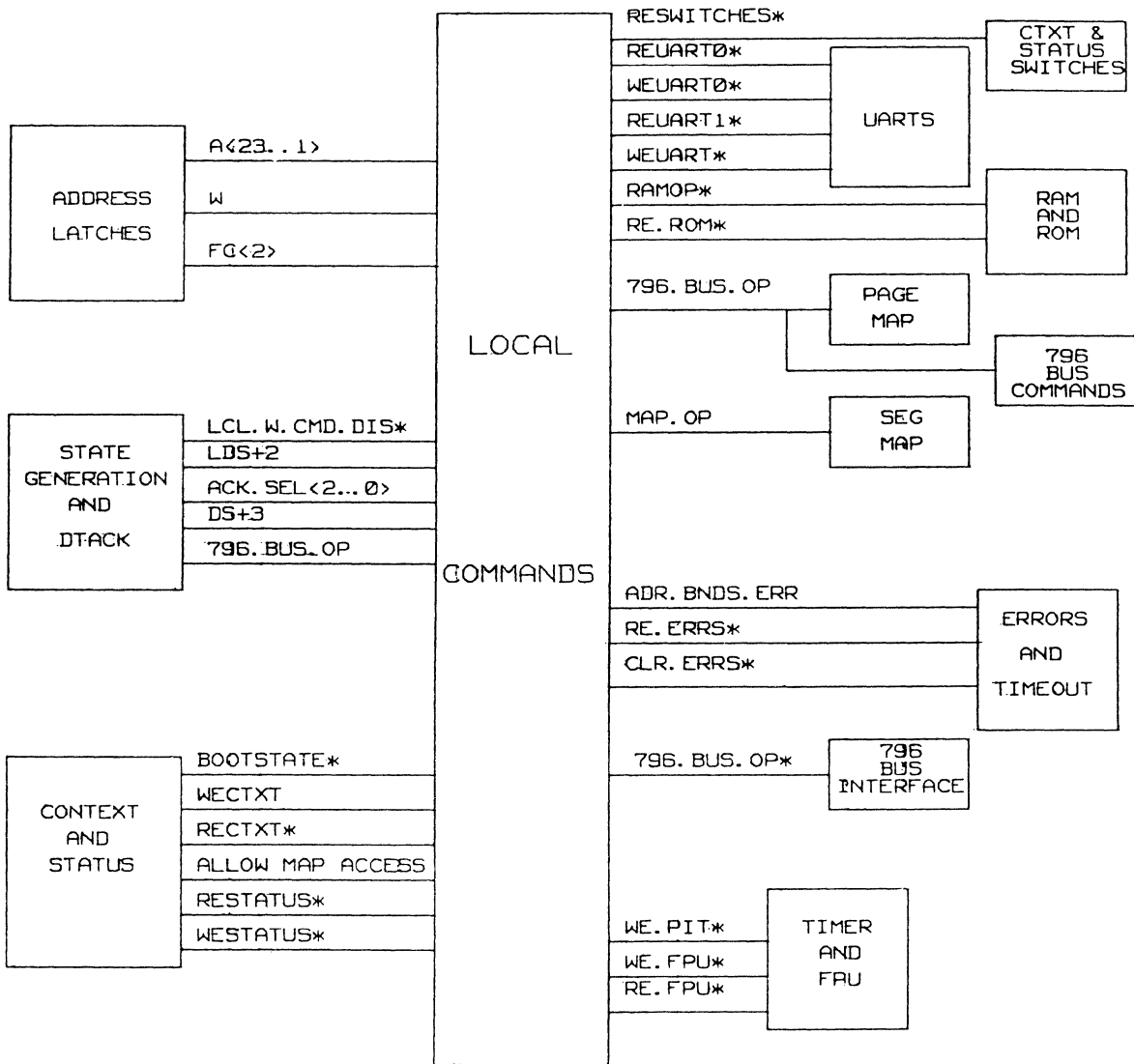
3.2 Local Commands/CPU Interface

The commands interface within the CPU, as shown in Figure 3-1, consists of receiving addressing, timing, and control signals from the address latches, state generation and DTACK, and context and status registers. The commands logic then provides command strobes to the various CPU functions as shown.

3.3 Commands Generation

The local commands logic consists of two programmable PROMs and two decoders. The PROMs (2048 x 4 and 512 x 8) are addressed from the CPU address latches over lines A<23..16> and A<15..11> respectively. Addressing to the decoders is over lines A<13..11>. In addition to the address lines, the PROMs are also addressed during the boot state controlled by one of the map access modes over function code line (FC<2>). The 512 x 8 PROM is addressed from the 2048 x 4 PROM, and the read.write line from the address latches. Commands generated from the PROMs include the following:

- o 796.BUS.OP* to the 796 bus interface
- o MAP.OP to the segment map
- o ADR BNDS.ERR to the errors and timeout logic
- o RAM and ROM commands for local operation
- o Acknowledge lines to the state generation and DTACK
- o Read/write commands to the command decoders



LOCAL COMMANDS/CPU INTERFACE FIG. 3-1

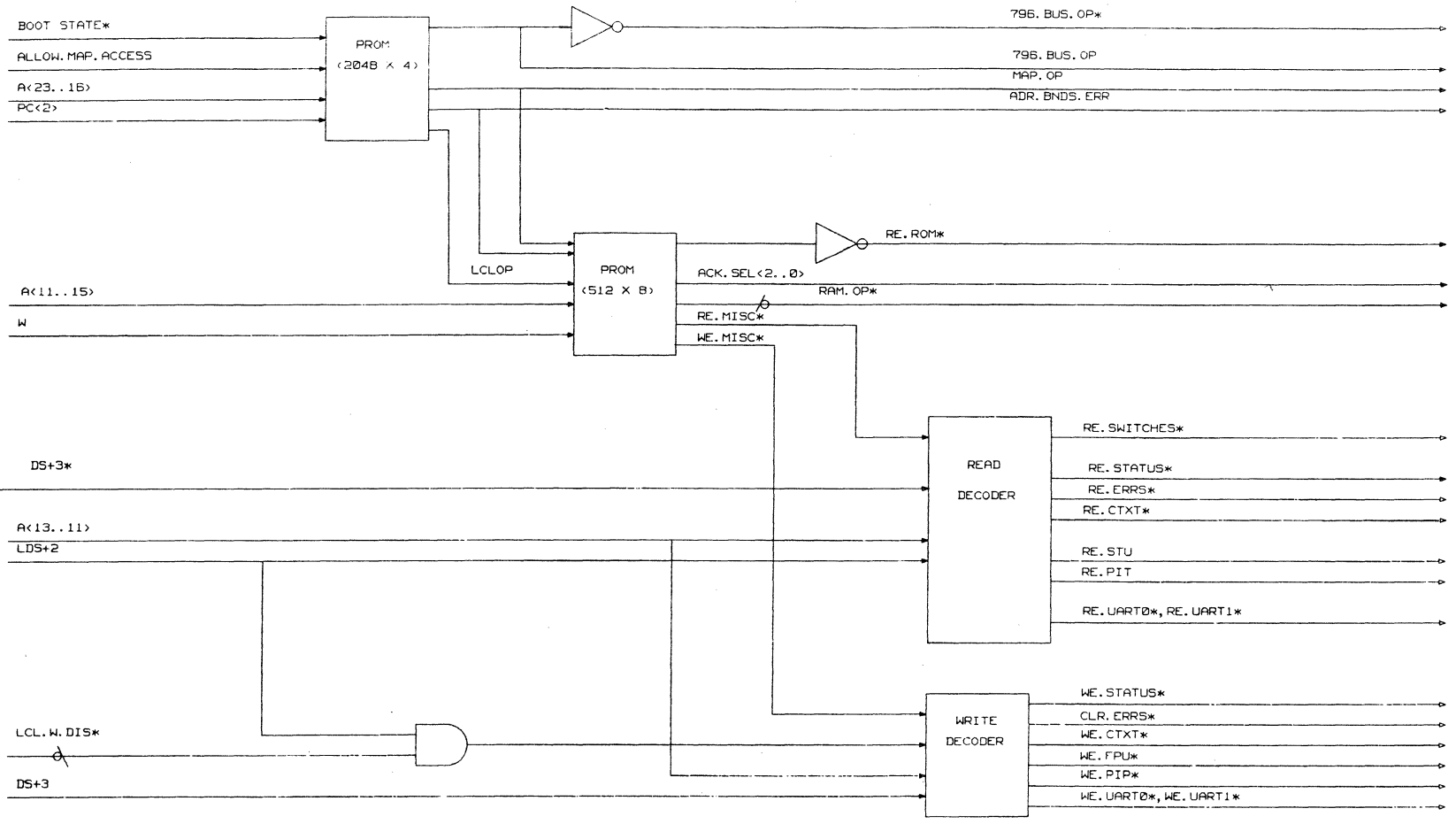
THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1982

TITLE: FIG 3-1
LOCAL COMMANDS/CPU INTERFACE

ABBREV:

EXPR:

VERSION:



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1982



TITLE: LOCAL COMMANDS DETAILED BLOCK DIAGRAM
 EXPR:

ABBREV: LOC COM
 VERSION: 6/14/82

The command decoders are operated by the data strobe (DS) and lower data strobe selection (LDS) lines from the state generation and DTACK logic. The write decoder can be disabled by the local write disable line (LCL.W.DIS*) generated by the state generation logic. The decoders operate in either a read or write mode. The read decoder is selected by a read (RE.MISC*) command from the 512 x 8 PROM, coupled with a true lower data strobe (LSD+2). The final read outputs are placed on the CPU lines as the DS+3* pulse occurs to the read decoder. For a write operation, a WE.MISC* line from the 512 x 8 PROM coupled with the LDS+2 selects the write decoder and the DS+3 pulse from the state generation logic places the write commands on the CPU lines.

SECTION 4

MEMORY

4.1 General

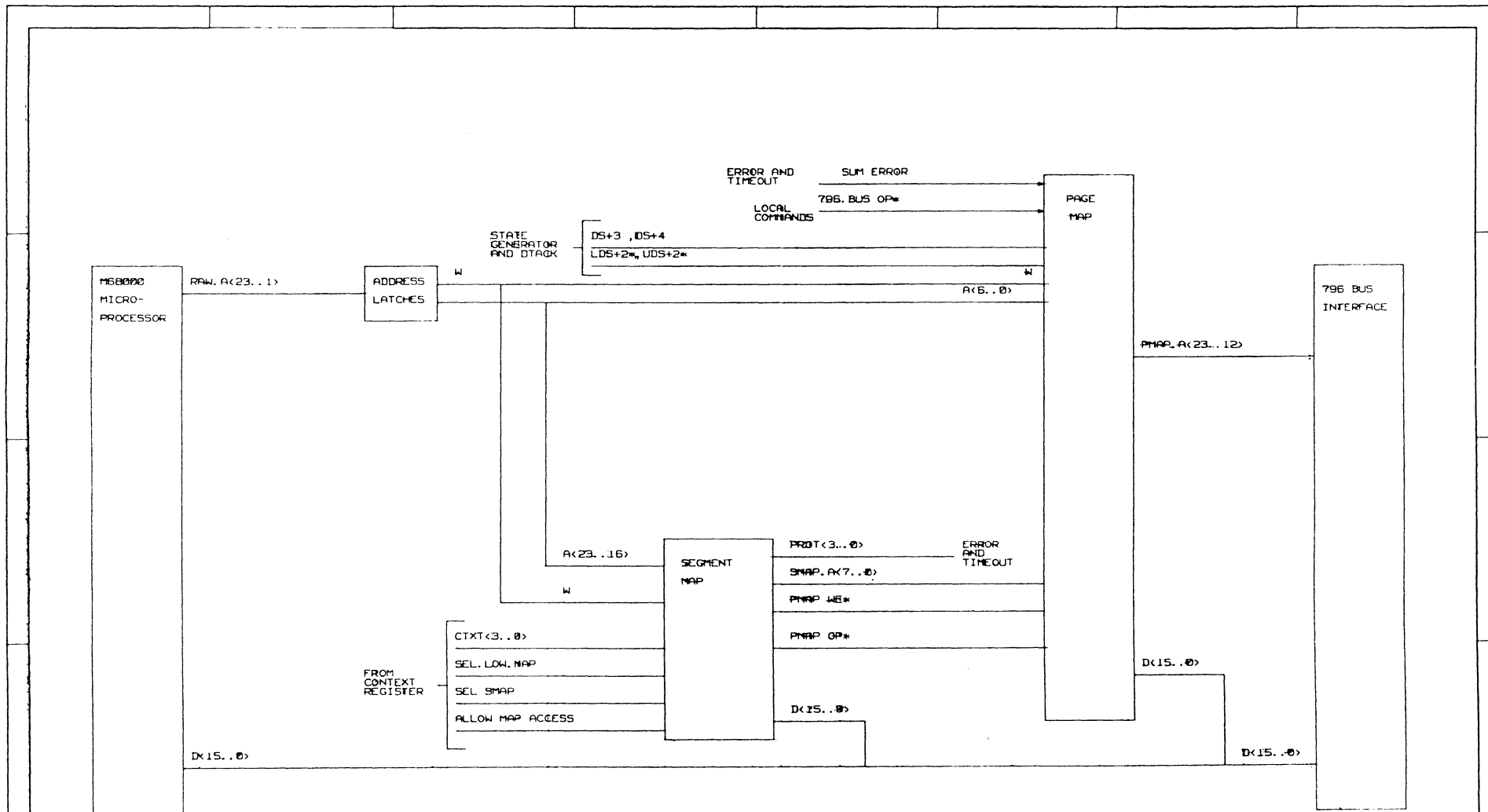
The CPU memory management consists of the M68000LB utilizing both segmentation and page mapping. The functions of the CPU board along with associated interfacing involved in the memory management task are shown in Figure 4-1. Shown are the segment map, page map, address, data, and select and control lines that interface with the various functions of the CPU board including the M68000LB, address latches, and 796 bus interface.

4.2 Segment and Page Map Interface

Addressing and read/write commands for the segment map are received from the M68000LB via the address latches. Map selection and upper and lower data strobe (UDS+2* and LDS+2*) are supplied by the context register and state generator and DTACK respectively. Segment map addresses as well as control and selection signals are supplied to the page map. Segment map error signals (PROT<3..0>) to the errors and timeout logic are lines that provide inputs for the generation of the SMAP.ERR signal to the error register (see Registers, Section 10). An interface to the bi-directional data lines (D<15..0>) is also provided for reading and writing of segment map data as described in paragraphs that follow.

The page map supplies map addresses to the 796 bus (PMAP.A<23..12>) and interfaces with the data bus for read/write purposes. The map receives addresses and read/write commands (W) from the address latches and data strobe (DS+3 and DS+4), selection (796.BUS.OP*), upper and lower byte selection (LDS+2* and UDS+2), and error signals (SUM.ERROR) from the stage generation and DTACT local commands, and error and timeout logic.

A detailed block diagram of the segment and page map is provided in Figure 4-2. In addition to the interface diagram of the segment and page map described in the previous paragraphs, the detailed diagram shows gating of the



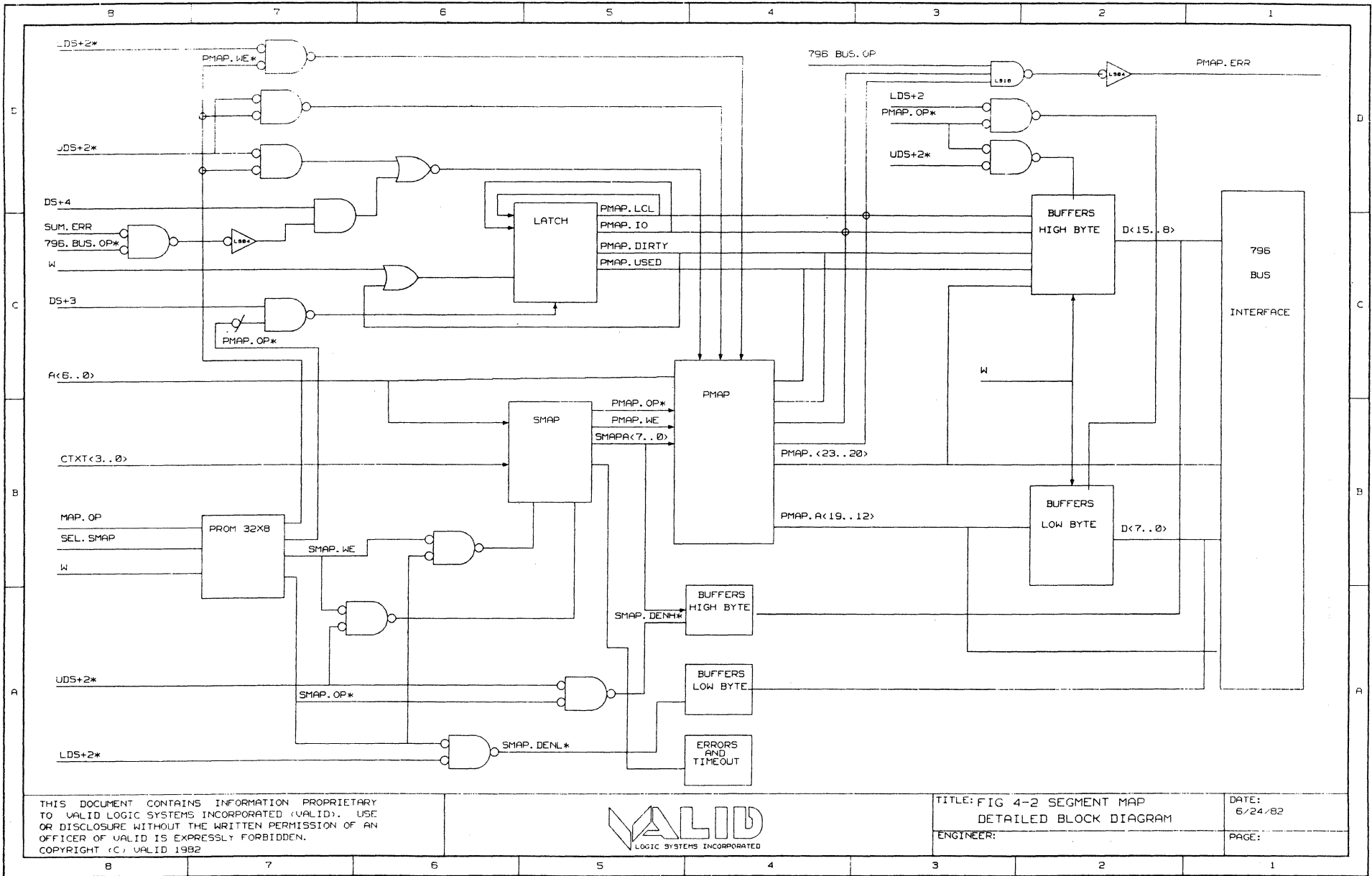
THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1982

TITLE: FIG 4-1 SEGMENT AND PAGE MAP/CPU INTERFACE

EXPR:

ABBREV:

VERSION:



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1982



TITLE: FIG 4-2 SEGMENT MAP
 DETAILED BLOCK DIAGRAM
 ENGINEER:

DATE:
 6/24/82
 PAGE:

control and selection signals and bi-directional buffers used for reading and writing of data to and from the respective maps.

The selection logic shown in the case of both the segment and page map provides lower and upper byte selection based on the upper and lower data strobes and segment and page map selection signals. Also shown are the PMAP.USED, PMAP.DIRTY, PMAP, IO and PMAP LCL which are discussed in the memory management paragraph.

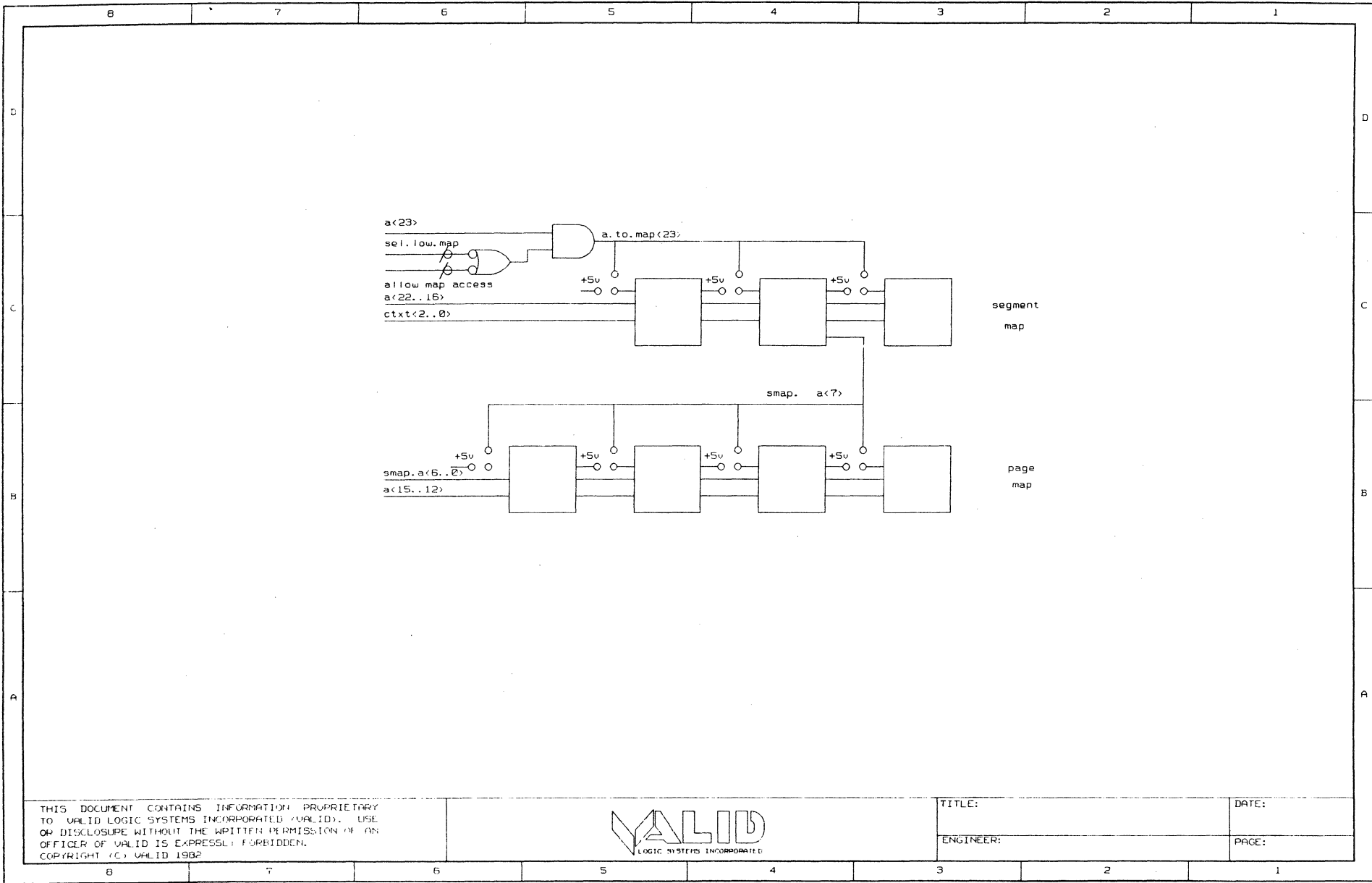
4.3 Segment and Page Map Strapping

The segment and page maps can be strapped to accept larger mapping RAMs. In that case, each of 16 processes can access a separate 16Mbyte logical address space containing 256 segments each of 16-4Kbyte pages. A diagram of the strapping configuration is shown in Figure 4-3. The strapping to a larger RAM mapping system involves using additional selection signals and address bit A<23> as shown instead of the +5V connection used for the VCC input of the smaller map. The segment map also provides the SMAP.A<7> bit for additional addressing space to the page map increased RAM size. This is also strapped to increase the addressing to the larger page map RAMs. The +5V input is then reconnected to the VCC input of the larger capacity RAMs.

4.4 Memory Management

The CPU board includes memory management sufficient to support a full capability multi-user operating system such as Unix running with 4Mbyte physical memory (16Mbyte using the 4K-map configuration). Each process can access to a separate 4Mbyte logical address space consisting of 64 pages of 64Kbyte each, which can be mapped anywhere within the 16Mbyte physical address space of the 796 bus.

As many as 16 processes can be mapped simultaneously. The low-order four bits of the context register (PID<3..0>) contains the process ID of the process currently being executed. In cases where there are more than 16 processes to be mapped, a process ID can be reused by reloading the appropriate section of the map.



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1982



TITLE:	DATE:
ENGINEER:	PAGE:

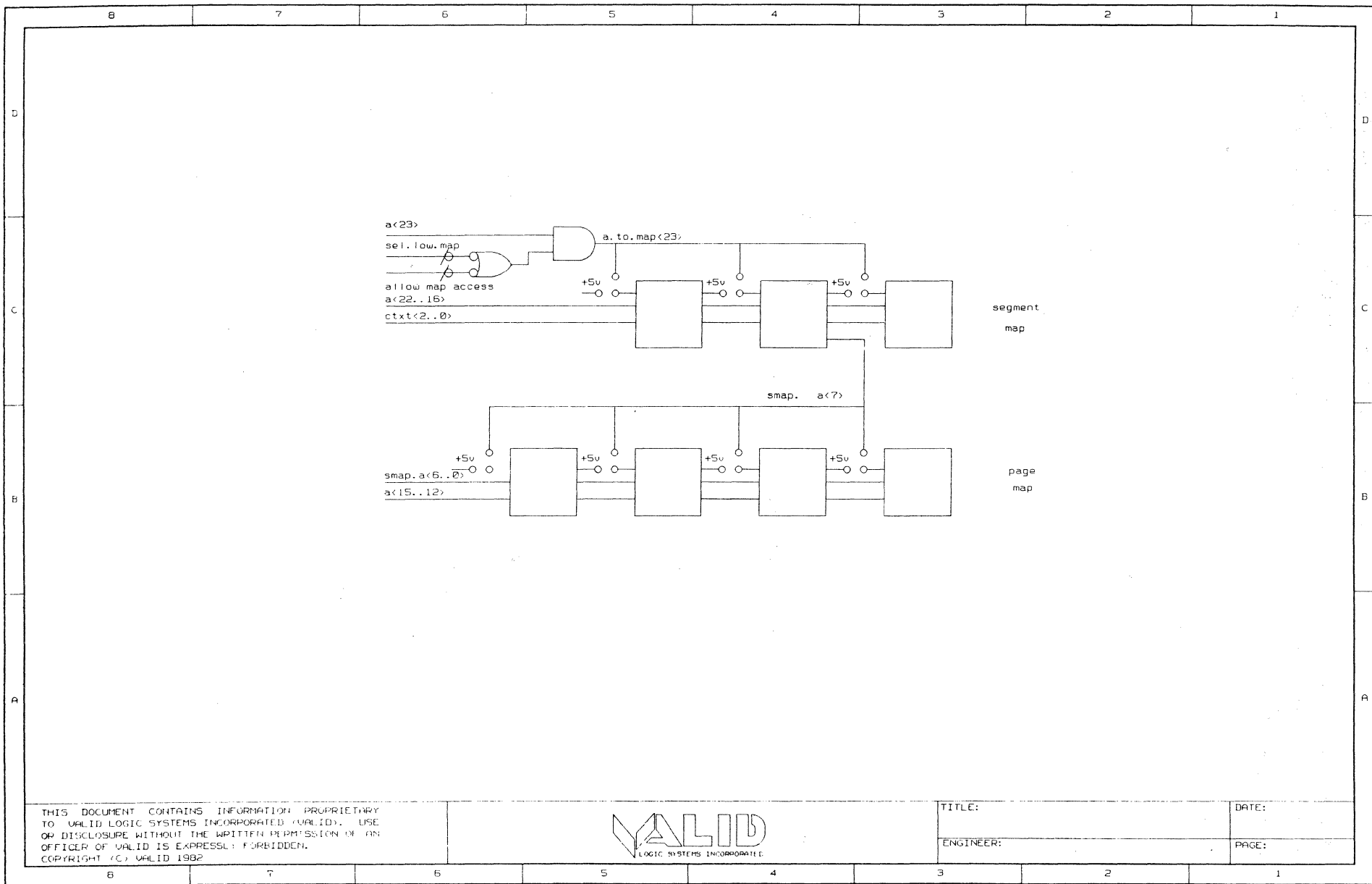
As shown in Figures 4-1, 4-2, and 4-4, the segment table contains 1K entries and is addressed by the context 4-bit process ID concatenated with six logical address bits A<21..16> from the M68000LB. A segment table entry so addressed has the following format:

FIELD	NAME	INTERPRETATION
<15..12>	Undefined	
<11..8>	ACCESS.MODE<3..0>	Per-segment legal access modes
<7..6>	Reserved	
<5..0>	A<5..0>	High order page table address

The high-order four bits of a segment table entry are undefined and can be neither read or written. The ACCESS.MODES<3..0> define the legal access modes for the segment which has the following structure:

FIELD	NAME	INTERPRETATION
<3>	USER	User mode access permitted
<2>	WRITE	Writes permitted
<1>	READ	Data space reads permitted
<0>	EXECUTE	Program space reads permitted

These bits have independent meanings. For example, a segment that is intended to have "supervisor-read or execute" access has ACCESS.MODES=3. Also note that all PC relative accesses are classified as "program space" accesses.



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1982



TITLE:	DATE:
ENGINEER:	PAGE:

The page map table shown in Figure 4-4 contains 1K 16-bit entries and is addressed by concatenating the 6-bit field SMAP.A<5..0> from a segment table entry with the four logical address bits A<15..12> from the M68000LB. Each accessible segment requires 16 contiguous page table entries. However, not necessarily 16 pages of physical memory since page table entries can be marked "not present". A page table entry addressed in this manner has the following format:

FIELD	NAME	INTERPRETATION
<15..12>	PAGE.CONTROL	Per-page control bits
<11..0>	A<11..0>	High-order physical address

Field A<11..0> is concatenated above the low-order 12 address bits from the M68000LB to form a full 24-bit 796 bus address. The PAGE CONTROL <3..0> field has the following format:

FIELD	NAME	INTERPRETATION
<3>	USED	This page was accessed
<2>	DIRTY	This page was written
<1>	LOCAL	Accesses to this page are not permitted
<0>	IO	I/O address

The USED and DIRTY fields are updated by hardware when the page is accessed. The USED is set whenever the page is accessed and DIRTY is set whenever the page is written. The exception is USED and DIRTY are not changed if the access is illegal. If LOCAL = 1, then the access is not legal for example, the page does not exist in physical memory. Otherwise, if IO = 1, the access

is to 796 bus I/O space, and if IO = 0, then it is to 796 bus memory space. If either the segment or page table strobes are sent, USED and DIRTY are not updated, and the M68000LB is forced to execute a bus-error trap. The CPU error register captures the exact cause of the trap.

All 16 bits of a page table entry can be read or written using byte or word operations, as can all 12 defined bits of a segment table entry. To access either a segment table entry or a page table entry, the context register must be set up appropriately. The high-order address bit (A<23>) of the M68000LB must be set and the low-order 23 bits must be set as if the access were to some location within the segment or page concerned. The segment map can be accessed directly, but the page map is accessed by first setting up a location in the segment map and then addressing the page map through the segment map.

Since the high-order M68000LB address bit distinguishes map references, the M68000LB can do map operations interleaved with 796 bus operations, for example, copying from main 796 bus memory into the map. References with A<23> set are to the map, and references with A<23> clear are to the 796 bus.

Note that supervisor and user share a single logical-address space, allowing the supervisor direct access to user data structures.

If the map is strapped to accept larger mapping RAMs, each of 16 processes can access a separate 16Mbyte logical-address space containing 256 segments each of 16-4Kbyte pages.

SECTION 5

CPU INTERRUPTS

5.1 Interrupts/CPU Interface

The CPU interrupts interface with the CPU as shown in Figure 5-1. Interrupts are received from the 796 bus interface, UARTS, FPU, and timer, and test and maintenance connector. The interrupts are strappable as described in the paragraphs that follow. The resulting interrupts to the M68000LB (3-bit code) are gated and clocked by EN.INT* and CLK* signals received from the status register and clock generator respectively.

The interrupts operate asynchronously in auto-vector mode to deliver the three bits (IPL<2..0>*) to the M68000LB. The bits track to one of seven fixed addresses. During interrupt cycles, the M68000LB address lines A<3..1> provide information about what level interrupt is being serviced while address lines A<23..4> are all set to logic high. During an interrupt acknowledge cycle, an external device supplies the vector number on data lines D<7..0>. Incoming interrupts are latched at the falling edge of the CLK* pulse. During the time between clock pulses, the latches provide stability and settling time between the asynchronous priority inputs and the encoder. The 796 bus interrupts are further described in the discussion of the 796 bus in the sections that follow.

5.2 Interrupt Strapping

Any of the interrupts (19 total) can be strapped in any priority level to the seven inputs of the latches as shown in Figure 5-2. The seven latch outputs are applied one-for-one to the priority encoder inputs and correspond to the priority scheme of the 796 bus (INT<7..0>) inputs as follows:

Line 7 = highest priority
Line 1 = lowest priority
Line 0 = no priority

STATUS REGISTER

EN. INT*

CLOCK GENERATOR

CLK*

796BUS INTERFACE

INT<7..0>* /8

UARTS

UART0 TXRDY*

UART0 RXRDY*

UART1 TXRDY*

UART1 RXRDY*

UART0 DSCHG*

FPU AND TIMER

FPU. SVREQ

PIT<2..0> /3

TEST AND MAINT. CONNECTOR

TST. INT

CPU INTERRUPTS

M68000 MICROPROCESSOR

IPL<2..0>* /3

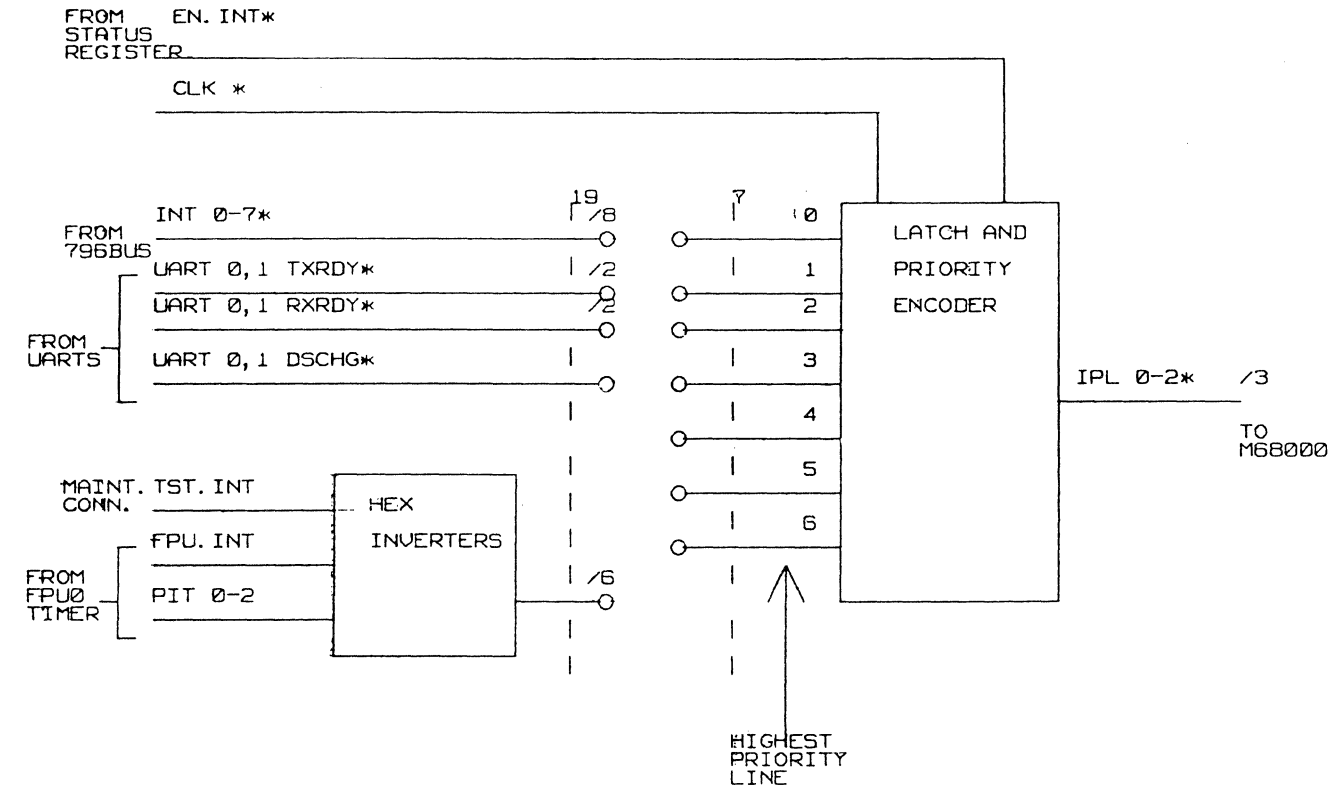
THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1982

TITLE: FIG 5-1 INTERRUPTS/CPU INTERFACE

ABBREV:

EXPR:

VERSION:



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1982

TITLE: FIG 5-2	ABBREV:
INTERRUPT STRAPPING	
EXPR:	VERSION:

SECTION 6

796 BUS

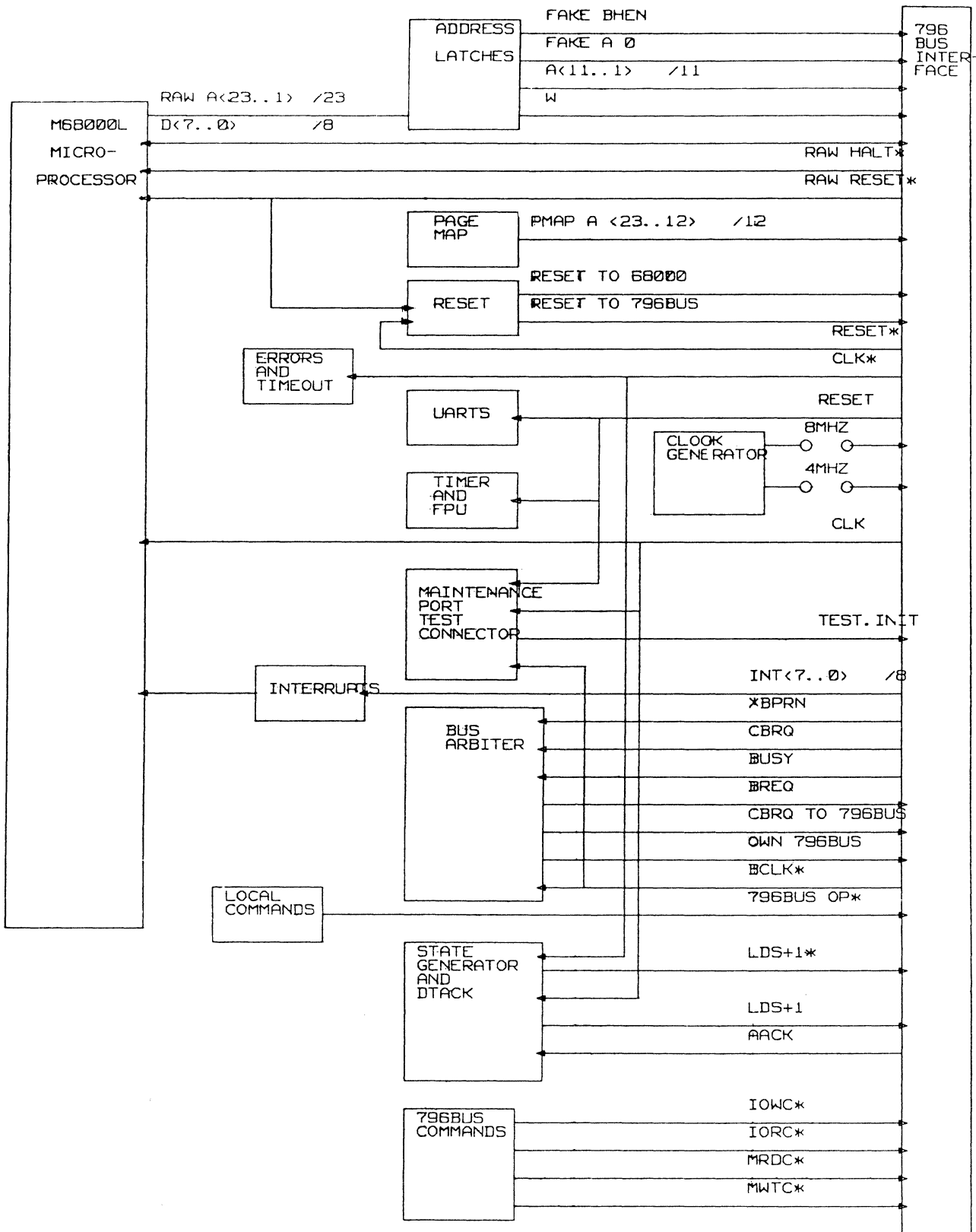
6.1 796 Bus Interface

The CPU/796 bus interface with the M68000LB and various functions located on the CPU board is shown in Figure 6-1. A more detailed diagram of the internal 796 bus including gating and buffers is shown in Figure 6-2. The bus interface contained on the CPU board consists of the required number of buffers and control logic to completely interface the functions of the M68000LB CPU with the bus. The buffers provide gated bidirectional data paths for communication between the M68000LB and other devices attached to the bus. Synchronous, serial bus arbitration also allows a large number of bus masters to exist in a single system.

Straps are available to drive the CCLK* and BCLK* signals of the 796 bus at either a 4MHz or 8MHz synchronous rate with the M68000LB (BCLK* at 4MHz allows a larger number of bus masters using serial arbitration). The XPU board drives the M68000LB from CCLK* (CLK) received from the bus. Bus request logic is driven from BCLK* also received from the 796 bus. In systems where more than one CPU exists, only one is strapped as a clock driver.

The CPU board both resets the bus or the board can be reset from the bus. The CPU devices are reset via an incoming X.INIT* signal from the bus. The M68000LB executes a reset instruction to reset all on-board devices (except status and context registers and the M68000LB) and sends a RESET.TO.796 BUS pulse to generate X.INIT* on the bus. A PWR.ON.TO 796 BUS pulse also resets the bus as well as a TST.INIT used during maintenance purposes.

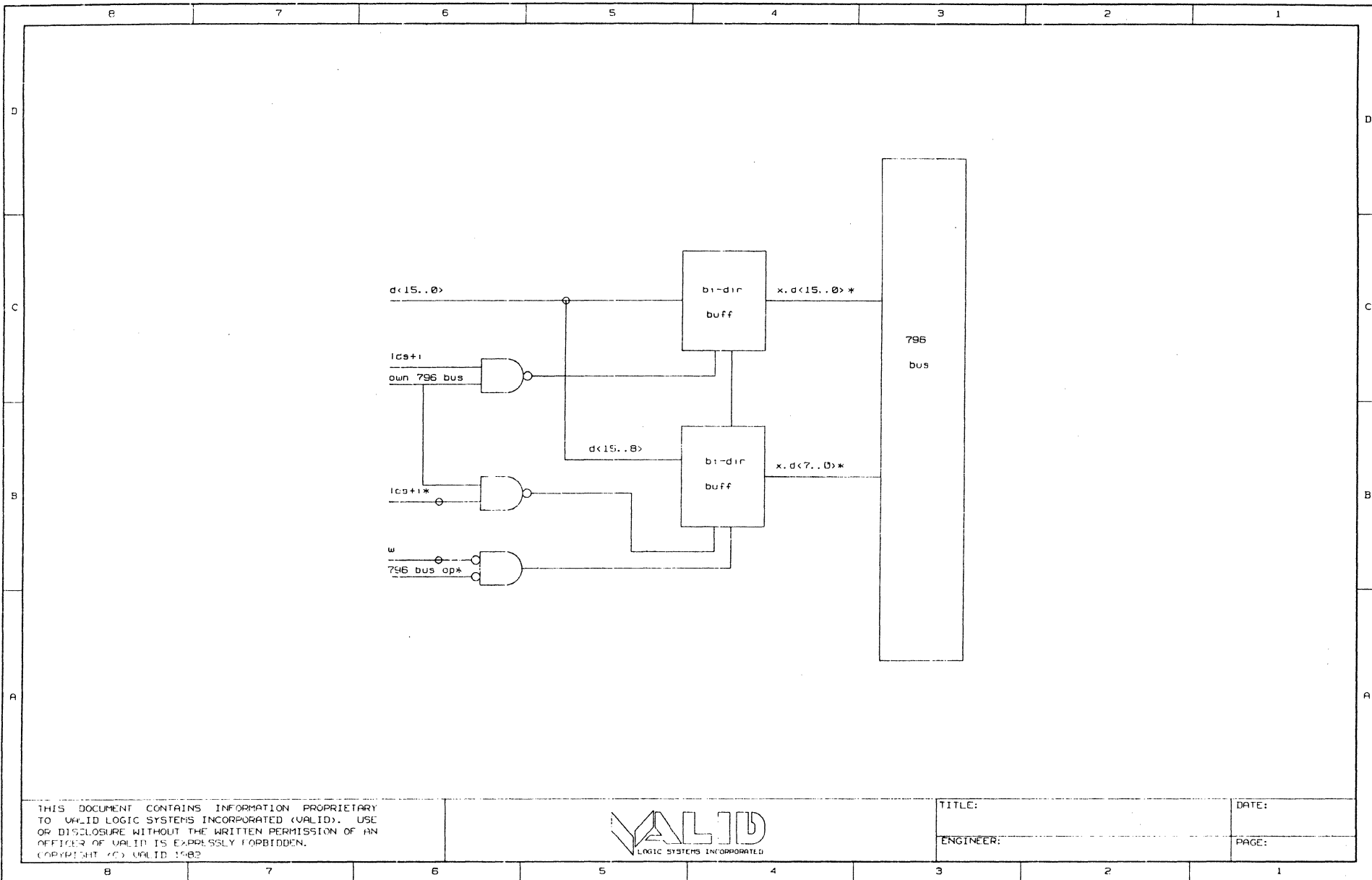
Both byte and word bus operations are supported by this interface. In the M68000LB, low-order bytes of words have odd addresses, while on the 796 bus, high-order bytes of words have odd addresses. The interface therefore translates the low-order address bit of M68000LB addresses. The address sent over the 796 bus has the low-order address bit (A<0>) set when the M68000LB is doing an even byte access. The bit in turn is clear when the M68000LB is doing an odd byte access or a word access.



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1982

TITLE: FIG 6-1
796BUS/CPU INTERFACE
EXPR:

ABBREV:
VERSION:



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT © VALID 1982

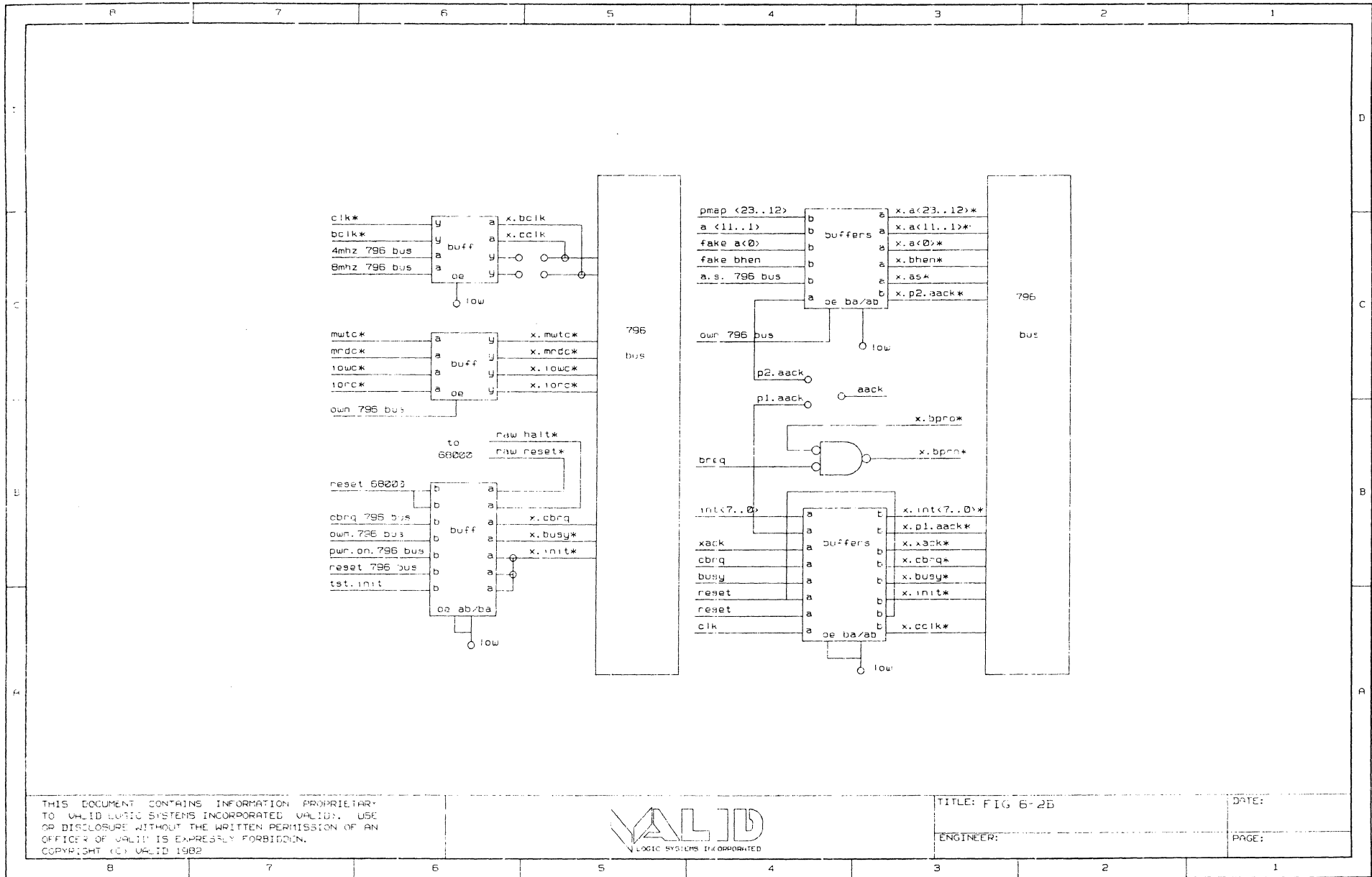


TITLE:

DATE:

ENGINEER:

PAGE:



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1982



TITLE: FIG 6-2B

DATE:

ENGINEER:

PAGE:

6.2 Non-Standard Bits

The non-standard 796 bus bits used in this system are shown below and are further described later in this section.

CONNECTOR	PIN(s)	SIGNAL
P1	25	X.P1.AACK*
P2	40	X.P2.AACK*
P2	39	X.AS*
P2	56,60	X.A<23>
P2	55,59	X.A<22>
P2	58	X.A<21>
P2	57	X.A<20>

The 796 bus advanced acknowledge signal (X.AACK*) is received on both connectors P1 and P2. The signal is strapped from one source or the other. The strapping of a tapped delay line minimizes the number of wait cycles the M68000LB executes while accessing 796 bus main memory.

The address strobe (X.AS*) signal is driven on connector P2. The signal allows 796 bus main memory to start access before mapping is complete. This minimizes M68000LB wait cycles while accessing memory boards which use the AS* signal.

Four high-order address bits (X.A<23..20>) are also driven on connector P2. Two different standards exist for the positions of X.A<22> and X.A<23> on P2. The bits are therefore driven according to both of these standards to allow the use of memory or peripheral board conforming to either standard.

6.3 796 Bus Description

The following paragraphs provide a general overview of the 796 bus operations. For more information as to the specific timing and technical details, refer to IEEE 796 specifications and related technical reference data listed under "Related Documents" contained in this manual.

The 796 bus is made up of the following signal categories:

- o 20 address lines
- o 16 bi-directional data lines
- o 8 multilevel interrupt lines
- o 16 bus control lines
- o 1 bus clock
- o 8 interrupt lines to M68000LB
- o Power supply lines

The 796 bus connects to the CPU board by means of connectors P1 and P2. Connector P1 contains the signal groups listed above. Connector P2 contains optional signal lines.

The address and data lines are driven by three-state devices. The interrupt and other control lines are open-collector driven.

6.3.1 Master-Slave Relationship

The 796 bus provides for both 8 and 16-bit bus masters and slaves. A bus master module (such as the CPU board) drives the command and address lines and controls the bus. A bus slave cannot control the bus and only performs functions based on commands from the bus. A particular system may have a number of masters. In this case, bus arbitration results when more than one requests control of the bus.

6.3.2 Bus Clock

A bus clock is provided by one bus master and is derived independent of the processor clocks. The bus clock is placed on the bus then routed back into the providing bus master equally with other boards connected to the bus. This is to prevent skewing of the clock between the source and other users. The bus clock provides a timing reference for bus arbitration between bus masters in cases where multiple requests occur from more than one bus master. Once a bus request is granted, single or multiple read/write transfers occur. Actual transfers on the bus proceed asynchronously with respect to the bus clock.

6.3.3 Signal Interface

The signals that make up the 796 bus interface for both connectors P1 and P2 are listed in table 1 and 2 respectively. The signal name and active logic level (* = active low) is listed as well as the function for each signal.

Table 6-1. 796 Bus Interface Signals (P1)

SIGNAL	FUNCTION
X.INIT* (Initialize)	Resets the entire system to a known state. Driven from bus master or front panel reset switch.
X.ADR0* - X.ADR9* X.ADRA* - X.ADRF* X.ADR10* - X.ADR13* (Address Lines)	20 address lines used to transmit address of memory location or I/O port to be accessed.
X.BHEN* (Byte High Enable)	Address control line used to specify that data will be transferred on the high byte (X.DAT8* - X.DATF*) 796 bus data lines.
X.INH1* (Inhibit RAM)	Prevents ROM memory from responding to the memory address on the system address bus.
X.INH2* (Inhibit ROM)	Prevents ROM memory from responding to the memory address on the system address bus.
X.DAT0* - X.DATF* (Data Lines)	Bi-directional lines used to transmit or receive information to or from a memory location or I/O port.

Table 6-1. 796 Bus Interface Signals (P1) (Continued)

SIGNAL	FUNCTION
X.BCLK* (Bus Clock)	X.BCLK is asynchronous to the processor clock. Negative edge (high to low) is used to synchronize bus priority resolution circuits. May be slowed, stopped, or single stepped for debugging.
X.CCLK* (Constant Clock)	Provides a constant clock signal for general use by modules on the system bus.
X.BPRN (Bus Priority-In)	Synchronized with X.BCLK*. Indicates to a particular bus master that no higher priority module is requesting use of the system bus.
X.BPRO (Bus Priority-Out)	Synchronized with X.BCLK*. Supplied to BPRN* input of the master module with the next lower bus priority.
X.BUSY* (Bus Busy)	Synchronized with X.BCLK*. Open-collector line driven by bus master currently in control to indicate bus is currently in use.
X.BREQ* (Bus Request)	Synchronized with X.BCLK*. Indicates a particular bus master requires use of the bus for one or more data transfers.

Table 6-1. 796 Bus Interface Signals (P1) (Continued)

SIGNAL	FUNCTION
X.CBRQ* (Common Bus Request)	Open-collector line driven by all potential bus masters to inform the current master that another wishes to use bus. If high, indicates to bus master no other bus master is requesting bus.
X.MDRC* (Memory Read Command)	Asynchronous with respect to X.BCLK*. Indicates address of a memory location is placed on system address lines and specifies contents of the addressed location are to be read and placed on system data bus.
X.MWTC* (Memory Write Command)	Asynchronous with respect to X.BCLK*. Indicated address of a memory location is placed on system address lines and that data is placed on the system data bus.
X.IORC* (I/O Read Command)	Asynchronous with respect to X.BCLK*. Indicates address of an input port is placed on system address bus and data is to be read and placed on system data bus.
X.IOWC* (I/O Write Command)	Asynchronous with respect to X.BCLK*. Indicates address of an output port is placed on system address bus and contents of system data bus are output to the address port.

Table 6-1. 796 Bus Interface Signals (P1) (Continued)

SIGNAL	FUNCTION
X.AACK (Transfer Acknowledge)	Asynchronous with respect to BCLK*. Indicates specific slave board read/write operation is complete.
X.INT0* - X.INT7* (Asynchronous Interrupt Lines)	Eight parallel interrupt lines used with interrupt resolution network.
X.INTA* (Interrupt Acknowledge)	Line driven by bus master transfer of interrupt information onto bus from slave priority interrupt controllers.

Table 6-2. 796 Bus Interface Signals (P2)

SIGNAL	FUNCTION
X.AACK	Same as P1.
X.ACLO (AC Low)	Monitor AC line voltage. Goes true when voltage drops below predefined level. Goes false when all DC voltages return to approximately 95% of requested value.
X.AS*	Allows main memory to start accessing before mapping is complete.
X.PFIN* (Power Fail Interrupt)	Interrupts processor when power failure occurs.
X.PFSN* (Power Fail Sense)	Latched output that indicates power failure has occurred.

Table 6-2. 796 Bus Interface Signals (P2) (Continued)

SIGNAL	FUNCTION
X.A<23..20>	Allows use of memory or peripheral boards.
X.PFSR* (Power Fail Sense Reset)	Used to reset power fail sense latch (PFSN*)
X.MPRO* (Memory Protect)	Prevents memory operation during period of uncertain DC power by inhibiting memory requests.
X.ALE (Address Latch Enable)	Generated by CPU to provide auxiliary address latch.
X.HALT*	Indicates master CPU is halted.
X.AUX RESET*	Initiates power up sequence.
X.WAIT (Bus Master Wait State)	Indicates processor is in wait state.

6.3.4 Data Transfers, Inhibit, and Interrupt Operations

Data transfer on the 796 bus consists of transferring read and write data. A typical data transfer occurs with a maximum bandwidth of 5MByte/sec for both single or multiple read/write transfers. Due to bus arbitration and memory access time, a typical maximum transfer rate is often on the order of 2MByte/sec.

6.3.4.1 Read Data. The read operation time is used by the 796 bus interface to decode the address and provide the required device selects. The device selects establish the data paths on the user system in anticipation of the

command strobe that follows. The read operation is initiated by the X.AS* signal from the M68000LB followed by strobe command (X.IORC* or X.MRDC*). Valid data is driven onto the bus following the command and is not removed until the command is cleared. When the command is cleared, the X.ACK* signal indicates the operation is complete.

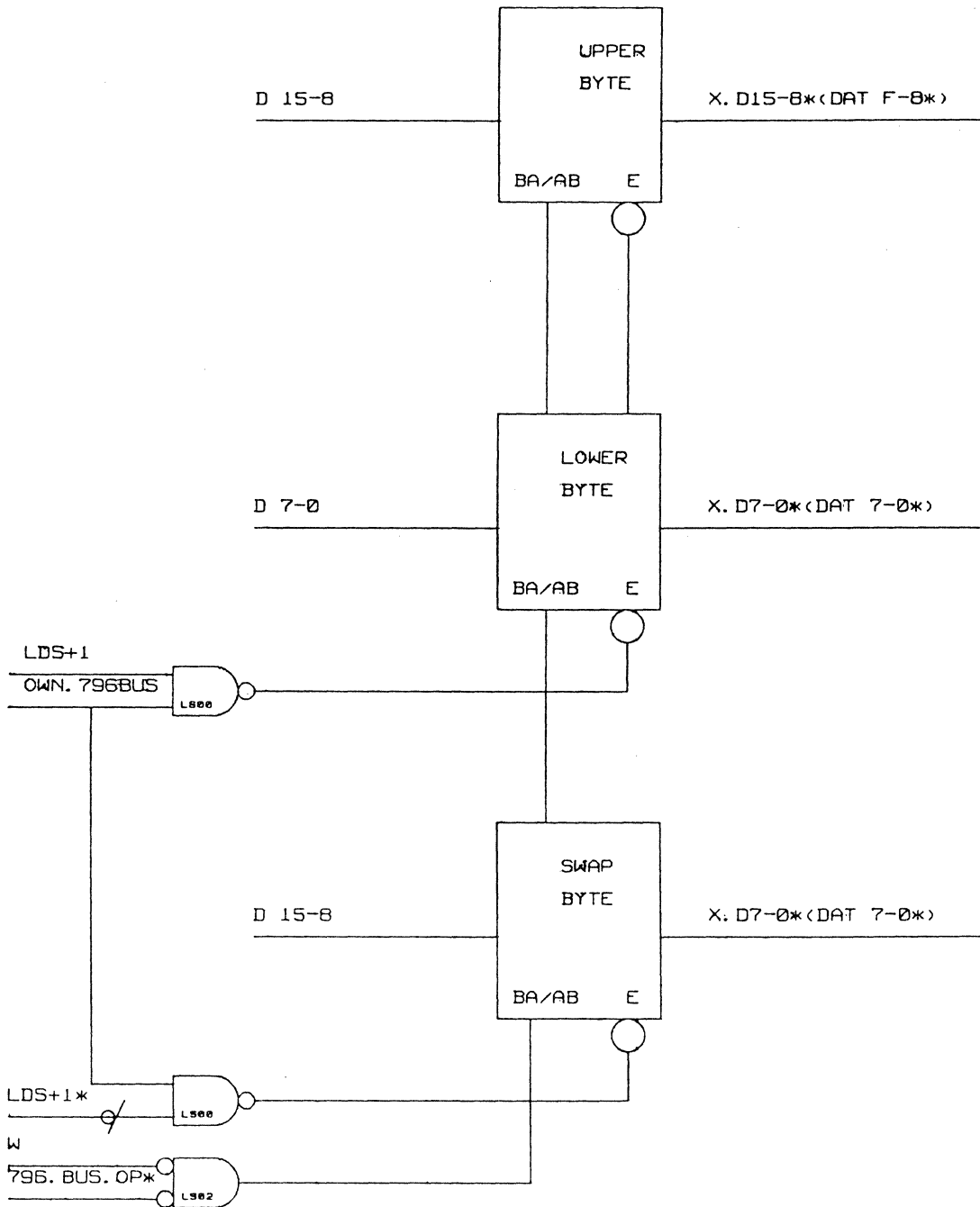
6.3.4.2 Write Data. During a write data operation, valid data is presented simultaneously with a stable address. Stable data before and after X.IOWC or X.MWTC enables the bus interface to latch data on either the leading or trailing edge of the command. The X.ACK* signal indicates the operation is complete.

A 16-bit master transfers data on the data lines using 8-bit or 16-bit paths depending on whether byte or word (2-byte) operation is specified. A word transfer specified with an odd I/O or memory address is actually executed as two single byte transfers on the data lines.

To adapt to both 8 and 16-bit devices, three buffers are used: lower, upper, and swap byte as shown in Figure 6-3. The lower byte buffer accesses X.DAT0* through X.DAT7*, the upper accesses X.DAT8* through X.DATF*, and the swap buffer accesses 796 bus data lines X.DAT0* through X.DAT7* and transfers the data from board data lines D8 through D15 and visa versa depending on the direction selected by the control logic. A read operation transfers low byte to high byte, a write transfers high to low.

6.3.4.3 Inhibit. Bus inhibit operations are required by bootstrap and memory mapped I/O configurations. The inhibit operation allows a combination of RAM/ROM memory mapped I/O to occupy the same memory address space. In the case of a bootstrap, it may be desirable to have both ROM and RAM occupy the same address space selecting ROM instead of RAM for low order memory only when the system is reset. Also, a system which has actual memory occupying the memory mapped I/O address space may need to inhibit RAM or ROM memory to perform its functions.

6.3.4.4 Interrupts. The interrupt lines (X.INT0* through X.INT7*) are used by a bus master to receive interrupts from bus slaves, other bus masters, or external logic such as power fail. The bus master may also contain external



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1982

TITLE: FIG 6-3
8/16-BIT DATA DRIVER
DATA TRANSFER

EXPR:

ABBREV:

VERSION:

interrupts (such as on the M68000LB CPU) which do not require the bus interrupt lines to interrupt the master.

There are two interrupt schemes used by bus interrupts: vectored and nonvectored. The M68000LB uses the vectored type of interrupts with the vector provided over the 796 bus address lines. The vectored interrupts are transferred from the slave to the bus master, when the master issues the X.INTA* command signal in response to an interrupt. When an interrupt occurs, the interrupt control logic in the CPU interrupts the M68000LB.

In single board systems which use a CPU board capable of bus vectored interrupt operation, the X.BPRN* pin must be grounded at master only if slave boards are to be accessed.

6.3.5 Slave Interface

The three basic elements of a slave bus interface are address decoders, bus drivers, and control logic.

6.3.6 Address Decoding

The address decode logic decodes the appropriate 796 bus address bits into RAM or ROM requests, or I/O selects.

6.3.7 Data Bus Drivers and Receivers

For users which only receive data from the 796 bus, buffers are used to ensure that maximum allowable bus loading is not exceeded. In systems that

place data onto the 796 bus data lines, three state drivers are required. For both read and write functions, bi-directional bus drivers are used.

6.3.8 Control Signal Logic

The control signal logic consists of circuits that forward I/O and memory read/write commands to their respective destinations, provide the bus with a transfer acknowledge response, and drive system interrupt lines.

6.3.9 Power Failures

Power failures are monitored by lines provided by connector P2. The power supply monitors the AC power level and when the power drops below an acceptable level, the supply raises the ACLO line. This tells the power fail logic that three milliseconds remain before the DC power falls below the regulated voltage levels. The power fail logic then sets a sense latch (X.PFSN*) and generates an interrupt (X.PFIN*) to the processor. After a 2.5 millisecond timeout, the memory protect signal (X.MPRO*) is asserted by the power fail logic to prevent any memory activity. As power fails, the memory goes on standby power.

As the AC line revives, the logic voltage level is monitored by the power supply. After power returns to its operating level for one millisecond minimum. The power supply sets the X.ACLO* signal to begin a restart sequence. This starts when the X.MPRO* then the X.INIT* become inactive. The bus master running then checks the power fail latch (X.PFSN*). If the latch is set, the power up routine branches to reset the latch (X.PFSR*), restore the environment, and resume execution.

SECTION 7

796 BUS COMMANDS

7.1 796 Bus Commands/CPU Interface

The 796 bus commands/CPU interface is shown in Figure 7-1. The 796 bus commands logic receives various enables, address lines, error inputs, and disables from the following CPU functions:

- o Address latches
- o Page map
- o Errors and timeout
- o Local commands
- o State generation and DTACK

Based on the configuration of the input signals (with no disables or errors), the command logic provides one of the following four commands to the 796 bus interface:

- o IOWC* : I/O write command
- o IORC* : I/O read command
- o MWTC* : Memory write command
- o WRDC* : Memory read command

7.2 796 Bus Commands Detailed Description

The 796 bus commands logic is shown in Figure 7-2. The logic consists of a one of eight decoder and associated enable gating. The decoder generates one of four outputs based on the address inputs (A0 through A2) when the decoder is enabled.

Either AND gate is enabled by a 796 BUS.OP true previous to a delayed data strobe (DS+2.DLY) occurring. The particular gate activated depends on the condition of the read/write selection lines (W and W*). The read output enable can always be activated however, the write function can be disabled by a 796 BUS.W.COMD.DIS* input. Without the disable or an SMAP.ERR present,

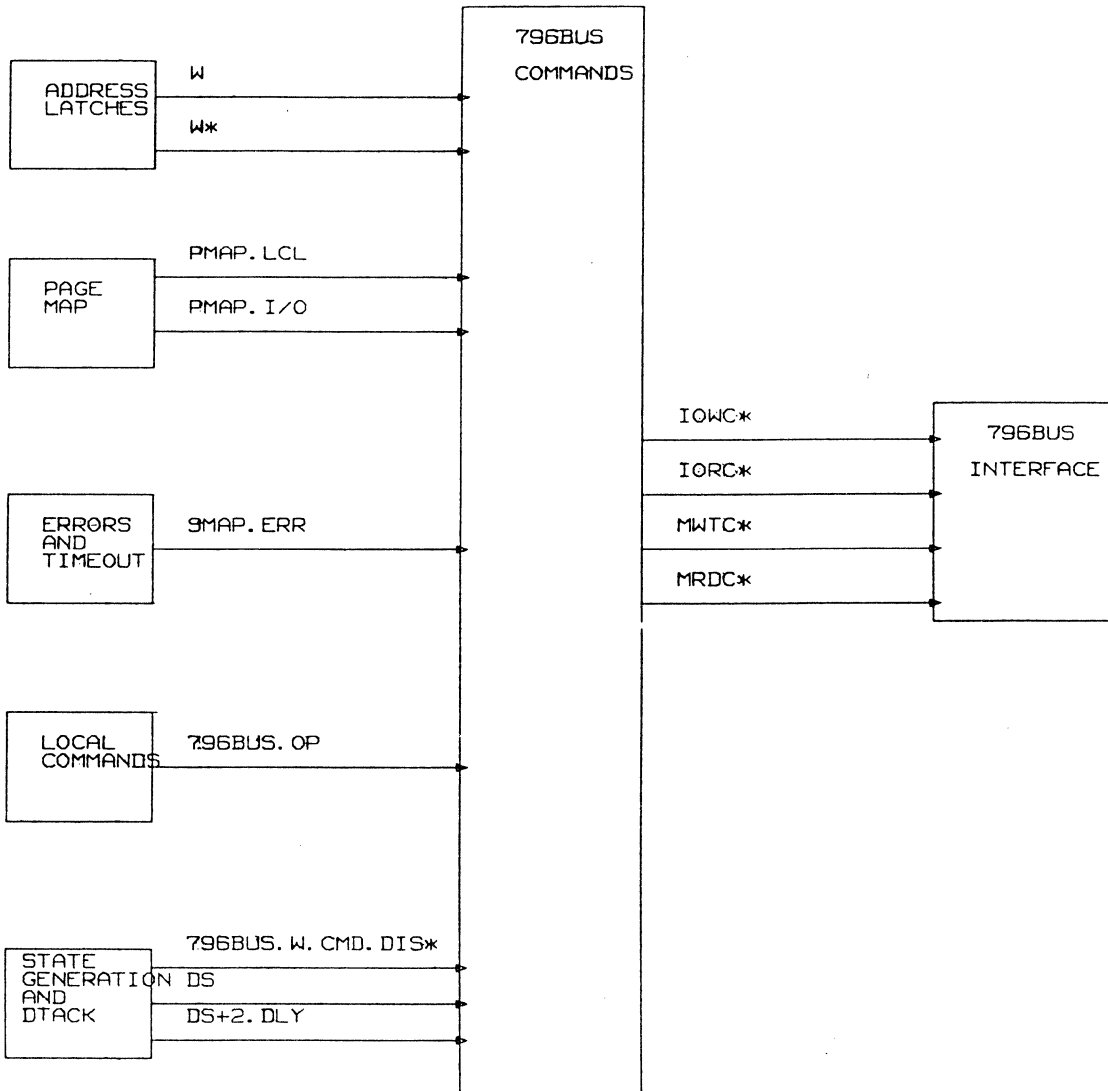


FIG 7-1 796BUS COMMANDS/CPU INTERFACE

THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1982

TITLE: FIG 7-1
796BUS COMMANDS
CPU INTERFACE

ABBREU:

EXPR:

VERSION:

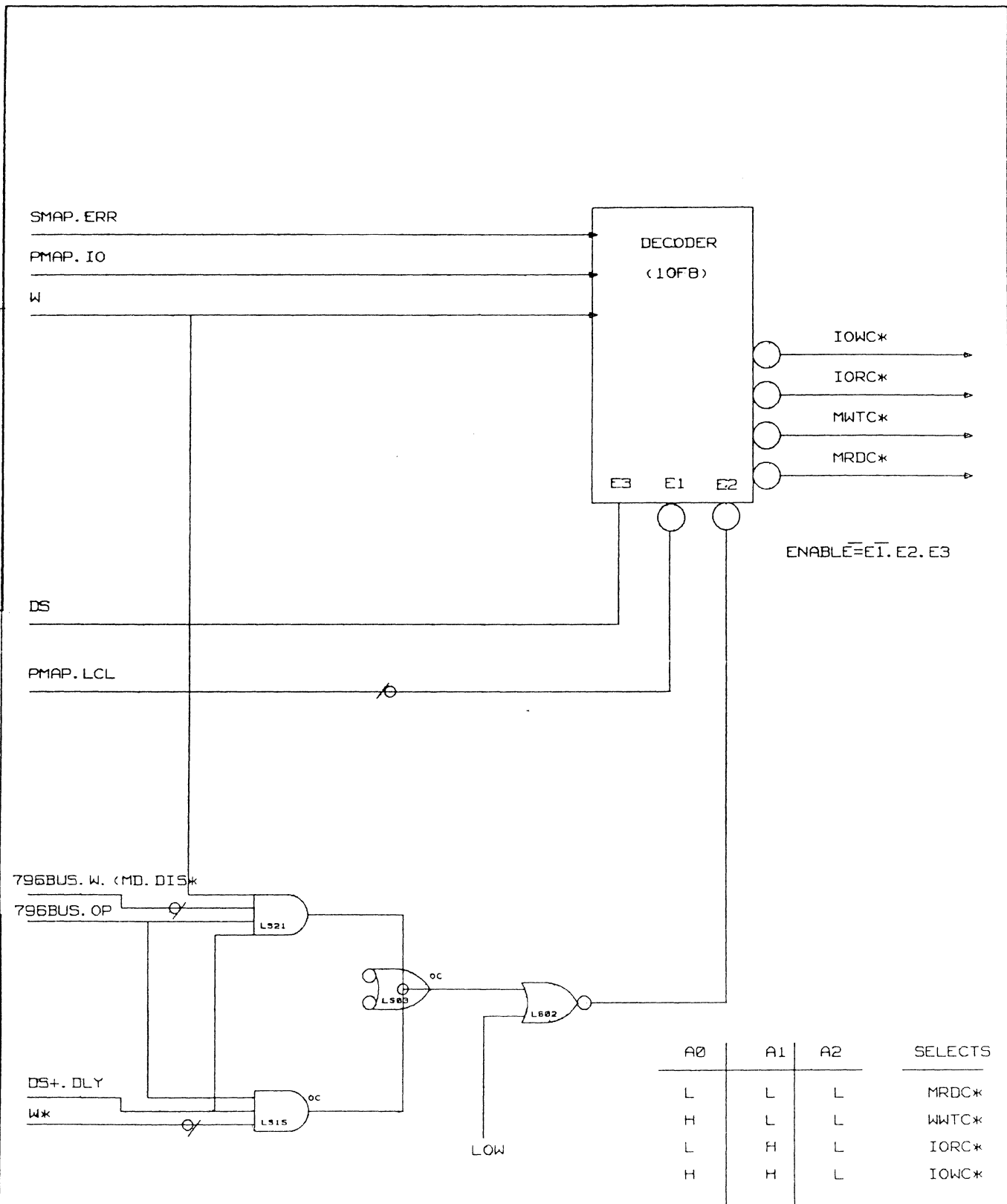


FIG 7-2 796BUS COMMANDS LOGIC

THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1982

TITLE: FIG 7-2
796BUS COMMANDS LOGIC

EXPR:

ABBREV:

VERSION:

the enable decoder address inputs provide one of the four selected outputs as follows:

PMAP.10 W = IOWC*

PMAP.10 W = IORC*

PMAP.10 W = MWTC*

PMAP.10 W = MRTC*

The SMAP.ERR signal sets all four I/O and memory output read/write selection lines to high therefore disabling the command function. After the error, a reset from the M68000LB to the errors and timeout logic resets the SMAP.ERR line thus re-enabling the command decoder.

SECTION 8
BUS ARBITER

8.1 Bus Arbiter/CPU Interface

The bus arbiter/CPU interface is shown in Figure 8-1. The bus arbiter implements 796 bus protocol and provides fast bus exchange for the CPU in conjunction with other bus masters connected to the 796 bus. The arbiter receives signal inputs from the status register and state generation and DTACK, and to and from the 796 bus interface. The signals are listed and described in table 8-1.

Table 8-1. Bus Arbiter Interface Signals

SIGNAL	FUNCTION
796 BUS.LOCK	Holds ownership of 796 bus and prevents BREQ line from being reset.
AS+1	Address strobe used for setting BREQ true.
OWN.796 BUS*	Activates 796 bus interface. Complementary with OWN.796 BUS.
OWN.796 BUS	Activates 796 bus interface bus commands buffer, generates X.BUSY* signifying bus is in use, and provides enable to data bus buffers. Complement to OWN.796 BUS*.

Table 8-1. Bus Arbiter Interface Signals (Continued)

SIGNAL	FUNCTION
CBRQ.TO.796 BUS	Generates X.CBRQ* through 796 bus interface buffers to signify that another master wishes to use the bus. When false, indicates no other bus master is requesting bus.
BREQ	Indicates CPU requires use of 796 bus.
CBRQ	Common bus request informing CPU that another bus master wishes to use 796 bus.
X.BPRN*	Indicates no higher priority module is requesting use of 796 bus. Synchronized with BCLK*.
BCLK*	Negative going edge sets current condition into bus arbiter logic.
BUSY	Indicates to bus arbiter that 796 bus is currently in use.

8.2 Bus Arbiter Logic

The bus arbiter logic is shown in Figure 8-2. The logic consists of two J-K flip-flops used to set and reset the 796 bus request and control lines as controlled by the bus selection inputs and related gating.

During operation, the BREQ is set by signals 796 BUS.OP, AS+1, and a low-going clock pulse to indicate the CPU wants the bus for the next or more cycles. At the same time, CBRQ.TO.796 BUS indicates to all bus masters that the CPU wishes to use the bus. If the bus is not busy and X.BPRN* is not set, a high to the second flip-flop J input along with the next low-going clock pulse generates OWN.796 BUS and OWN.796 BUS* to activate the bus interface buffers.

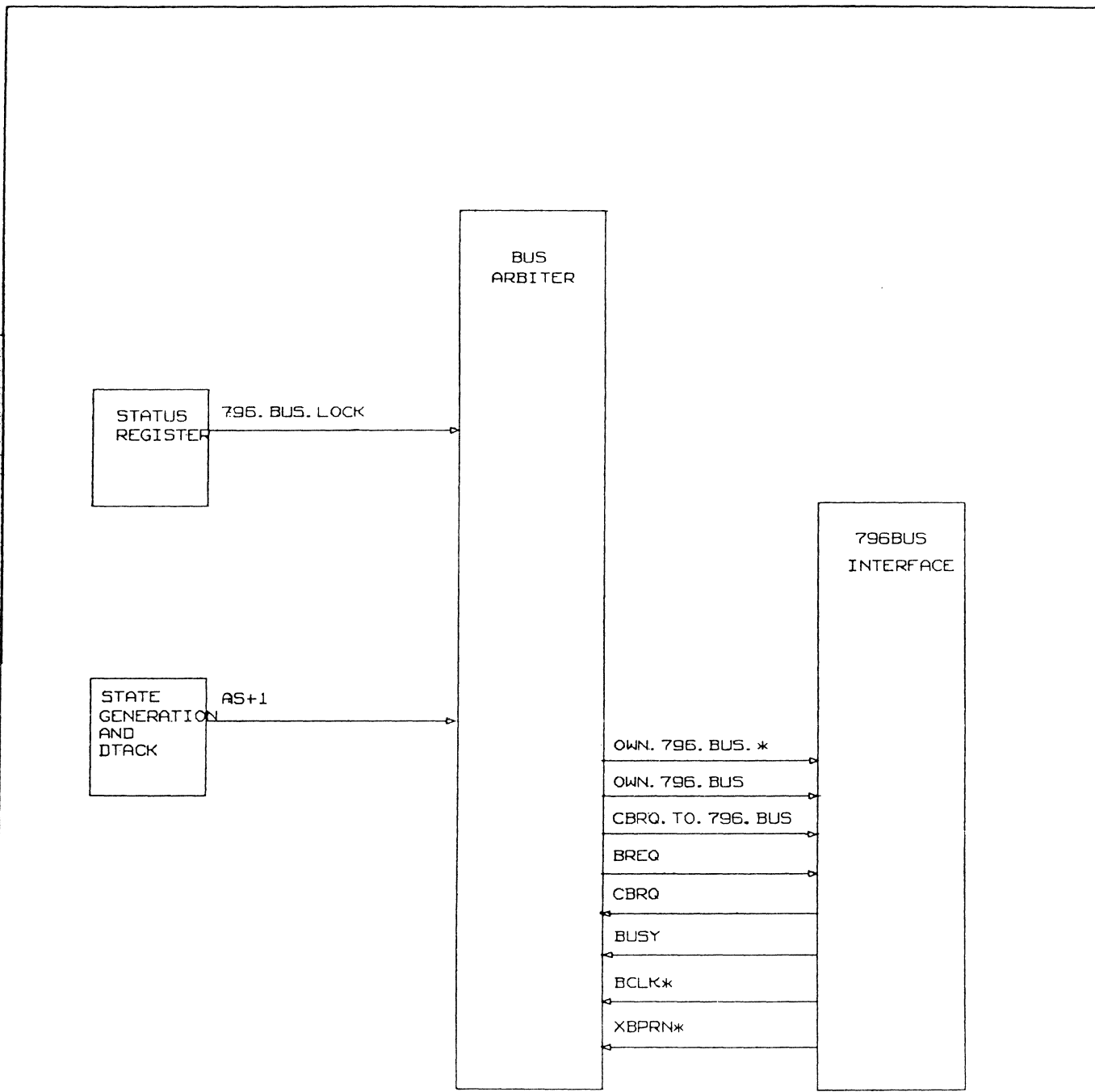
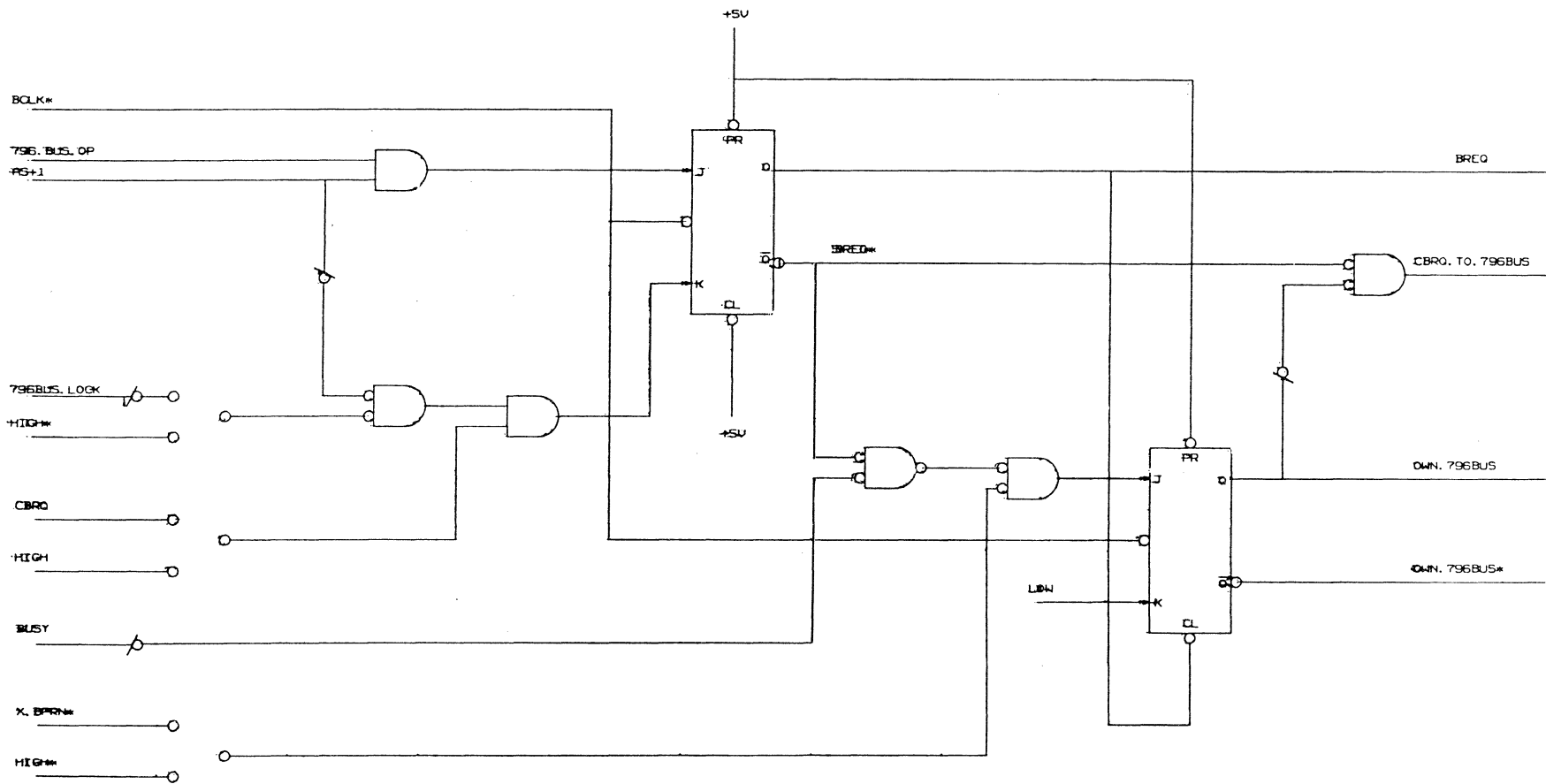


FIG B-1 BUS ARBITER/CPU INTERFACE

THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1982

TITLE: FIG B-1
 BUS ARBITER/CPU INTERFACE
 EXPR:

ABBREV:
 VERSION:



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN.
 COPYRIGHT (C) VALID 1982

TITLE:

FIG B-2 BUS ARBITER LOGIC

ABBREV:

EXPR:

VERSION:

If it is desired to lock the bus into a constant bus request situation, a 796 BUS.LOCK line can lock the flip-flop so the BREQ line cannot be reset. A low common bus request line (CBRQ) also disables the reset of BREQ as long as no other bus master desires to use the bus. A BUSY line from the bus indicates the bus is in use and disables the OWN.796 BUS outputs. These lines in turn disable the CPU 796 bus interface.

8.3 Bus Arbiter Strapping

Strapping of certain signals is used in situations where constant domination of the CPU on the 796 bus is desired (one example is where no other bus masters exist). These situations involve the strapping of the CBRQ or X.BPRN* signals. The CBRQ feature can be constantly enabled by strapping the input true. The X.BPRN* feature can be constantly enabled by strapping it false. The true CBRQ line then generates a constant bus request and the X.BPRN* always simulates a bus priority request. The 796 BUS.LOCK feature can also be disabled by strapping it false.

SECTION 9 CPU ADDRESSING

9.1 Address Bus/CPU Interface

The address bus/CPU interface is shown in Figure 9-1. Addressing is provided on the address bus (RAW A<23..1>) from the M68000LB to the address latches.

The latches operate in an asynchronous manner (constantly enabled). Each address from the M68000LB is strobed onto the address bus lines (A<23..1>) by the data strobe (DS) along with the read/write command (W), upper and lower data strobes (UDS* and LDS*) which in turn generate FAKE.A<0> and FAKE.BHEN as required. Because the M68000LB is capable of updating the address lines in approximately 30 nanoseconds, a delay line is used to delay the data strobe by approximately 125 nanoseconds before placing a new address on the latch outputs. This allows time for the addresses to be used either by the CPU or 796 bus by keeping them available on the latch outputs for the correct amount of time.

The address bus interfaces various functions on the CPU board. These are each listed and described in detail in table 9-1. The FAKE.A<0> interfaces with 796 bus address bit X.A<0>*. The bit is generated by the address latches LDS* signal from the M68000LB. The bit is used to signify data is to be transferred on the lower byte 796 bus data lines. The FAKE.BHEN indicates data is to be transferred on the high byte 796 bus data lines. The AS* signal is generated by the M68000LB to the state generation and DTACK logic to indicate a valid address is on the address bus.

Table 9-1. CPU Addressing Features

FUNCTION	DESCRIPTION
Local Commands	Decodes commands for all CPU functions.
796 Bus Commands	Uses W and W* with other commands and control signals to generate one of four bus commands depending on the CPU mode.

Table 9-1. CPU Addressing Features (Continued)

FUNCTION	DESCRIPTION
Segment Map	Addresses (A<22..16>) and context register CTXT<3..0> address the segment map to generate map addresses to the page map. The signal along with other control signals is used to address a PROM which in turn generates segment and page map status signals. The W signal also controls bi-directional buffers which allow either segment map data onto the data bus or lines from the data bus directly to the page map and errors and timeout logic.
Page Map	Address A<15..12> along with segment map addresses SMAP<7..0> are used to generate page map addresses to the 796 bus and status signals to other CPU functions. The W signal is used in the generation of PMAP.DIRTY (see memory management). The W signal also controls bi-directional buffers which allow addressing and generation of status signals directly from the data bus to the 796 bus.
RAM and ROM	Address bit A<10..1> provides addressing to the RAM memory. Address bits A<13..1> provide addressing to the ROM memory. The W* signal along with the UDS* and LDS* select the read and write operations of the RAM.

Table 9-1. CPU Addressing Features (Continued)

FUNCTION	DESCRIPTION
TIMER and FPI	Address bits A<2..1> selects the interval timer when the CPU selects it for use. The floating point processor is addressed by address bit A<1> when CPU selected.
UARTS	Address bits A<2..1> selects the UARTS internal function for use when selected for use by the CPU.

9.2 Logical Address Space Structure

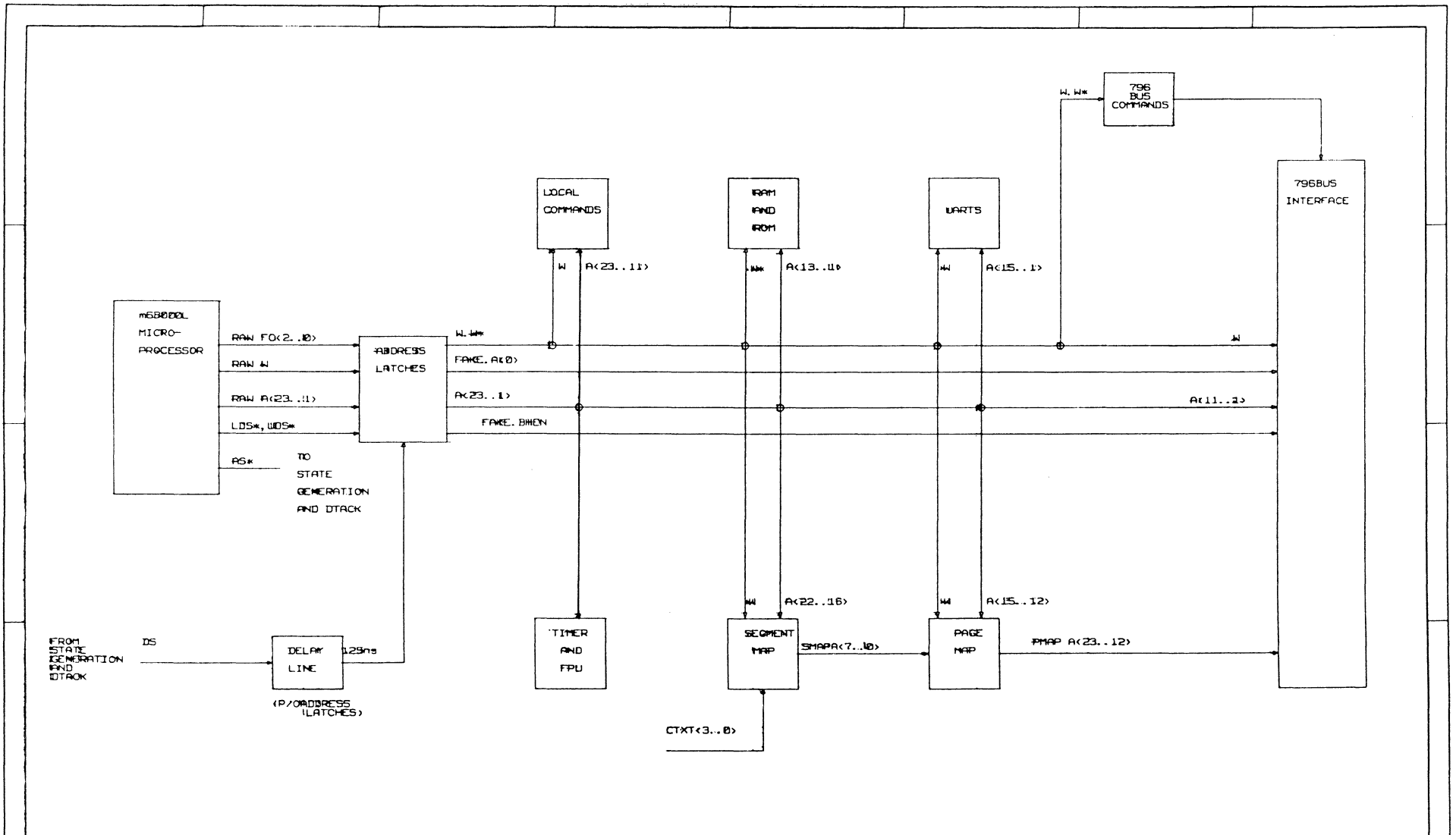
The 16 megabyte logical address space of the M68000LB is divided into 256-64K bytes segments. Logical addressing operates in both the user and supervisor modes.

9.2.1 User Mode

In the user mode, accesses within the lowest 64 segments (4Mbytes from \$000000 through \$3FFFFF) are mapped and checked for correct access mode as specified in the map. Correct accesses map to either the I/O space or memory space of the 796 bus. All others cause an address bounds bus-error trap. The user cannot access on-board RAM or other local resources.

9.2.2 Supervisor Mode

In the supervisor mode, the second segment (64Kbyte from \$010000 through \$01FFFF) is reserved to access local resources such as on-board RAM and ROM. Accesses in this segment are not mapped. All other supervisor mode accesses within the lowest 64 segments are mapped and checked for correct access mode as specified in the map.



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1982

TITLE: FIG B-1: ADDRESS BUS/CPU INTERFACE

ABBREV:

EXPR:

VERSION:

In general, supervisor references to other than the lowest 64 segments cause an address-bounds bus-error trap. However, the context register contains a field which controls reading and writing the map. If the context register contains a value which allows access to the map, then (supervisor mode only) addresses in segments 128 through 191 (4Mbyte from \$800000 through \$BFFFFFF) are not mapped through to the 796 bus, but reference the map itself.

When the CPU is in the boot state (see status register), supervisor mode accesses to the first segment are unmapped and forced to local resources.

9.3 Local Resources

The CPU board contains certain local resources. These include ROM, RAM, UARTS, timer, floating point processor, and various status registers. The software running in supervisor mode accesses these local resources by generating addresses in the second segment (64Kbytes from \$010000 through \$01FFFF). First segment addresses (64Kbytes from \$000000 through \$00FFFF) also access these resources if the CPU is in the boot state.

NOTE

Accesses to local resources are not mapped since some local register control the mapping function itself. Local resources are not accessible by other 796 bus masters.

The second segment, low-order 16-bits select the local resources as follows:

Address Range	Size	Function
010000..013FFF	16Kbytes	Local ROM
014000..017FFF	16Kbytes	(reserved, timeout)
018000..0187FF	2Kbytes	Local RAM
018800..018FFF	14Kbytes	(reserved, timeout)
01C000..01FFFF	16Kbytes	Local Devices

Local devices are selected by the low-order 14-bits as follows:

Base Address	Device
01C001	Host UART
01C801	Terminal UART
01D001	8253 Programmable Interval Timer
01D801	8232 Floating-Point Processor
01E001	Context Register
01E801	Error Register
01F001	Status Register
01F801	Switches (Context)

All devices in region \$01C000 through \$01FFFF are byte devices. Odd byte addresses should be used to access these devices. For a device with internal addressable functions, the bare address for the device must be odd, and the low-order ten word address bits (A<10..1>) select the function within the device. Byte accesses on even addresses within this area do not activate any device, and only the odd byte is read or written during a word access within this area.

SECTION 10

REGISTERS

10.1 Registers/CPU Interface

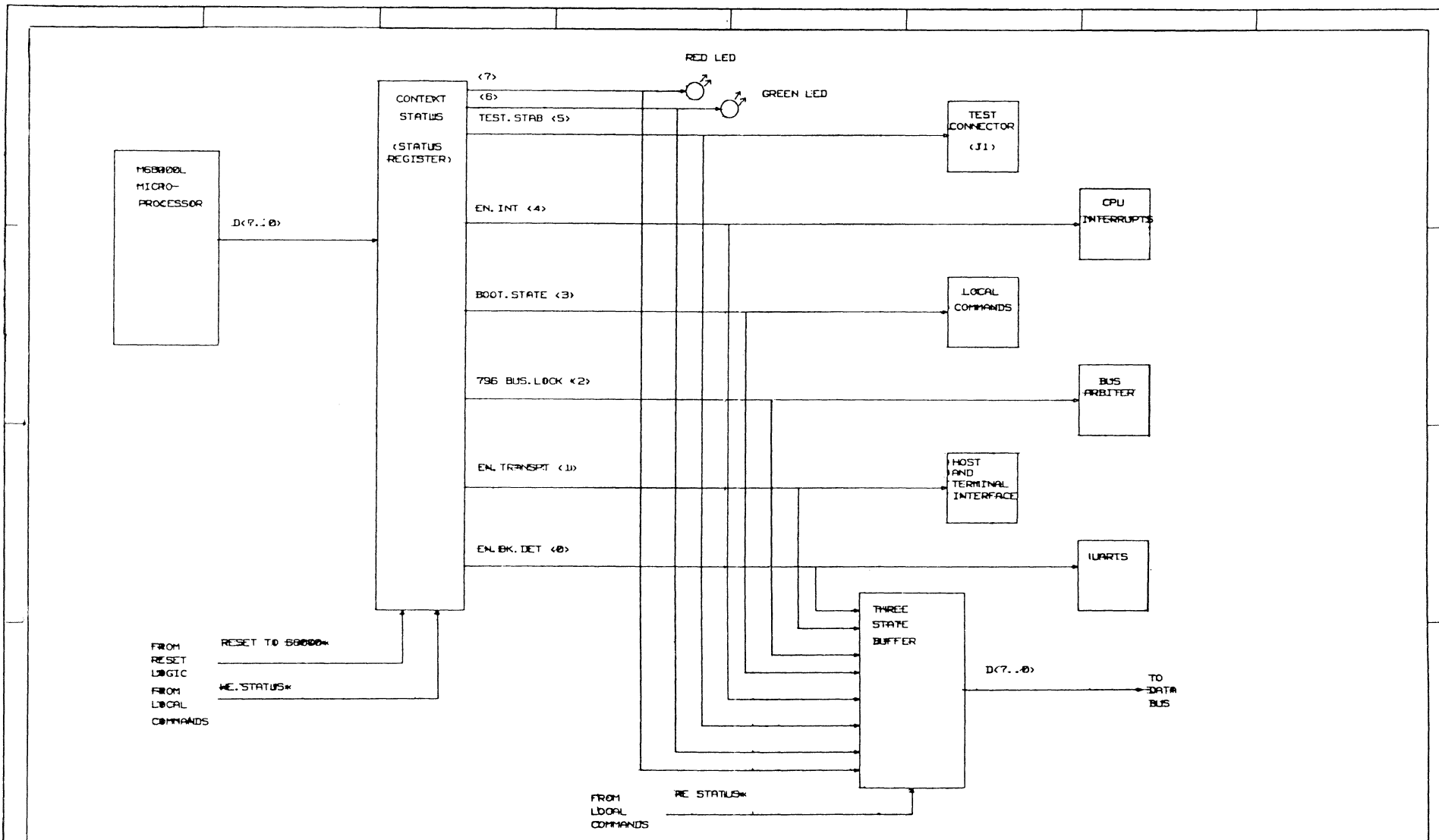
The CPU includes status, error, and context registers contained as part of the context and status switches and errors and timeout logic functions of the board. The purpose of the registers is to have available certain conditions of the CPU for monitoring and information purposes when required. The registers may be written-into and read-from during CPU operations. The registers interface with various functions of the CPU board and the data bus as shown in Figures 10-1, 10-3, and 10-5. A description of each register and its particular functions are included in the paragraphs that follow.

10.2 Status Register

The status register interface is shown in Figure 10-1. The purpose of the status register is to light the red and green LEDs based on the results of CPU diagnostics and provide control signals to various functions on the CPU board as well as place the status of the bits onto the M68000LB data bus.

The register can be either written-into or read-from using M68000LB byte or word operations. The register can also be cleared by resetting all bits (using RESET TO 68000*) during a 796 bus or power-up reset. The effects of writing into the status register is observed after the write is completed (before the beginning of the next instruction). The register is written into from the data bus by a WE STATUS* signal generated from the local commands logic.

The data written into the register is made available on the register output lines to the CPU board functions depending on the CPU operation being performed. The same data is also made available to the three state interface buffers for gating onto the M68000LB data bus. This is accomplished by generating a RE.STATUS* signal to the buffers from the local commands logic.



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1982

TITLE:	FIG 10-1 STATUS REGISTER/CPU INTERFACE	ABBREV:	
EXPR:		VERSION:	

The register contains fields related to each bit position as shown in Figure 10-2. These correspond to bit lines on the data bus. The fields are also listed in table 10-1 as well as each name and definition.

Table 10-1. Status Register Fields

FIELD	NAME	DEFINITIONS
<7>, <6>	RED.LED, GREEN.LED	Used by diagnostic software to display results of diagnostics. After successful completion. EPROM CPU diagnostics light the green LED and red LED is lighted for unsuccessful completion. Both LEDs are dark if diagnostics do not complete. Both are dark at power-up.
<5>	TST.STB	Connected to test connector. Used by software to signal events to external diagnostic equipment.

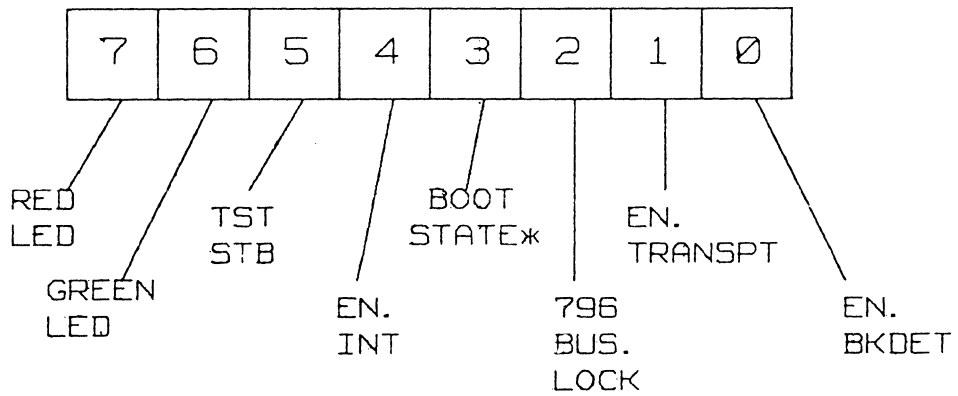
Table 10-1. Status Register Fields (Continued)

FIELD	NAME	DEFINITIONS
<4>	EN.INT	Handle interrupts to M68000LB. Since status register is cleared at power-up, interrupts are disabled and are not re-enabled by software until map and trap vectors are initialized.
<3>	BOOT.STATE*	Input to local commands. Controls mapping of the lowest 64K byte segment of each address space. Normally, only the second segment of each address space accesses local resources (on-board ROM and RAM). However, in boot state both the first and second segments access those resources. This allows the M68000LB reset vector to be located in on-board ROM.

Table 10-1. Status Register Fields (Continued)

FIELD	NAME	DEFINITIONS
<2>	796 BUS.LOCK	Controls release of 796 bus after next bus access. Otherwise, it is released to any requestor, regardless of priority.
<1>	EN.TRANSPT	Directly connects two RS-232C ports by hardware in a bi-directional channel allowing the two UARTS to continuously monitor the ports (two ports operating at the same baud rate). Allows compatibility with Motorola MACSBUG monitor.
<0>	EN.BK.DET	When true, a break detected by either UART causes a power-on reset to be issued on the 796 bus. This allows resets to be issued on the bus remotely using the break key of a standard ASCII terminal.

STATUS REGISTER



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN.
 COPYRIGHT (C) VALID 1982

TITLE: FIG 10-2
 STATUS REGISTER FIELDS

ABBREV:

EXPR:

VERSION:

10.3 Error Register

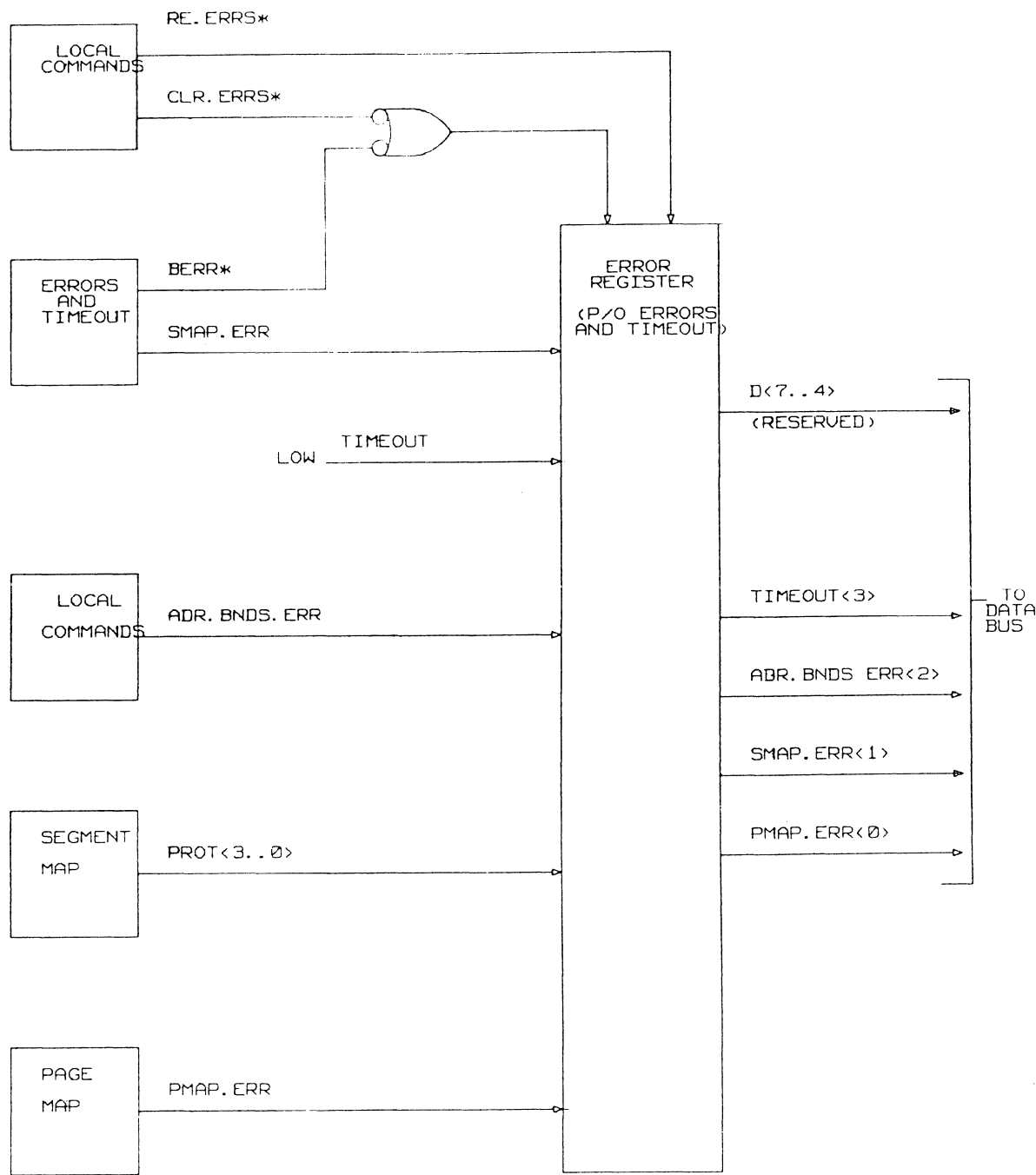
The register loads error information onto data bus bits <3> through <0>. Incoming error information is received from the errors and timeout logic, local commands, segment map, and page map. The error register latches the current errors onto the data bus upon receipt of an RE.ERRS* input from the local commands when the BERR* signal is received from the timeout logic. The latches are cleared when a CLR.ERRS* is received from the local commands. The error register fields are shown in Figure 10-4. The errors and their definitions are listed in table 10-2.

Table 10-2. Error Register Fields

FIELD	NAME	DEFINITIONS
<7..4>	Reserved	
<3>	TIMEOUT	Indicates addressed location did not respond.
<2>	ADDRESS BOUNDS	Indicates an illegal logical address was generated.
<1>	SEGMENT MAP	Indicates access mode violation detected in segment map.
<0>	PAGE MAP	Indicates access mode violations detected in page map.

10.4 Context Register

The context register contains the process ID of the current process being executed as well as the map control field which enables the map to be

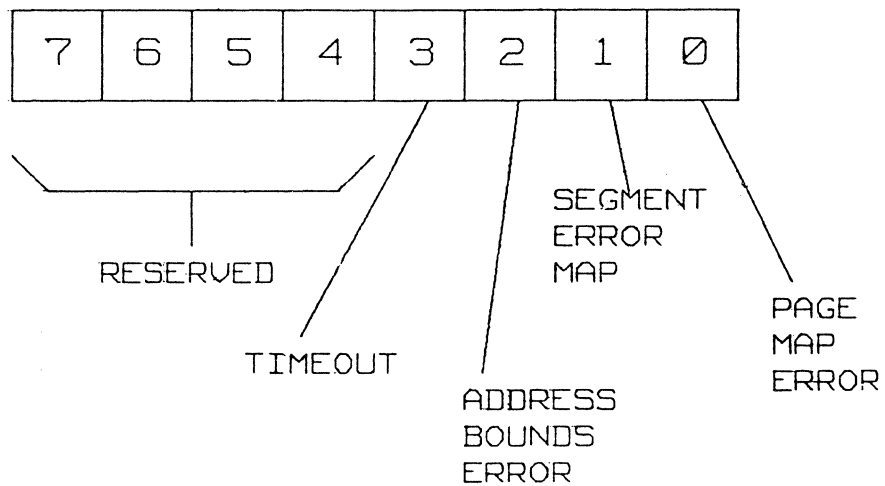


THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1992

TITLE: FIG 10-3
 ERROR REGISTER/CPU INTERFACE
 EXPR:

ABBREV:
 VERSION:

ERROR REGISTER



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN.
 COPYRIGHT (C) VALID 1982

TITLE: FIG 10-4
 ERROR REGISTER FIELDS
 EXPR:

ABBREV:

VERSION:

accessed. The context register interfaces within the CPU board as shown in Figure 10-5. The register can be read or written by the M68000LB using byte or word operations. All bits in the register are reset by a 796 bus reset or a power up (RESET TO 68000*) signal from the CPU reset logic. The signal occurs from the local commands. For reading the register bits onto the data bus, an RE CTXT* signal is applied to the three-state interface buffers from the local commands.

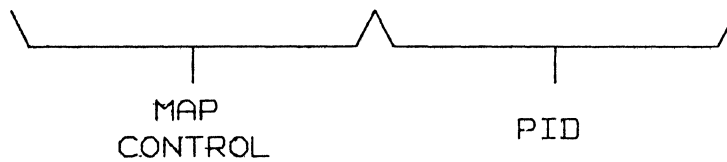
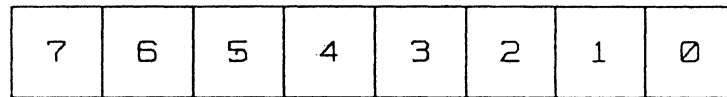
The context register fields are shown in Figure 10-6. The two fields contained in the register control map accesses (CTXT<3..0>) and name of the current process (PID). The PID field (current name of the 4-bit process), determines the section of the segment map is used in the address translation. The map control field controls access to the map. The context register fields, names, and definitions are listed in table 10-3.

Table 10-3. Context Register Fields

FIELD	NAME	DEFINITIONS
<7..4> (map control)	CTXT<3..0>	Controls map accesses.
PID:		
<3>	ALLOW.MAP.ACCESS	Enables map accesses.
<2>	SEL.LOW.MAP	Enables high (low) map accesses.
<1>	SEL.SMAPP	Enables segment map accesses.
<0>	reserved	

The ALLOW.MAP.ACCESS bit allows reading or writing of the segment or page map. In those cases, the SEL.SMAPP bit determines which map is accessed as follows:

CONTEXT REGISTER



MAP ACCESS CONTROL 7..4

NAME OF CURRENT PROCESS:

- | | |
|--------------------|---|
| ALLOW. MAP. ACCESS | 3 |
| SEL. LOW. MAP* | 2 |
| SEL. SMAP | 1 |
| RESERVED | 0 |

THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN.
COPYRIGHT (C) VALID 1982

TITLE: FIG 10-6	ABBREV:
CONTEXT REGISTER FIELDS	
EXPR:	VERSION:

- o SEL.SMAP true enables segment map access.
- o SEL.SMAP false enables page map access.
- o SEL.LOW.MAP control access to the low half of the segment map (not relevant to 4K-map configuration).

When map access is enabled, an address from the M68000LB accesses the map only if its high order address bit (A<23> is set. All other addresses are mapped as described in memory management.

SECTION 11 ERRORS AND TIMEOUT

11.1 Errors and Timeout/CPU Interface

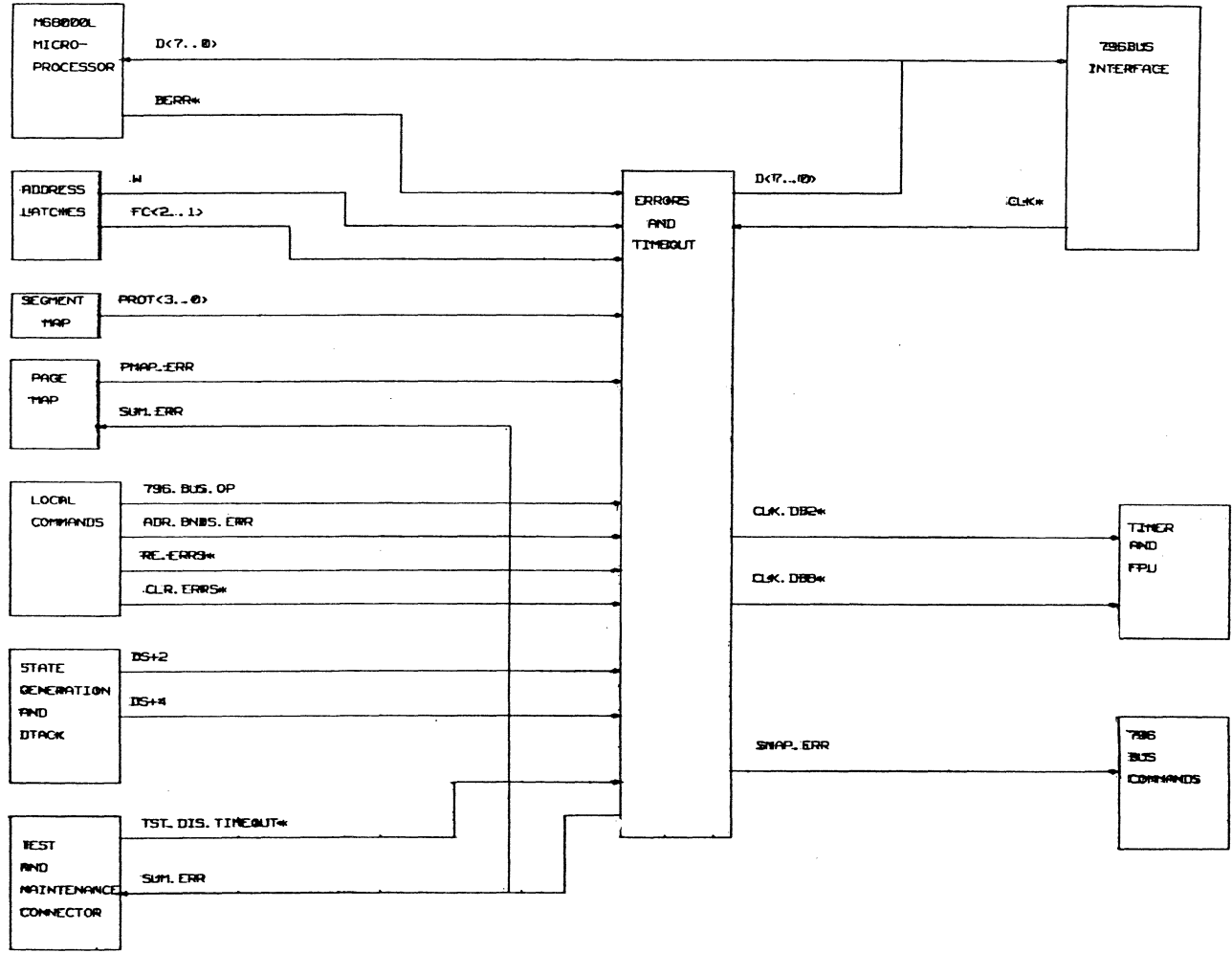
The errors and timeout logic interfaces within the CPU as shown in Figure 11-1. The purpose of the logic is to capture the most recent cause of a bus error trap. It is then read by the 68000 using byte or word operations (see error register section of this manual).

The errors and timeout logic receives a BERR* input from the M68000LB, PMAP.ERR from the page map, and address bounds error (ADR.BNDS.ERR) from the local commands logic. The logic generates an SMAP.ERR to the 796 bus commands logic and a SUM.ERR output to the page map and test and maintenance connectors depending on the type of error occurring.

Data strobes (DS+2 and DS+4) are received from the state generation and timeout to provide a clear signal to the internal timeout counter and a strobe (DS+4) to clock the current error (BERR*) into the error register latches. The current error is then placed on the 68000 data bus by the read errors (RE.ERRS*) signal issued from the CPU local commands logic. The local commands also provides a CLR.ERRS* signal used to reset the error register to its initial state.

A PROM is also used in the errors and timeout logic to monitor conditions of the address latches W and FC<2..1> outputs, 796.BUS.OP line, and PROT<3..0> from the segment map. Incorrect conditions from these lines as well as a page map error indicates an access mode violation.

The errors and timeout logic generates two clock signals (CLK.DB2* and CLK.DB8*) for use by the timer and FPU from the timeout counters. These are tapped from the timeout counter outputs. A TST.DIS.TIMEOUT* line is also included to disable the timeout logic as required for testing and maintenance purposes.



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO WALD LOGIC SYSTEMS INCORPORATED (WALD). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF WALD IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) WALD 1992

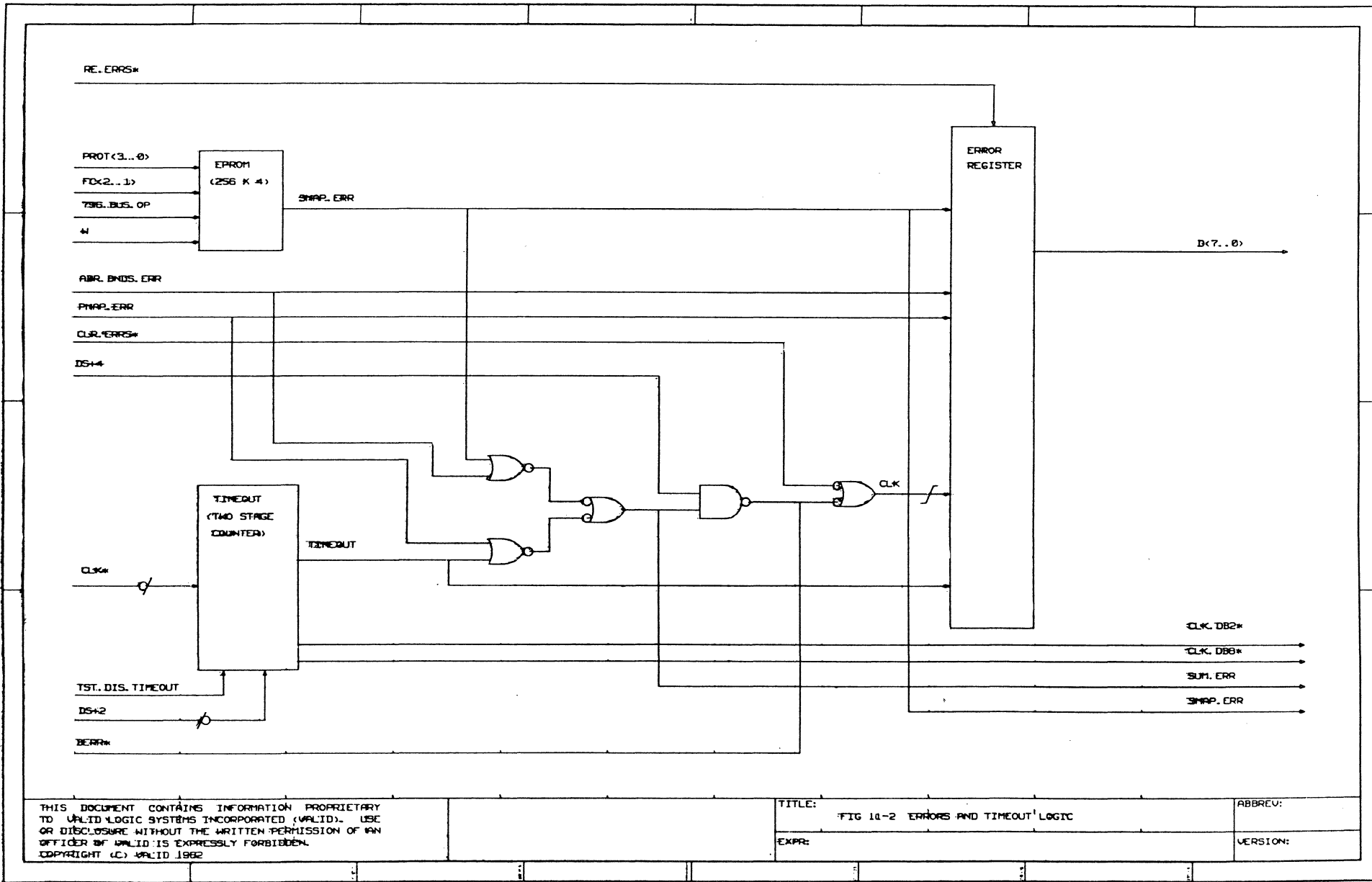
TITLE: FIG 11-1 ERRORS AND TIMEOUT/CPU INTERFACE	ABBREV:
EMPR:	VERSION:

11.2 Errors and Timeout Detailed Operation

The errors and timeout logic is shown in Figure 11-2. The logic consists of a two-state timeout counter, 256 x 4 EPROM, error register, and associated gating. The timeout counters allow time for a response from a 796 bus device. If the response is not received within the allotted time (approximately 15 microseconds), the TIMEOUT line causes the generation of a SUM.ERR to the page map to terminate mapping operations. The DS+2 line clears the timeout second stage counter to its initial state for the next response.

If an error other than timeout occurs (ADR.BNDS.ERR, SMAP.ERR, or PMAP.ERR), data strobe DS+4 enables the gating to clock the error over the BERR* line and also latch it into the error register. The error is then read onto the data bus by the M68000LB via the local commands logic by generating a RE.ERRS* signal to the error register enable. After an error condition is reset, the CLR.ERRS* line clears the latches.

The 256 x 4 PROM (PA) generates an SMAP.ERR to the error logic, error register, and disables the 796 bus commands logic. For test and maintenance purposes the timeout logic can be disabled over the TST.DIS.TIMEOUT* line and the SUM.ERR line monitored for page and segment map and address bounds errors.



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1992

TITLE:	FIG 10-2 ERRORS AND TIMEOUT LOGIC	ABBREV:	
EXPR:		VERSION:	

SECTION 12

RAM AND ROM MEMORIES

The CPU board incorporates both RAM and ROM memories. The RAM is used as a scratch pad memory and the ROM is used for storing monitor programs and self-test diagnostics. The function and features of both memories are described and illustrated in the paragraphs that follow.

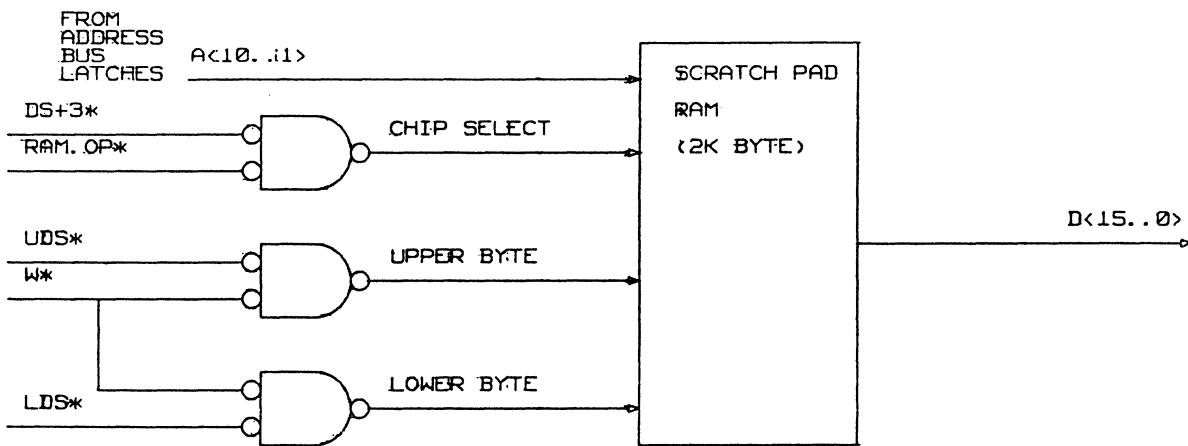
12.1 Scratch Pad RAM/CPU Interface

The scratch pad RAM interface within the CPU board is shown in Figure 12-1. The RAM is made up of four 1K x 4 bit random access memory chips combined to form 2Kbytes of RAM memory. The RAMs are addressed from the M68000LB via the address latches. The chip select input of each RAM is activated by the DS+3 and RAM.OP* select lines from the state generation and DTACK logic and the local commands respectively. Either the upper or lower bytes are selected for use by either the upper data strobe (UDS) or the lower data strobe (LDS). The selected byte is then written-into or read-from depending on the condition of the W* line. A true W* line enables a read cycle, a false W* enables the write cycle.

The memory can be accessed by either a word or byte operation. The cycle time is fast enough so that no wait states are introduced into the M68000LB. The memory allows CPU diagnostics to run without the bus. The operating system can also place small programs such as wait loop in the RAM memory in order to remove instruction-fetch traffic from the bus at critical times.

12.2 ROM/CPU Interface

The erasable-programmable read-only-memory (EPROM) interface within the CPU is shown in Figure 12-2. The EPROMS are used for monitoring and performing CPU on-board diagnostics. The EPROM consists of two plug-in 8K x 8 bit chips making a total of 16Kbytes available. The EPROM is addressed from the address bus latches and enabled by RE.ROM* and DS+3 from the local commands and state generation and DTACK logic. Outputs are provided onto the M68000LB data bus.

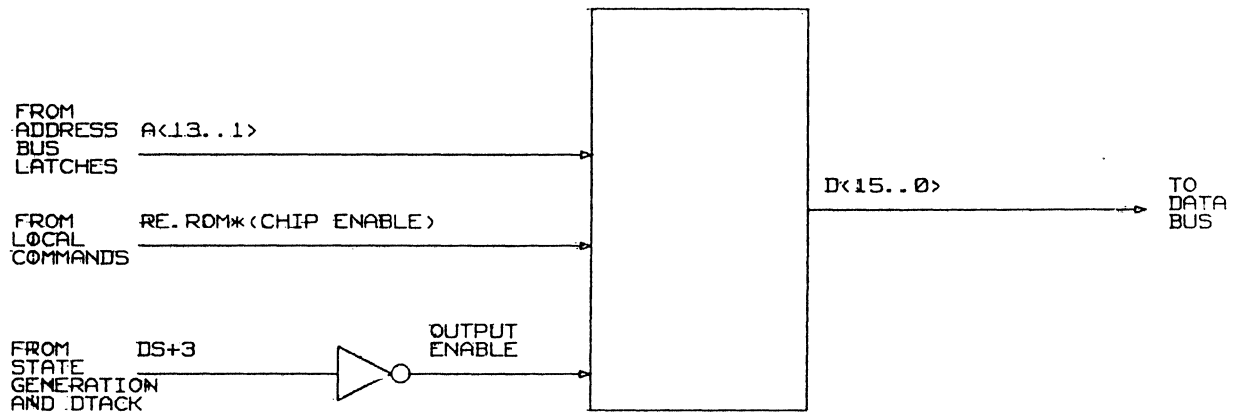


SCRATCH PAD RAM/CPU INTERFACE

THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1982

TITLE: FIG 12-1
SCRATCH PAD RAM
EXPR:

ABBREV:
VERSION:



ROM/CPU INTERFACE

THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN.
 COPYRIGHT (C) VALID 1982

TITLE: FIG 12-2
 ROM/CPU INTERFACE

ABBREV:

EXPR:

VERSION:

SECTION 13

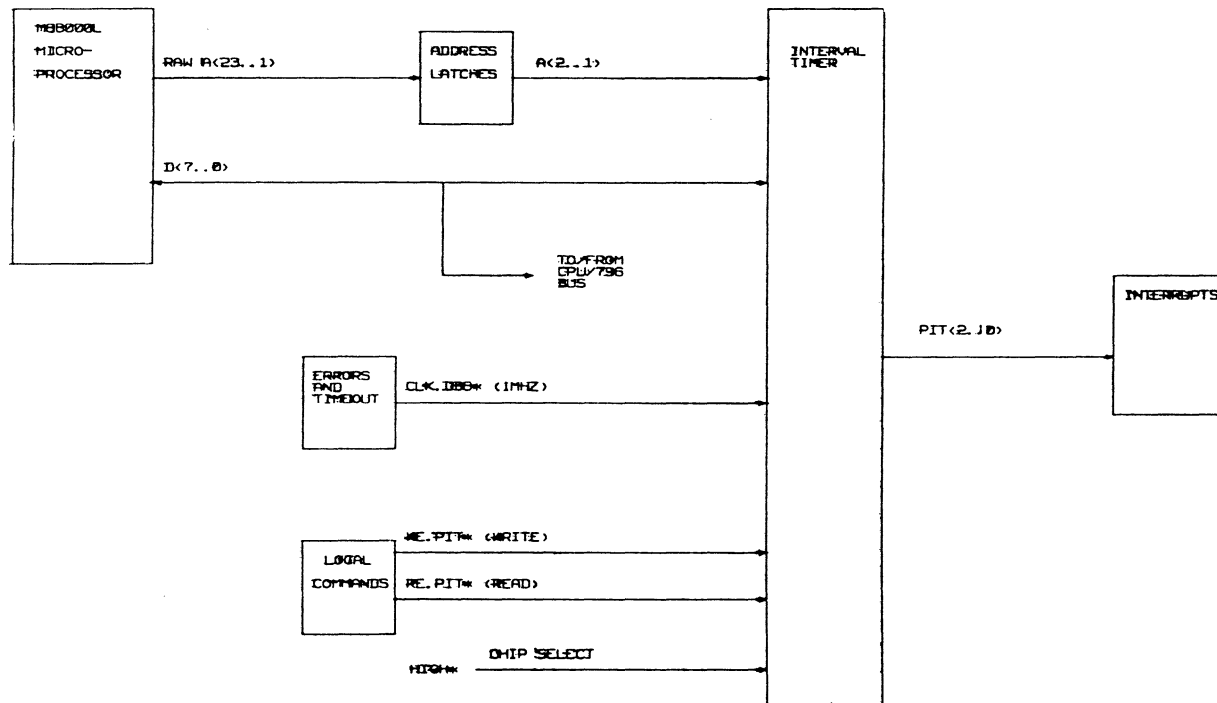
INTERVAL TIMER

13.1 Timer/CPU Interface

The CPU board includes an 8253 programmable interval timer which contains three 16-bit timers, each of which receives a 1MHz clock. Each of the timers output signals can be strapped to interrupt the CPU (see Interrupts, Section 5). The 8253 contains four 8-bit registers accessed by odd byte addresses.

The purpose of the timer is to provide the programmer with the capability of generating accurate time delays. Upon command, one of the counters counts out a delay, then interrupts the M68000LB when the task is completed.

The timer interface within the CPU board is shown in Figure 13-1. Counter selection lines (1 of 3) are supplied by address bus bits A<2..1>. Programmed data for time delays is provided over data bus bits D<7..0>. The 1MHz clock is supplied by a counter in the errors and timeout logic. Time delay values contained on the data bus for a particular counter are read into the timer by the RE.PIT* signal. Upon completion of a time delay, an interrupt is generated and supplied to the CPU interrupts which in turn interrupts the M68000LB depending on how the PIT<2..0> signals are strapped within the CPU interrupt logic.



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1992

TITLE: FIG 13-1 INTERVAL TIMER/CPU INTERFACE
 EXPR:

ABBREV:
 VERSION:

SECTION 14

STATE GENERATION AND DTACK

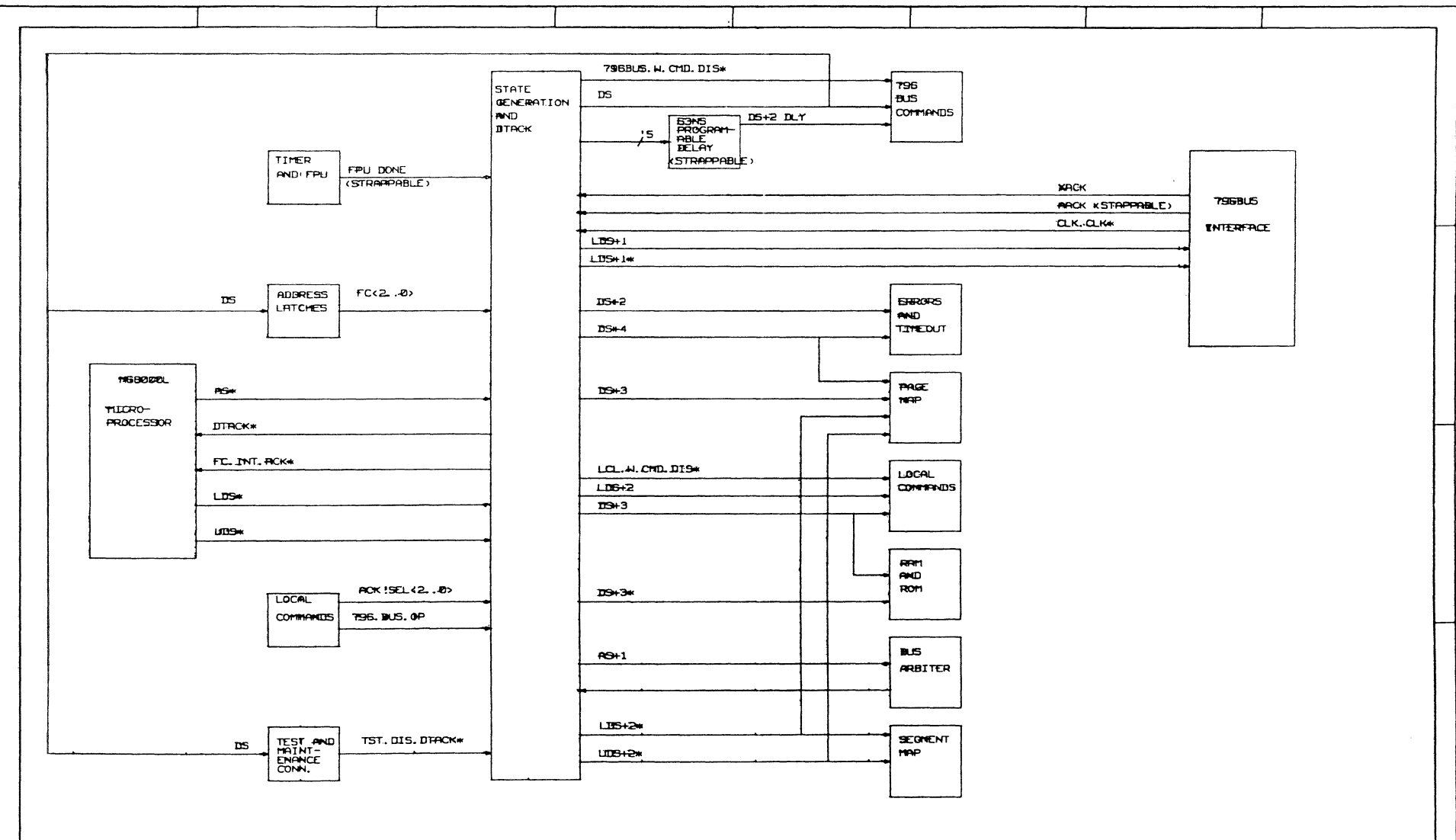
14.1 State Generation/CPU Interface

The state generation and DTACK logic interface within the CPU is shown in Figure 14-1. The logic is generally controlled by input signals from the M68000LB microprocessor, address latches, timer and FPU (if used), local commands, and in the diagnostic mode through the test and maintenance connector. The logic contains various latches, shift register, multiplexers, programmable time delays and associated gating. Certain critical signals are strappable to the time delays as described in the paragraphs that follow.

The M68000LB provides an address strobe (AS*) signifying a valid address on the address bus and the basic lower and upper data strobes (LDS* and UDS*). The state generation and DTACK provides both a function code acknowledge (FC.INT.ACK*) and data acknowledge (DTACK*) to the M68000LB. The function code is ANDed in the state generation logic to provide the acknowledge whenever AS* is active. The DTACK signifies the end of a data transfer and allows the M68000LB to start a new cycle.

The timer and CPU provides a FPU DONE which can be strapped into the DTACK generation logic for FPU operations if included with the CPU (optional item). The ACK.SEL<2..0> lines are applied to the select inputs of the DTACK multiplexer to select which one of the timing or acknowledge signals cause a DTACK output.

The 796 bus supplies the clock signals (CLK and CLK*) required for generation of the timing signals from the state generation logic. The 796 bus also supplies the normal acknowledge (X.ACK) and a special advance acknowledge (AACK). The X.ACK is gated directly to the DTACK multiplexer over the BUS.ACK line. This line acknowledges the M68000LB is ready. The AACK is provided to a time delay where it can be strapped at different times to accommodate system requirements. This line also causes the generation of BUS.ACK to the DTACK multiplexer.



TITLE:	FIG 14-1	ABBREV:	
	STATE GENERATION/CPU INTERFACE		
EXPR:		VERSION:	

THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN.
 COPYRIGHT (C) VALID 1992

14.2 State Signal Generation

All state generation signals are clocked into the CPU function by the clock signal received from the bus. All even numbered signals are generated by the rising edge of the clock and all odd numbered signals are generated by the falling edge of the clock. A command strobe (DS+2 DLY) is also supplied to the 796 bus commands logic as well as the data strobe (DS) and disable signal (796 BUS.W.CMD DIS*). The delay of the command signal is adjustable to fit system requirements. Both the program inputs and the data strobe state generation signals are strappable as shown in Figure 14-2. The program inputs adjust the timed output and the data strobe inputs allow selection of either DS+2 or DS+3 as an input source for the delay.

The data strobe (DS) is used as one of three enables to the 796 bus commands decoder logic. The DS signal is also supplied to the CPU address latches to gating the addresses onto the CPU address bus. The signal is also supplied to the test and maintenance connector for servicing requirements. The 796 BUS.W.CMD.DIS.* is used for disabling the 796 BUS.OP write command (W) to the 796 bus commands.

SECTION 15

FLOATING POINT PROCESSOR

15.1 Floating Point Processor/CPU Interface

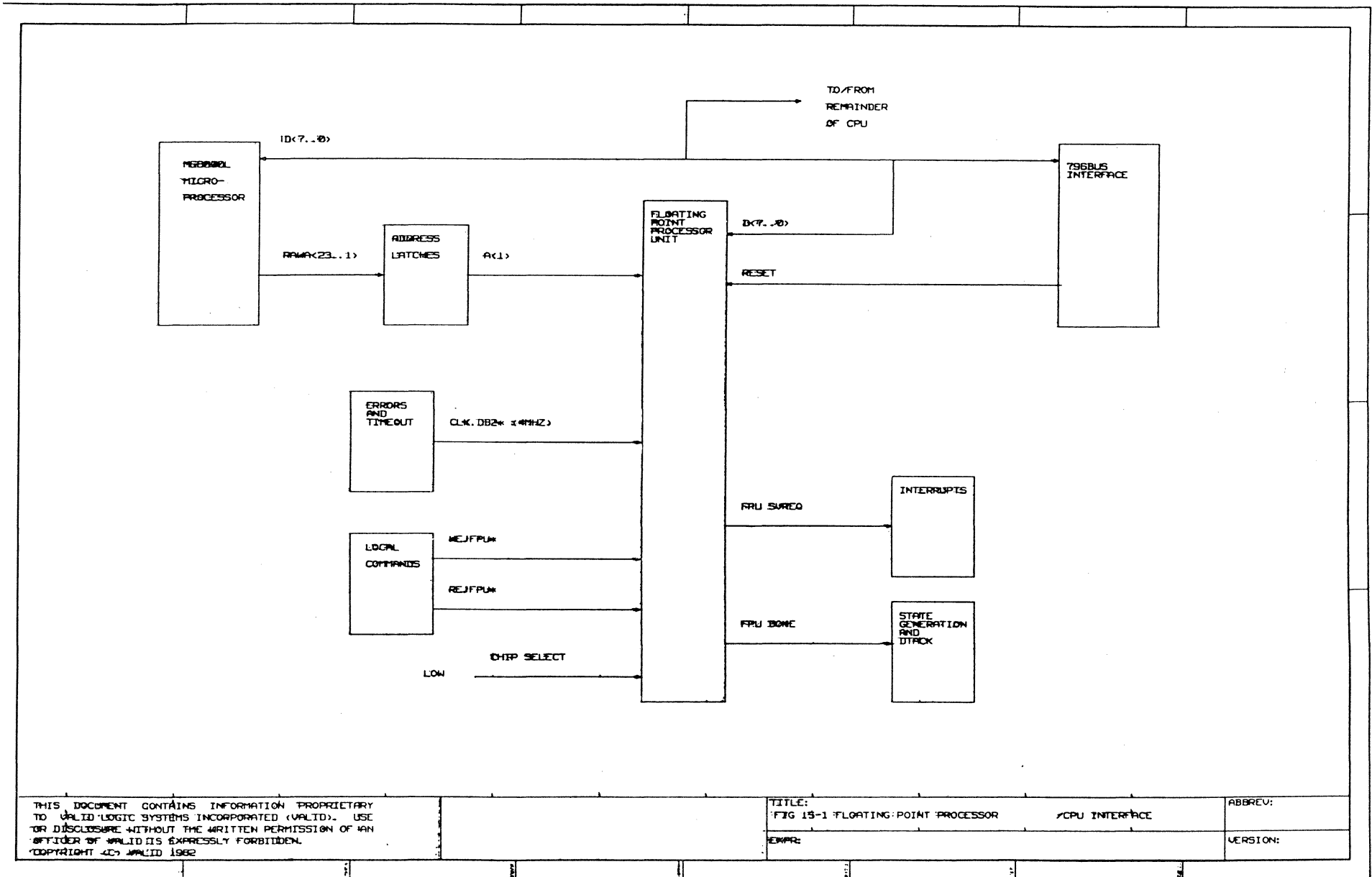
The floating point processor/CPU interface is shown in Figure 15-1. The purpose of the 8232 floating point processor is to enhance the computational capabilities of the CPU M68000LB microprocessor and eliminate the necessity to write and verify floating point software. The processor executes single-precision (32-bit) and double precision (64-bit) add, subtract, multiply, and divide operations at 4MHz clock rate. The processor executed a single-precision floating-point add in 15 microseconds and a multiply in 50 microseconds, making it considerably faster than the M68000LB software.

The operand, result, status and command information for the processor take place over the CPU bi-directional data bus. Operands are placed into the processor and commands to perform an operation are issued by the CPU. Results are provided to the CPU on the data bus.

The floating point processor interface is shown in Figure 15-1. The processor interfaces with the M68000LB via the address latches. These supply the address bit required to perform the operating modes of the processor. The address bit in conjunction with the WE.FPU* and RE.FPU* determine the type of transfer to be performed on the data bus as follows:

A<1>	WE.FPU*	RE.FPU*	Transfer
0	1	0	Enter data byte
0	0	1	Read data byte
1	1	0	Enter command
1	0	1	Read status

The 4MHz clock is supplied from a counter located in the errors and timeout logic over line CLK.DB2. A RESET line from the bus terminates any operation in progress and clears the processor to zero. After completion of an operation, the processor executes an end of execution (FPU.SVREQ) to



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT © VALID 1982

TITLE:
FIG 13-1 FLOATING POINT PROCESSOR /CPU INTERFACE
REV#:

ABBREV:
VERSION:

indicate the completion of a command. This signal is a strappable signal supplied to the CPU interrupts logic.

The FPU DONE also indicates a completion of command to the state generation and DTACK logic. The FPU DONE is cleared by any read or write operation or reset.

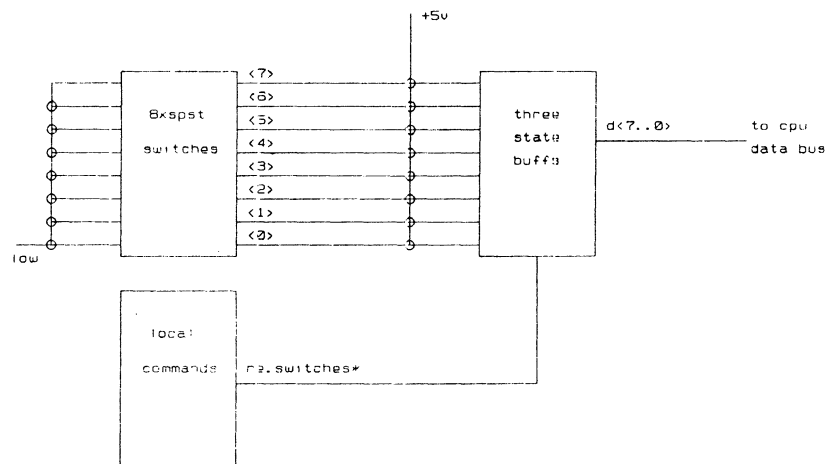
SECTION 16

SWITCHES

16.1 Switches/CPU Interface

A bank of eight dual inline package (DIP) switches are available for use under software control. They read as zero when closed and one when open. They can be used to indicate to mode or configuration of the board. For example, special diagnostics to be executed, etc. Writing this onto the data bus causes a timeout bus trap error.

The switches interface the CPU data bus as shown in Figure 16-1. The configuration of the switches is gated from the three-state buffers by the RE.SWITCHES* input from the local commands. The open switches are pulled up by +5V through a pull up resistor to each switch line. By closing a particular switch, the +5V is clamped to ground producing a ground level onto the respective data bus line when the buffers are enabled.



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1982



TITLE: FIG 16-1

DATE:

ENGINEER:

PAGE:

SECTION 17

UARTS

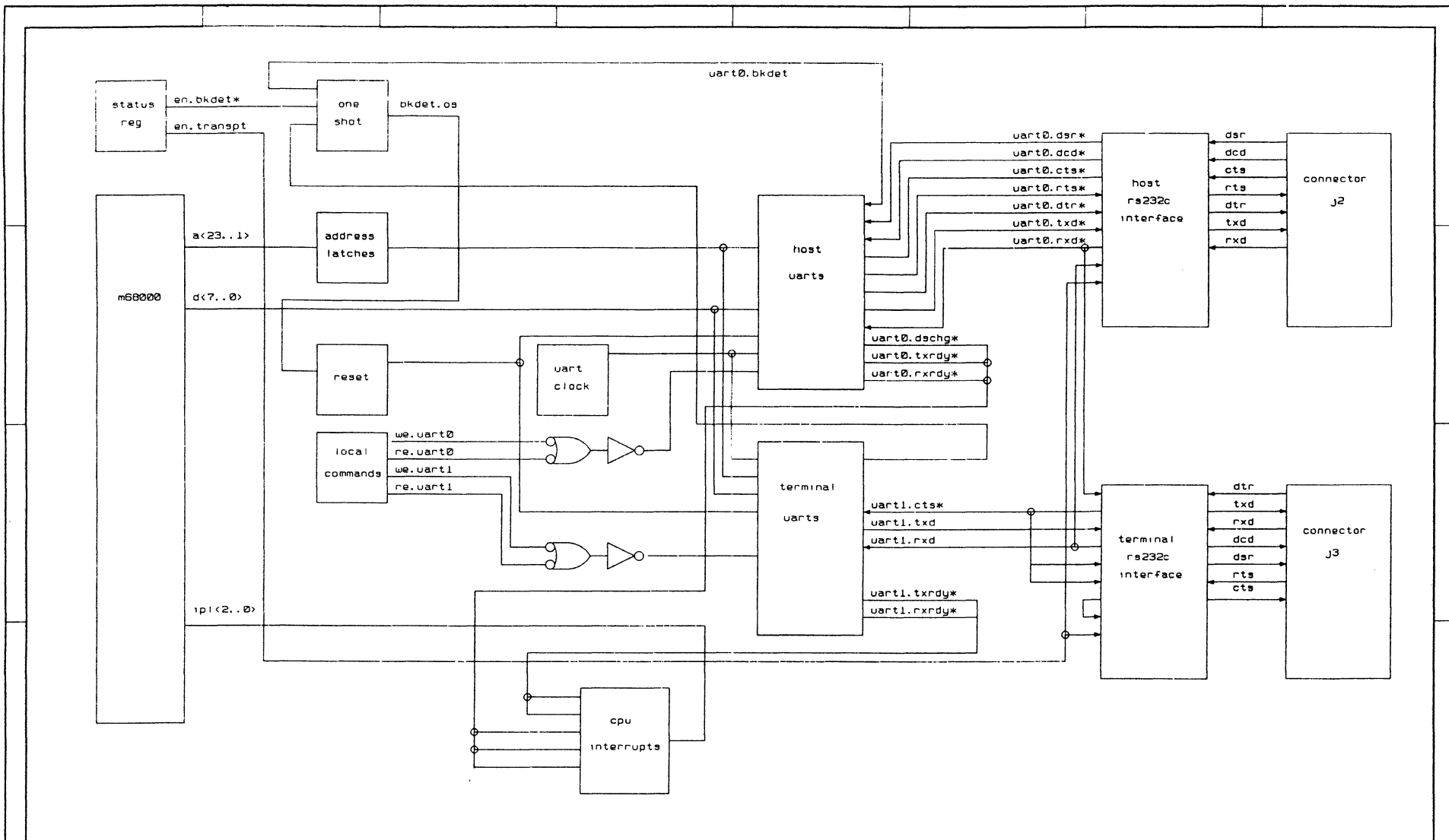
17.1 UARTS/CPU Interface

The CPU board contains two 2662-2 programmable UARTS and associated RS232C drivers and receivers as shown in Figure 17-1. Each UART contains four eight-bit registers accessed by odd byte addresses. Address bits A<2..1> select the internal functions within each UART. The UARTS are capable of transmitting and receiving serial data at a rate up to 38400 baud in a full duplex mode. The two UARTS are designated as host and terminal UARTS. The host UART is wired to communicate with a modem, the second with a terminal. The host UART features full modem control with strappable interrupts. The terminal UARTS is wired to communicate with a terminal. Both the host and terminal UARTS must be operating at the same baud rate.

17.2 UARTS Operation

During operation, if EN.TRANSPT is true in the CPU status register (transparent mode) and both UART transmitters are disabled, the received data from the host RS232C port is routed to the terminal, and vice-versa. This also allows the UARTS to monitor incoming data. This feature allows communication between the terminal and the host with complex software buffering and rate control. This may also be used with Motorola's MACSBUG monitor.

The host or terminal UARTS is addressed over lines from the address latches (A<2..1>) via the M68000LB. The eight data lines D<7..0> are connected to the CPU bidirectional data bus. An external 4.9152 MHz external clock is provided for input to the UARTS internal baud rate generator for synchronizing baud rates between the host and terminal UARTS. The EN.TRANSPT and EN.BKDET* are supplied to the host and terminal interfaces and the UARTS one shot is used for issuing a break detect (BK.DET.OS) pulse to the reset logic. This is used for performing a master reset to both UARTS.



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1982

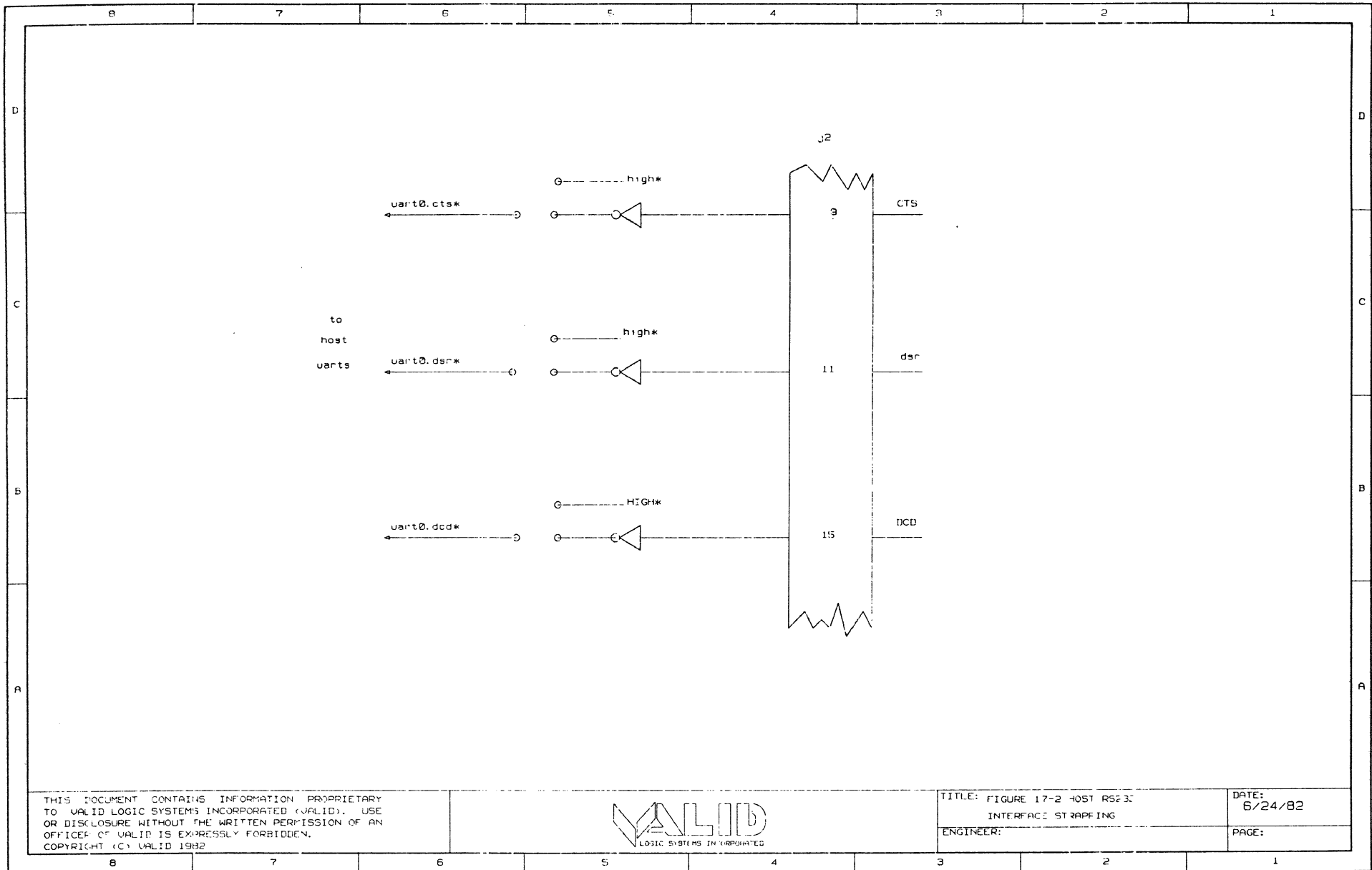


TITLE:	FIG 17-1 UARTS/CPU INTERFACE	ABBREV:	
EXPR:		VERSION:	

The local commands read and write decoders supply both read and write enables (WE.UART 0, RE.UART 0 and WE.UART 1, RE.UART 1) signals to both UARTS chip enable (CE*) inputs for enabling one or both UARTS depending on the operation being performed. Interrupts, which are strappable in the interrupt logic, are supplied to the CPU interrupts. These may be used as required depending on the required configuration of the board. For a more detailed description of the 2661 UARTS, refer to the manufacturer's technical data.

17.3 UARTS Strapping

The host and terminal interface strapping are shown in Figures 17-2 and 17-3. The received signals CTS, DSR, and DCD can be strapped low to allow communications with devices which do not generate those signals.



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1982

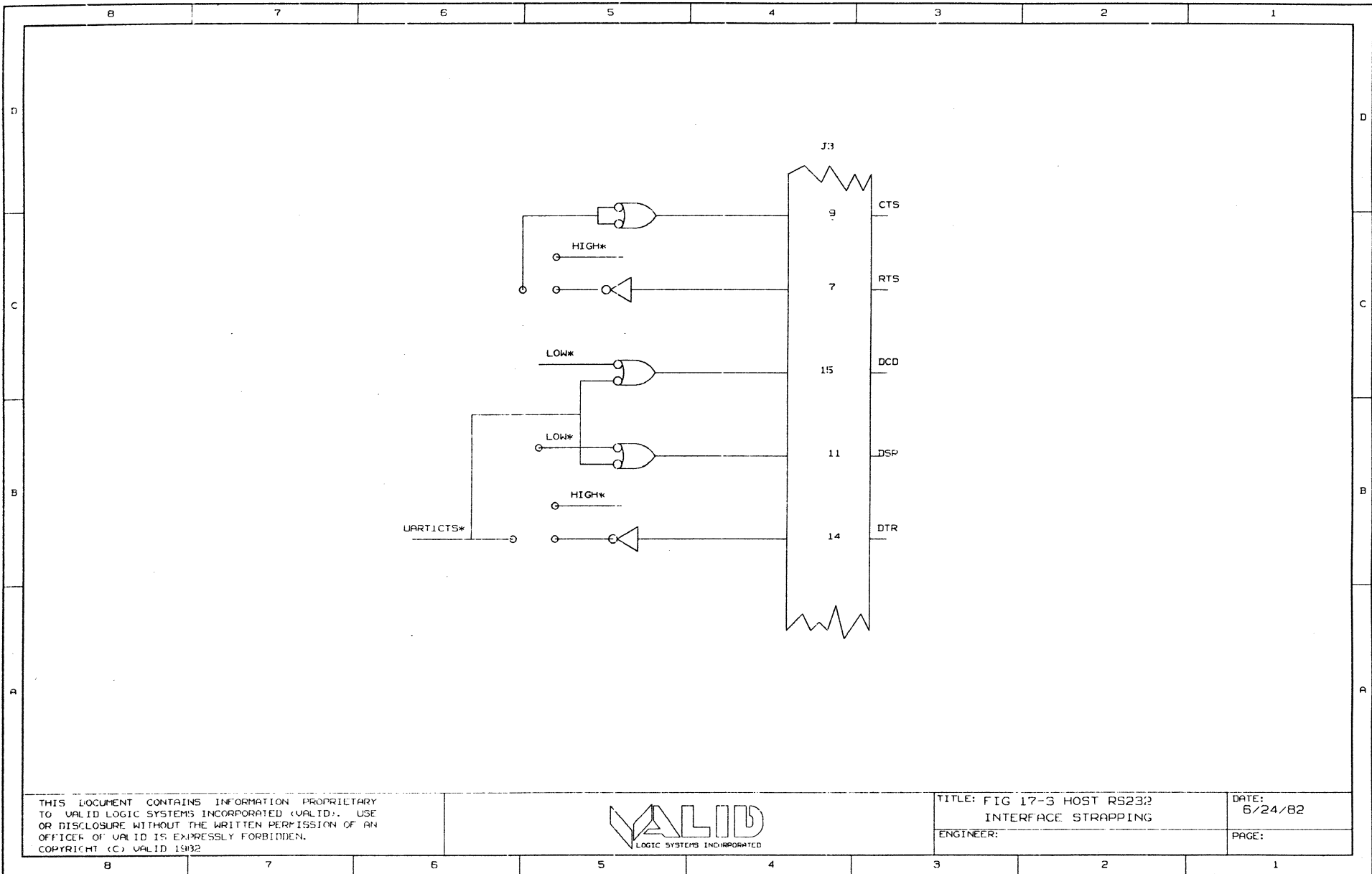


TITLE: FIGURE 17-2 HOST RS232C
INTERFACE STRAPPING

ENGINEER:

DATE:
6/24/82

PAGE:



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1982



TITLE: FIG 17-3 HOST RS232C
INTERFACE STRAPPING
ENGINEER:

DATE: 6/24/82
PAGE:

SECTION 18

CPU RESETS

The CPU reset circuit on the CPU board provides resets to either the CPU board or the 796 bus. The following describes the reset logic within the CPU as well as the detailed operation of the logic used to generate individual resets.

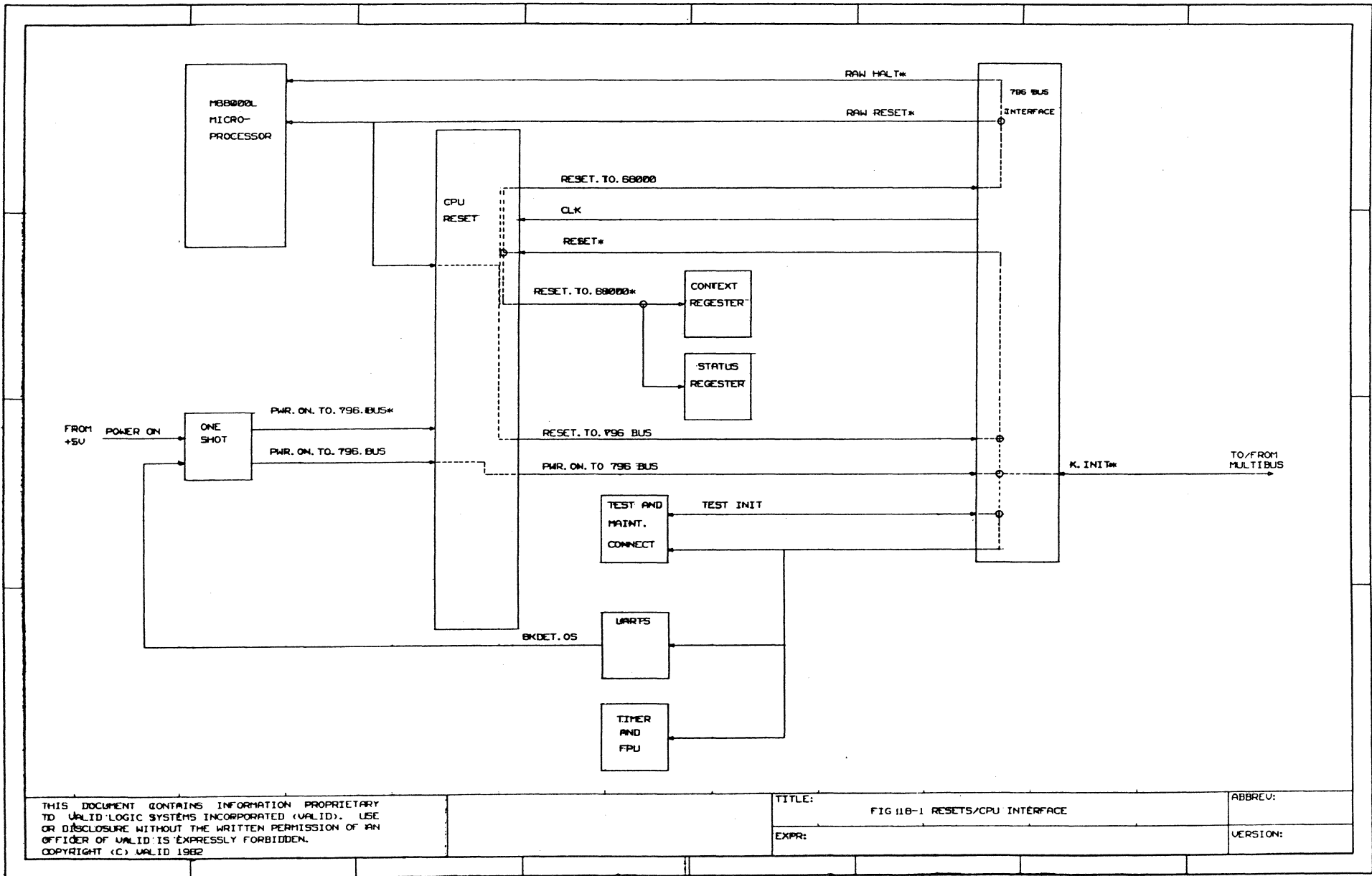
18.1 Resets/CPU Interface

The resets/CPU interface is shown in Figure 18-1. The reset function consists of the CPU reset logic and a one shot which generates a master clear to the reset logic, M68000LB, CPU functions, and 796 bus devices. The one shot generates the clear signal upon receipt of a CPU power-up or a break detect signal from the UARTS. The reset logic receives a RESET* input from the bus and a RAW RESET* from the M68000LB. The RAW RESET* line causes the reset logic to generate a RESET.TO.796 BUS signal to the bus which then sends a reset onto the bus (X.INIT*) line. The RESET.TO.796 BUS line also sends a RESET to the UARTS and timer and FPU to set those functions to their initial states. The signal can also be monitored over the test and maintenance connector.

A reset can also be initiated from the bus to reset the CPU logic exclusively. This is received over the X.INIT* line which in turn generates both the RESET and RESET* signals. The RESET line provides a reset to the UARTS, timer and FPU circuits, and the test and maintenance connector for monitoring purposes. The RESET* line is applied to the reset logic which in turn generates a RAW HALT* and a RAW RESET* to the M68000LB and a RESET.TO.68000* used to reset the context and status registers. The RESET* line can also be accessed through the test and maintenance connector for service and testing purposes.

18.2 Resets Detailed Operation

The logic circuits for the CPU reset function with operational states is shown in Figure 18-2. Shown are the initial power-up state, reset from the M68000LB, and reset initiated from the bus.



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1982

TITLE: FIG 118-1 RESETS/CPU INTERFACE
 EXPR:

ABBREV:
 VERSION:

18.2.1 Power On Reset

The power on reset is started when power is applied to the CPU board. The +5V level is sensed by the one-shot as shown in Figure 18-2, example A. The one-shot generates a clear pulse to the reset logic latches and a true level over the PWR.ON.TO.796 BUS line to the bus interface logic. This sets the latches and bus devices to their initial state. At the same time, the power-cn state also generates a RESET signal to clear the UARTS and timer and FPU and a RESET* back to the reset logic. The RESET* line the reset logic (Figure 18-2, example B) to generate a RAW HALT* and a RAW RESET* to the M68000LB to place it in its initial state.

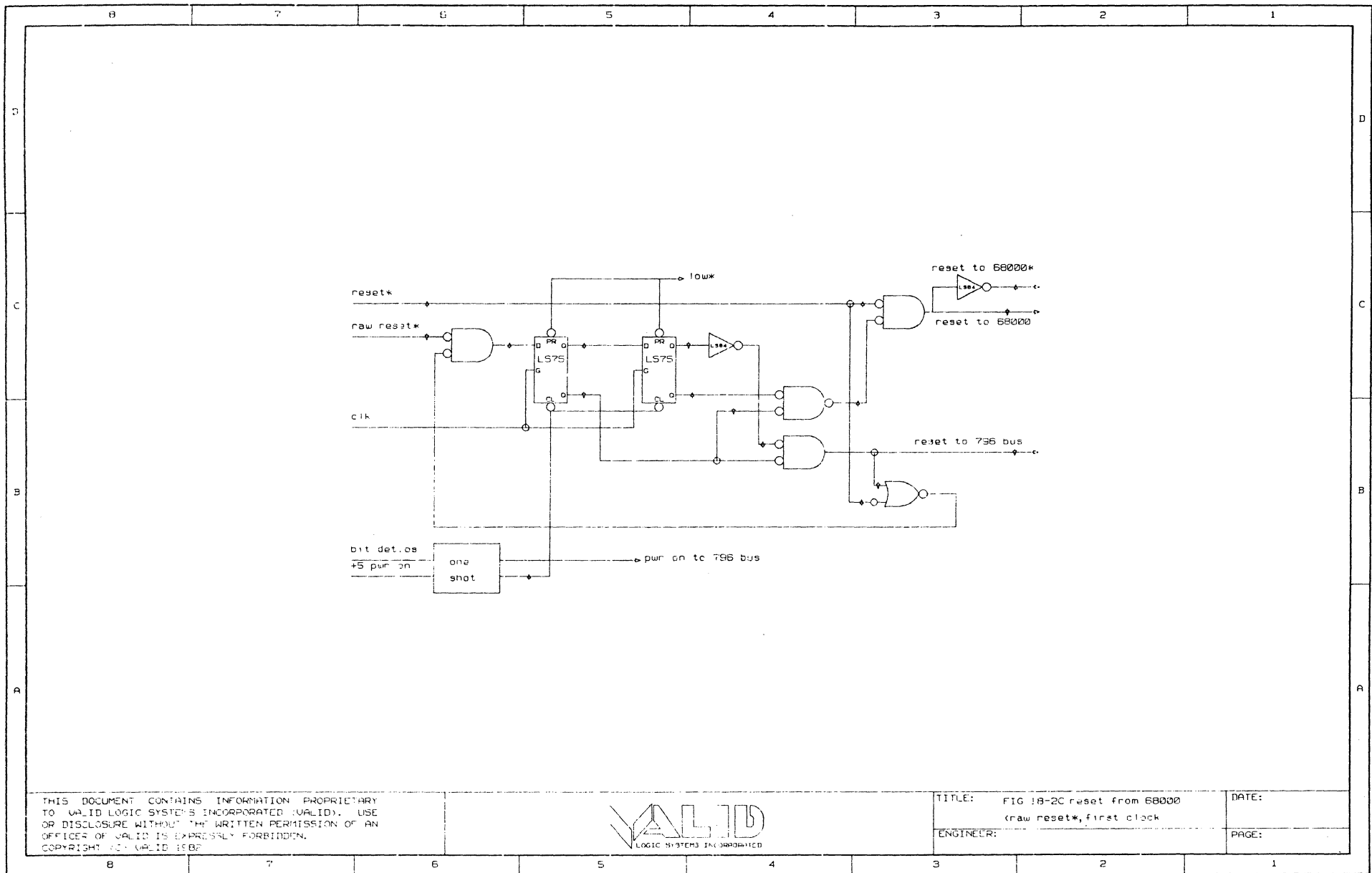
18.2.2 Reset From M68000LB

The states of the reset logic for generating a RESET.TO.796 BUS when a reset is selected from the M68000LB are shown in Figure 18-2, examples C and D. The generation of the reset is accomplished in two successive positive clock transitions. The first transition, upon receipt of a RAW RESET*, clocks the resulting high through the first D latch onto the D input of the second latch. The second positive going clock pulse then clocks the high from the first D latch and to the output of the second latch. These lines (Q) in addition to the latch complementary outputs (Q) provides gating to block any M68000LB resets and also generate a RESET.TO.796 BUS signal to the bus interface. At the same time, a feedback line to the RAW RESET* input gate is set to a true condition. Therefore, after the RAW RESET* condition is clear (goes false), a low is generated to the first D latch input. Each subsequent clock pulse then transfers the low through the latches to set them to their initial states preparing them for the occurrence of another reset condition.

18.2.3 Reset From Bus

A reset initiated from the bus is received on the RESET* line as shown in Figure 18-2, example E. Upon receipt of a RESET* true signal, two false levels are applied to the AND gate that controls M68000LB resets. A false level is also applied to the NOR gate which in turn sends a true level to the RAW RESET* gate to block any attempt to reset from the M68000LB. The

resulting RESET.T0.68000 is generated by the two lows at the reset gating. The line is also inverted to provide a RESET.T0.68000* level which is used by the CPU board to reset the context and status registers. The RESET.T0.68000 causes the generation of both the RAW HALT* and RAW RESET* to the M68000LB as described in previous paragraphs. After the M68000LB and registers reset is complete, the RESET* line goes true thus resetting the logic to its original state.



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT © VALID 1987



TITLE: FIG 1B-2C reset from 68000
(raw reset*, first clock)
ENGINEER:

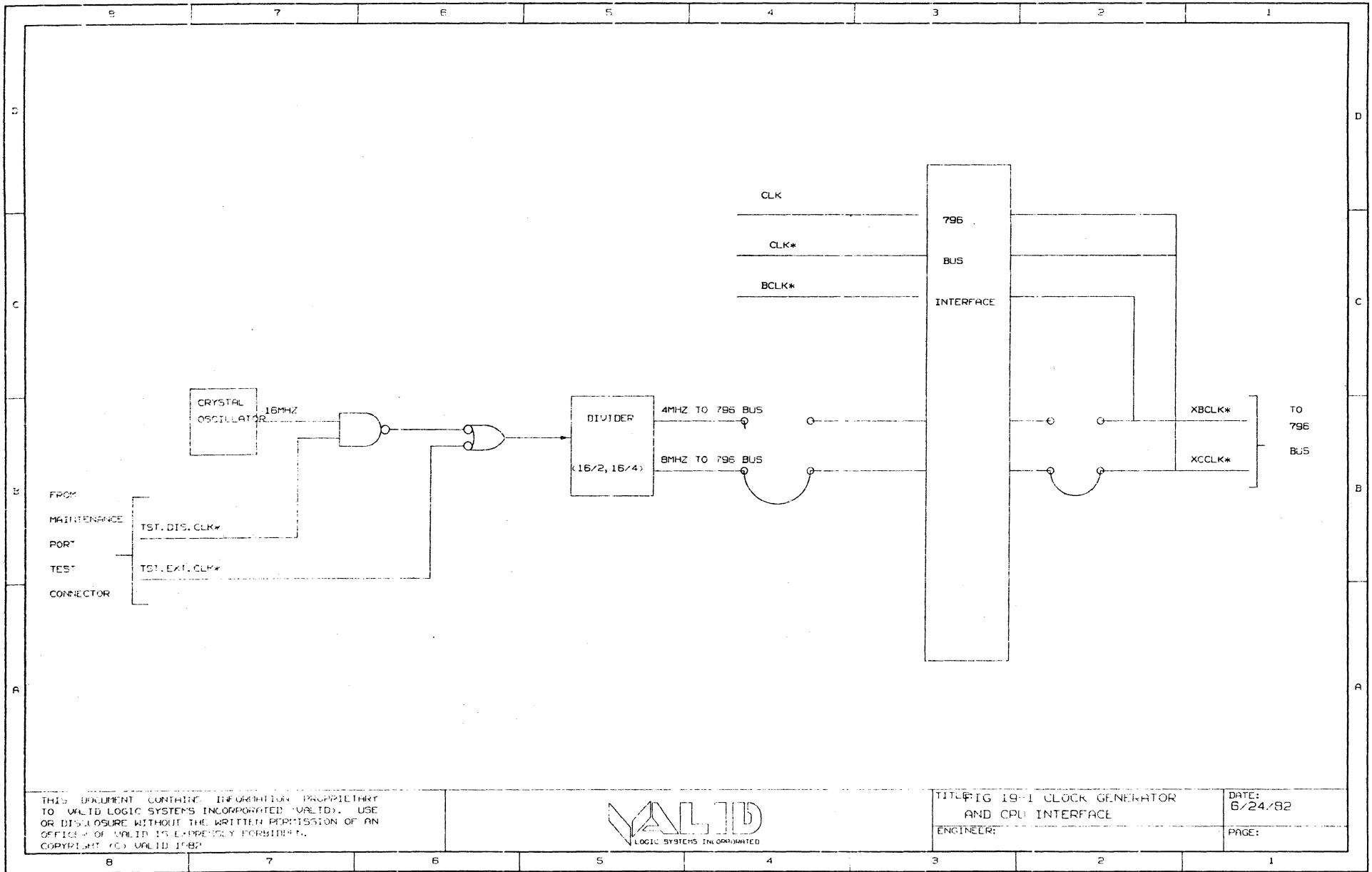
DATE:
PAGE:

SECTION 19 CLOCK GENERATOR

19.1 Clock Generator/CPU Interface

The clock generator interface with the 796 bus and CPU is shown in Figure 19-1. The clock circuits are located on the CPU board. The features of the clock include a crystal oscillator, gating, and two dividers to produce both 4MHz and 8MHz outputs from the basic oscillator frequency of 16MHz. The two outputs are then provided to the bus interface. Strapping is provided to select the required frequency. The selected clock output is placed on the bus from the interface for use by other bus users. The clock is also routed back into the CPU board. This provides a common clock source as well as an equal, stable clock input to all users and the CPU as further described in the 796 bus description in this manual (Section 6).

For test and maintenance purposes, inputs are provided to the oscillator gates for disabling the oscillator and injecting a test clock input signal directly to the divider network.



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1982



TITLE: FIG 19-1 CLOCK GENERATOR AND CPU INTERFACE

ENGINEER:

DATE: 6/24/82

PAGE:

SECTION 20 MAINTENANCE PORT

20.1 Maintenance Port/CPU Interface

The maintenance port interface is shown in Figure 20-1. The port consists of connector J1 and fully buffered test signals used to exercise and monitor various functions of the CPU board for maintenance purposes. One advantage of the port is that the board can be single stepped or clocked with an external clock. The buffers are continuously enabled eliminating any special gating or enabling signals for test purposes. The signals to and from the various functions of the CPU board and their particular purpose are listed in table 20-1.

Table 20-1. Maintenance Port Interface Signals

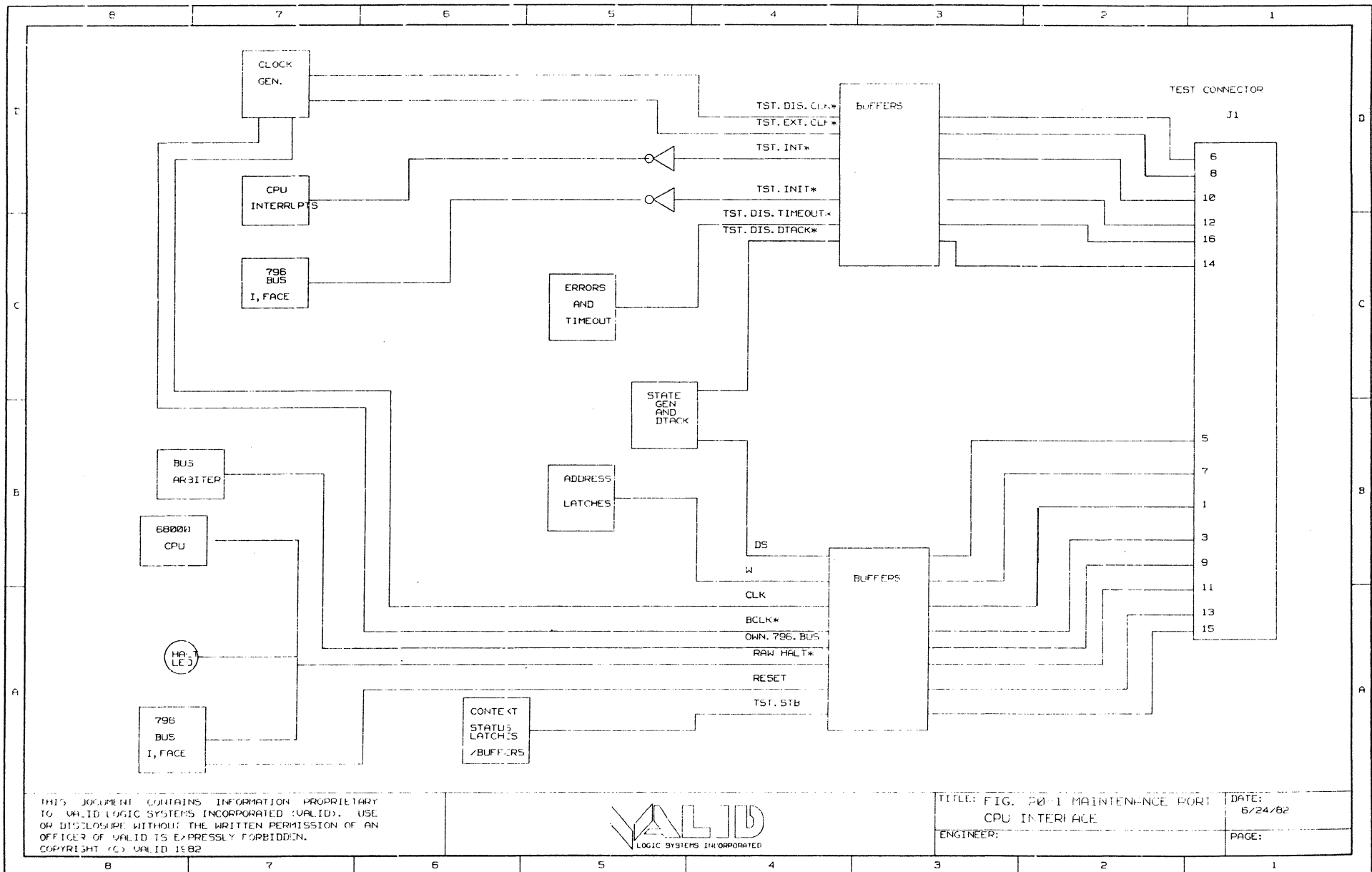
SIGNAL	PURPOSE
TST.DIS.CLK*	Disables clock 16MHz oscillator.
TST.EXT.CLK*	Provides path to clock generator divider for external clock signal after oscillator is disabled by TST.DIS.CLK*.
TST.INT*	Test interrupt line to CPU interrupts (strap).
TST.INIT*	Test line to bus to reset entire system. Generates RESET and *RESET on CPU board.

Table 20-1. Maintenance Port Interface Signals (Continued)

SIGNAL	PURPOSE
TST.DIS.DTACK*	Disable CPU data acknowledge (DTACK*) line to M68000LB.
TST.DIS.TIMEOUT*	Disable errors and timeout logic timeout counter.
CLK	Provide monitoring of M68000LB clock output.
BCLK*	Provide monitoring of bus clock output.
DS	Provide monitoring of state generation and DTACK data strobe output.
W	Allow monitoring of write strobe from address latches.
OWN.796 BUS	Allow monitoring of bus address buffers enable signal from CPU bus arbiter.
RAW.HALT*	Provide monitoring of CPU and bus halt signal and exercise HALT LED.

Table 20-1. Maintenance Port Interface Signals (Continued)

SIGNAL	PURPOSE
RESET	Allow monitoring of reset from bus.
TST.STB	Used as test status bit to check operation of status register.



THIS DOCUMENT CONTAINS INFORMATION PROPRIETARY TO VALID LOGIC SYSTEMS INCORPORATED (VALID). USE OR DISCLOSURE WITHOUT THE WRITTEN PERMISSION OF AN OFFICER OF VALID IS EXPRESSLY FORBIDDEN. COPYRIGHT (C) VALID 1982



TITLE: FIG. 20-1 MAINTENANCE PORT CPU INTERFACE
 ENGINEER:

DATE: 6/24/82
 PAGE:

DIAGNOSTICS

VALID RESIDENT MONITOR (VRM)

This document is intended to enable the user of a Valid Logic Systems Inc. M68000L8 CPU board to execute the on board diagnostics provided on prom.

To obtain a menu of commands the operator must use a terminal connected to the console port on the CPU. By typing a "?" menu of commands will be displayed.

Menu Of Commands

COMMAND	DESCRIPTION	SYSTEM ACTION
?	Print this menu	The list of commands will be printed.
-	Store a byte	A byte of data which has been specified by the user will be written in a user selected location.
=	Store a word	A word of data which has been specified by the user will be written in a user selected location.
L	Load address register from the accumulator	The address entered on the console, will be loaded into the address register.
T	Display this byte	The byte of data at the address selected by the user will be printed on the console.
N	Display next byte	The byte of data which follows the address selected will be printed on the console.
P	Display previous byte	The byte of data prior to the address selected will be printed on the console.
J	Restore and jump	The CPU will jump to an address specified by the user.
Z	Zero accumulator	Entries made on the console will be cancelled.

*	Display this BAB location	This is a special command used by Valid system products.
)	Display next BAB location	Same as above
(Display previous BAB location	Same as above
X?.	Print menu of basic extended commands	A list of extended commands will be printed on the console.
X?TST.	Print menu of diagnostic commands	A list of diagnostic commands will be printed on the console.
X?BAB.	Print menu of BAB commands	Special commands for Valid system products.
X?DK.	Print menu of disk commands	A list of disk drive commands will be printed on the console. These may or may not be applicable to disk drives other than those used by Valid system products.
X?TP.	Print menu of tape commands	A list of tape drive commands will be printed on the system console.

DIAGNOSTIC COMMANDS

COMMAND	DESCRIPTION	ACTION
XLLAT.	Loop on logical address (write or read)	System will loop on an address selected by the user.
XLRT.	Execute local ram diagnostic (destructive)	System will execute a test of the local RAM. Data in The local RAM will be destroyed.
XMEMT.	Execute memory test	System will execute a write read test of user specified memory using a semi-random data pattern. The test will prompt the user for test limits.
XONET.	Execute one test	System will execute only the test selected by the user.
XALLT.	Execute all tests	System will execute all tests which have a single digit as their test number.

TEST NUMBERS

1	Context register
2	Switches
3	Segment map
4	Page map
5	Timer
6	Uarts
201	Fixed memory area
203	Status Register
204	User specified memory
205	Floating point unit

TEST COMMAND FORMAT

The following format must be used to execute VRM tests:

nnnnnnLeeeeeLtttLrrrXONET.

Where nnnnnn=The starting address to be tested.

eeeeee=The ending address plus 1.

L=The load command (address register)

ttt=The test number of the test to be executed.

rrr=The repetition count.

XONET.=The appropriate command.

NOTE

All extended commands must
be terminated by a "."

No carriage return.

EXAMPLES

20000L100000L204L5XONET.

This will execute the memory diagnostic from address HEX 20000 to
HEX FFFFF five (5) times. The test output will be as follows:

```
000001P      000000F
000002P      000000F
000003P      000000F
000004P      000000F
000005P      000000F
```

done

This indicates 5 successful passes and 0 failures. The first
column
indicates the number of cumulative successful passes. The second
column
indicates the cumulative number of failing passes.

Abstract:

Describes the full strapping options for the Valid CPU Board 710-00001 Revision A, including the default strap configuration.

Notes:

1. Straps marked with "X" are connected by traces on the PC board.
2. Straps marked with "*" are connected using blue w/w wire during assembly.
3. Each "=" represents a single possible connection from the one pin on the left to any one of the pins on the right.

SHT	LOC	PIN	DESCRIPTION
3	16BA	1	M68000 INTERRUPT LEVEL 7 INPUT
		2	M68000 INTERRUPT LEVEL 6 INPUT
		3	M68000 INTERRUPT LEVEL 5 INPUT
		4	M68000 INTERRUPT LEVEL 4 INPUT
		5	M68000 INTERRUPT LEVEL 3 INPUT
		6	M68000 INTERRUPT LEVEL 2 INPUT
		7	M68000 INTERRUPT LEVEL 1 INPUT
3	16BB	1	HOST/CONSOLE BREAK-DETECT INTERRUPT
		2	MAINTENANCE-CONNECTOR INTERRUPT
		3	FPU-SERVICE-REQUEST INTERRUPT
		4	TIMER<2> INTERRUPT
		5	TIMER<1> INTERRUPT
		6	TIMER<0> INTERRUPT
3	16BC	1	CONSOLE UART TXRDY INTERRUPT
		2	CONSOLE UART RXRDY INTERRUPT
		3	HOST UART TXRDY INTERRUPT
		4	HOST UART RXRDY INTERRUPT
		5	HOST UART DSCHG INTERRUPT
3	16BD	1	MULTIBUS INTERRUPT LEVEL 0
		2	MULTIBUS INTERRUPT LEVEL 1
		3	MULTIBUS INTERRUPT LEVEL 2
		4	MULTIBUS INTERRUPT LEVEL 3
		5	MULTIBUS INTERRUPT LEVEL 4
		6	MULTIBUS INTERRUPT LEVEL 5
		7	MULTIBUS INTERRUPT LEVEL 6
		8	MULTIBUS INTERRUPT LEVEL 7
6	2T	7 = 8	DISABLE AACK
		6 *	ENABLE AACK WITH ZERO ON-BOARD DELAY

		5		ENABLE AACK WITH 12T TAP #1 DELAY
		4		ENABLE AACK WITH 12T TAP #2 DELAY
		3		ENABLE AACK WITH 12T TAP #3 DELAY
		2		ENABLE AACK WITH 12T TAP #4 DELAY
		1		ENABLE AACK WITH 12T TAP #5 DELAY
6	7L	2 = 3	*	ENABLE FPU
		1		DISABLE FPU
6	4T	2 = 1	*	TIME MULTIBUS CMD FROM (DS+2)+63NS
		3		TIME MULTIBUS CMD FROM (DS+3)+63NS
6	6T	1 = 6	*	ADVANCE MULTIBUS CMD BY 32NS
		2 = 5	*	ADVANCE MULTIBUS CMD BY 16NS
		3 = 4		ADVANCE MULTIBUS CMD BY 8NS
6	11H	2 = 1	*	END MULTIBUS WRITE COMMANDS EARLY
		3		DO NOT END MULTIBUS WRITE CMDS EARLY
10	1A	2 = 1	*	ENABLE SOFTWARE MULTIBUS LOCK
		3		DISABLE SOFTWARE MULTIBUS LOCK
10	11D	2 = 1	*	ENABLE CBRQ
		3		DISABLE CBRQ
10	21D	2 = 3		FORCE CPU TO BE HEAD OF BPRN CHAIN
13	23M	2 = 1	*	RECEIVE AACK ON P1-25
		3		RECEIVE AACK ON P2-40
13	23B	1 = 2		EXTRA PULLUP ON BUSY
		3 = 4		EXTRA PULLUP ON CBRQ
		5 = 6		EXTRA PULLUP ON INIT
14	19B	2 = 1	*	ENABLE DRIVING CCLK
		4 = 3	*	ENABLE DRIVING BCLK
14	21B	2 = 1	*	8MHZ CCLK (IF DRIVING CCLK IS ENABLED)
		3		4MHZ CCLK (IF DRIVING CCLK IS ENABLED)
		4 = 1	*	8MHZ BCLK (IF DRIVING BCLK IS ENABLED)
		3		4MHZ BCLK (IF DRIVING BCLK IS ENABLED)
14	19A	2 = 1	*	USE BCLK AS MAIN INTERNAL CLOCK
		3		USE CCLK AS MAIN INTERNAL CLOCK
16	13E	2 = 3	X	1KX4 MAP
		1		4KX4 MAP
16	11E	2 = 3	X	1KX4 MAP
		1		4KX4 MAP
16	9E	2 = 3	X	1KX4 MAP
		1		4KX4 MAP

17	11F	2 = 3 1	X	1KX4 MAP 4KX4 MAP
17	13F	2 = 3 1	X	1KX4 MAP 4KX4 MAP
17	15F	2 = 3 1	X	1KX4 MAP 4KX4 MAP
17	17F	2 = 3 1	X	1KX4 MAP 4KX4 MAP
22	3K	2 = 1 3	*	IGNORE HOST CTS ENABLE HOST CTS
22	1K	2 = 1 3	*	IGNORE HOST DSR ENABLE HOST DSR
22	1H	2 = 1 3	*	IGNORE HOST DCD ENABLE HOST DCD
23	3M	2 = 1 3	*	IGNORE CONSOLE RTS ENABLE HOST DSR
23	1M	2 = 1 3	*	IGNORE CONSOLE DTR ENABLE CONSOLE DTR
26	23N	2 = 1 4 = 3	* *	DRIVE A<23> ON P2-56 DRIVE A<23> ON P2-60
26	23T	2 = 1 4 = 3	* *	DRIVE A<22> ON P2-55 DRIVE A<22> ON P2-59

Interrupts:

Any M68000 interrupt input (16BA) can be connected to any number of interrupt outputs (16BB, 16BCC, or 16BD).

UNIX interrupt strapping:

68000 BUS

L7		16BA1 =	
L6		* 16BA2 = 16BB6	PIT<0>
L5	L2	* 16BA3 = 16BD3	INTERPHASE FUJITSU CONTROLLER
L4	L3	* 16BA4 = 16BD4	RIMFIRE CONTROLLER
L3		* 16BA5 = 16BC1, 16BC2	CONSOLE TXRDY, RXRDY
L2	L5	* 16BA6 = 16BD6	UTB
L1	L6	* 16BA7 = 16BD7	VGB, PIB

Command-strobe tuning:

For operation with the Valid ECM, tune X.MRDC* falling edge to occur on the Multibus between 20ns and 12ns before the falling edge of CLK internal to the CPU board.

Switches:

The switches at location SW are used by VRM. Set the switches as specified below. "ON" is "0" and "OFF" is "1".

SWITCH NUMBER		SWITCH MEANING
<7..8>		0 => ILLEGAL
		1 => TRANSFER TO \$18000 AFTER RESET TRAP
	*	2 => TRANSFER TO VRM AFTER RESET TRAP
		3 => AUTOBOOT AFTER RESET TRAP
6		0 => DO NOT RUN DIAGNOSTICS AFTER RESET
	*	1 => RUN DIAGNOSTICS AFTER RESET
5	*	0 => INITIALIZE 4MBY MAP
		1 => INITIALIZE 16MBY MAP
4		0 => DO NOT USE MULTIBUS MEMORY
	*	1 => OK TO USE MULTIBUS MEMORY

PROMs:

LOC	TYPE	CHECK	NAME
10A	27S21	\$0260	PA
8M	27S185	\$0BA4	PB1K
10M	27S29	\$AB66	PC
14B	27S19	\$017C	PD

EPROMs:

LOC	TYPE	CHECK	NAME
16N	2764-3	\$D04E	VRM/5.5/9-18-82 H
20N	2764-3	\$2951	VRM/5.5/9-18-82 L

NOTE: EPROMs 2764-4 are also acceptable to use

4

3

2

1

NOTE: THIS PAGE PROVIDES A RUNNING HISTORY OF ALL CHANGES ASSOCIATED WITH THIS DWG

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED

REV	SHEETS	DESCRIPTION	DATE	APPROVED	REV	SHEETS	DESCRIPTION	DATE	APPROVED
A		AS ISSUED PER ECO #179	1-13-84	<i>[Signature]</i>					

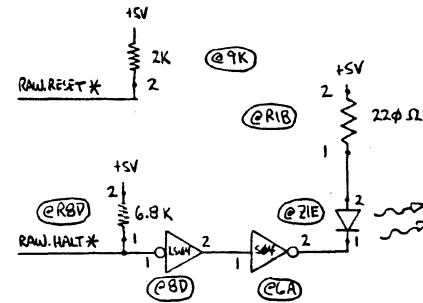
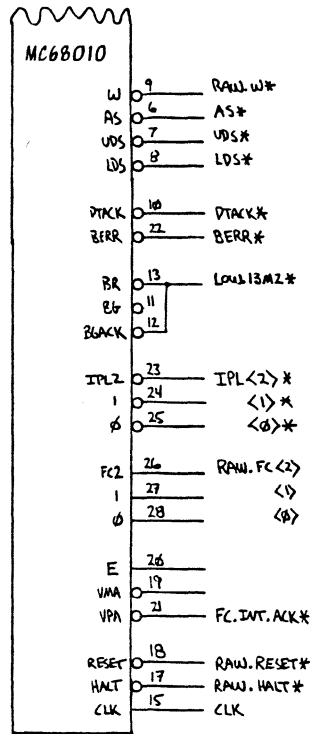
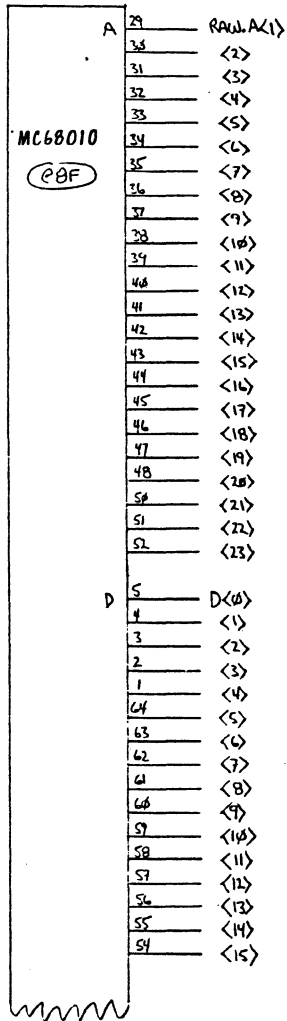
UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± ± .XXX ± ±		CONTRACT NO.	
MATERIAL		APPROVALS	DATE
FINISH		DRAWN <i>WIDDOES</i>	1/13/84
DO NOT SCALE DRAWING		CHECKED <i>[Signature]</i>	1/13/84
		<i>[Signature]</i>	1/13/84
		SIZE B	CODE IDENT NO. REV A
		DRAWING NO. 710-00035-2	
		SCALE NONE	SHEET 1 OF 20

4

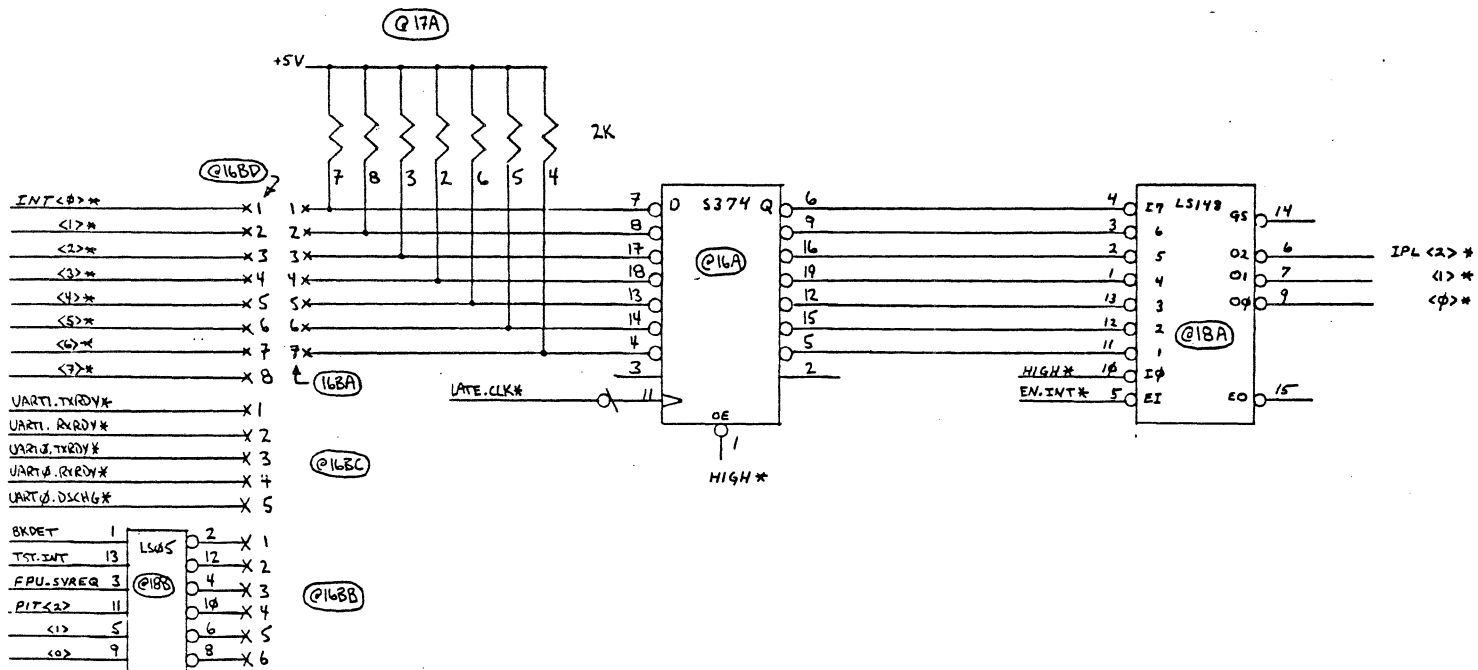
3

2

1

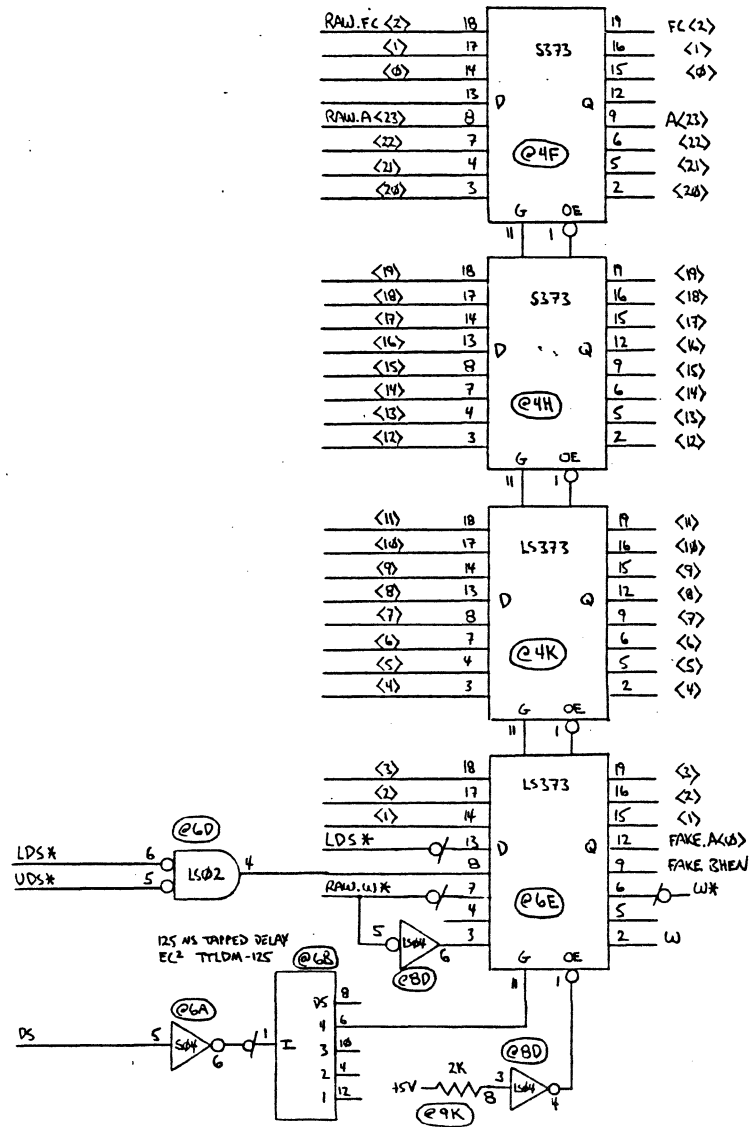


VALID LOGIC SYSTEMS		
SCALE:	APPROVED BY:	DRAWN BY: WIDDOES
DATE: 11/5/81		REVISED:
SCHEMATIC CPU VIRTUAL MEMORY		
CODE IDENT NO REV. A	SHEET 2 OF 26	DRAWING NUMBER 710-00035-2

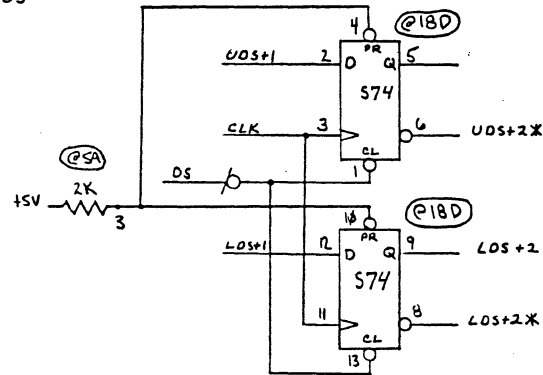
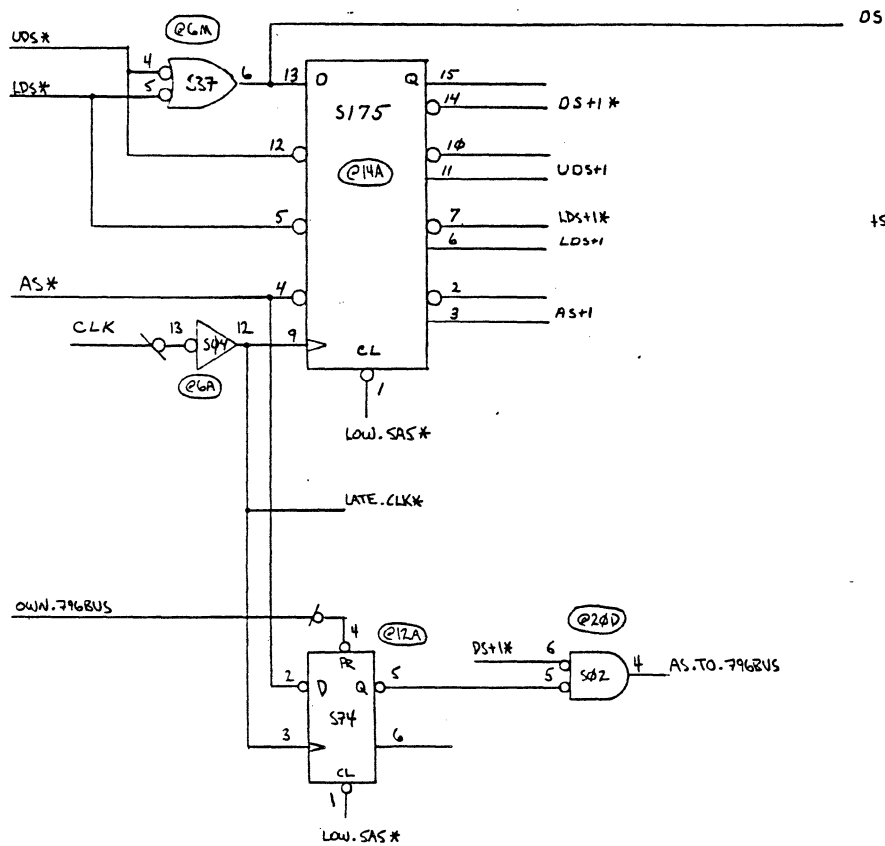


VALID LOGIC SYSTEMS INC.

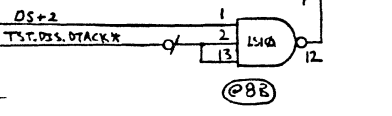
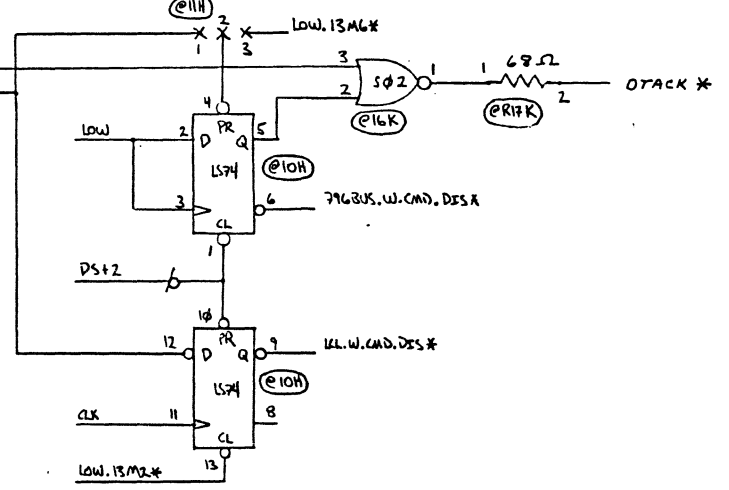
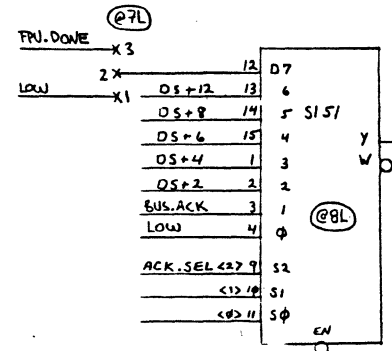
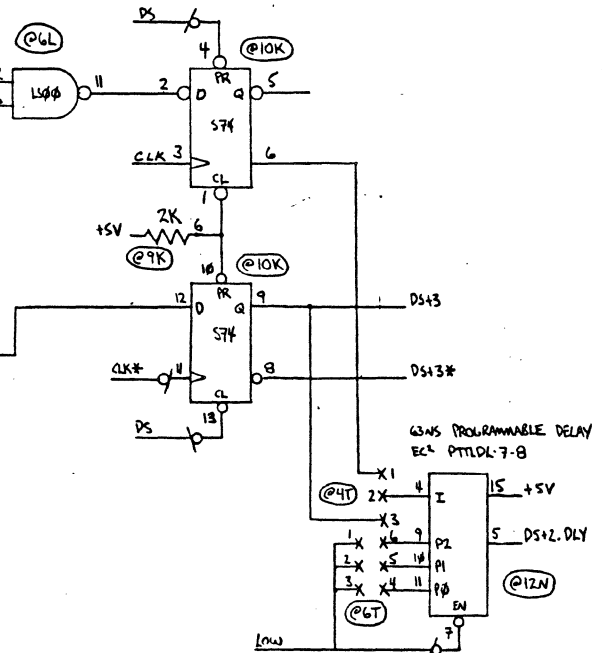
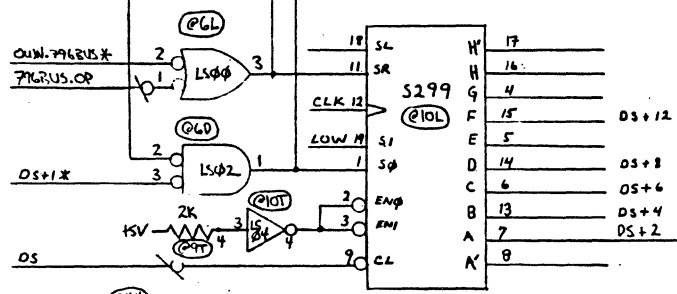
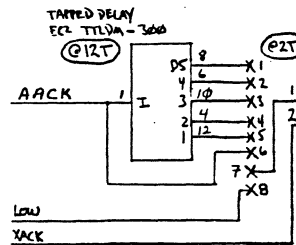
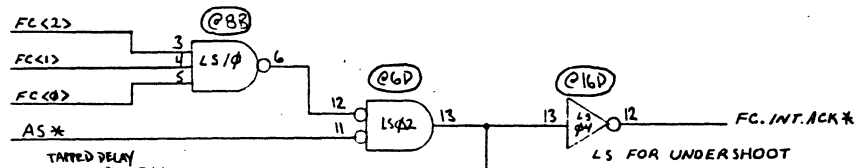
SCALE:	APPROVED BY:	DRAWN BY: WIDDOES
DATE: 11/5/81		REVISED
SCHEMATIC CPU VIRTUAL MEMORY		
CODE IDENT NO REV. A	SHEET 3 OF 26	DRAWING NUMBER 710-00035-2



VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY WIDDOWS
DATE: 11-5-81		REVISED
SCHEMATIC CPU VIRTUAL MEMORY		
CODE IDENT NO REV. A	SHEET 4 OF 26	DRAWING NUMBER 710-000352

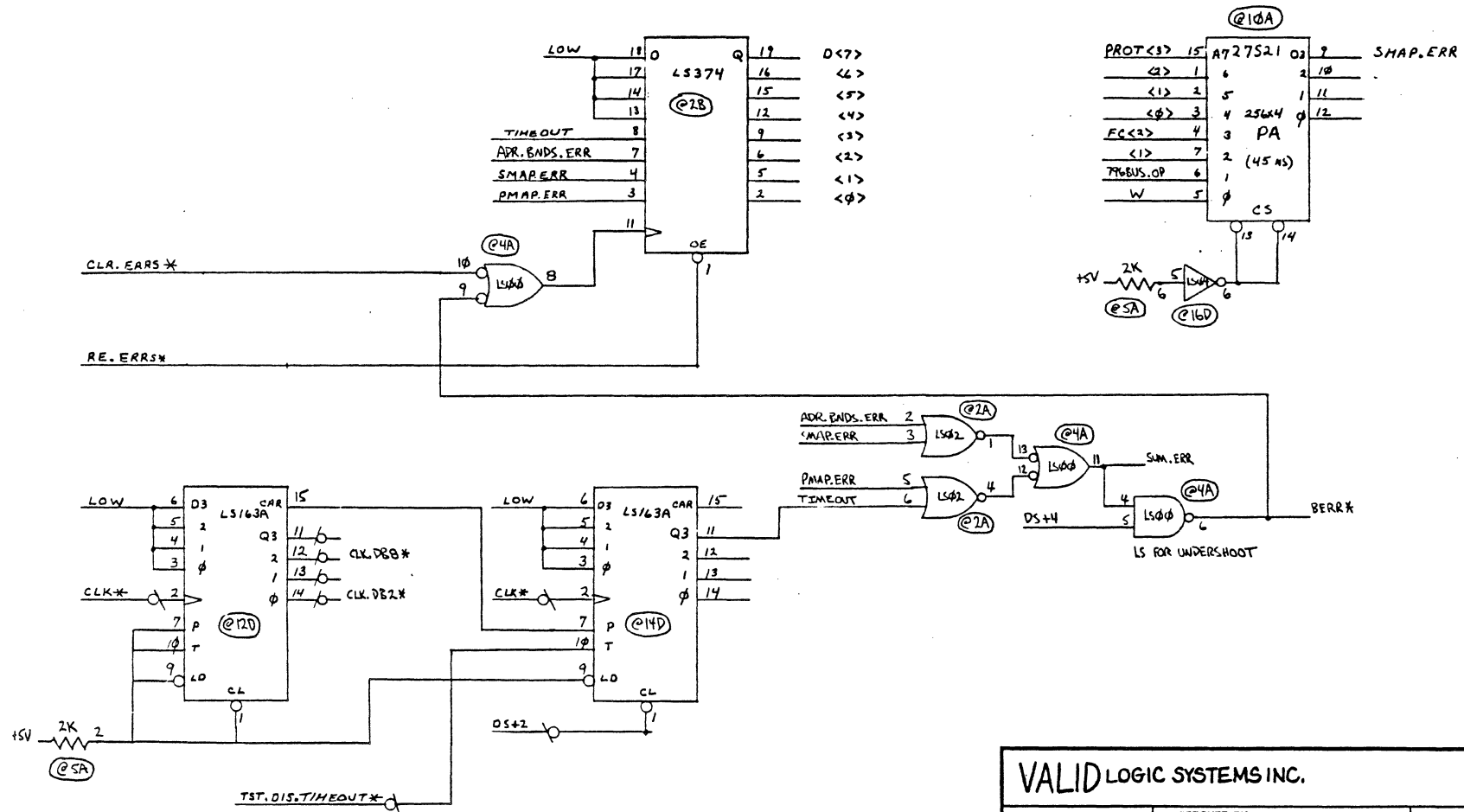


VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY WIDDOES
DATE: 3/8/82		REVISED
SCHEMATIC CPU VIRTUAL MEMORY		
CODE IDENT NO REV. A	SHEET 5 OF 26	DRAWING NUMBER 710-00035-2



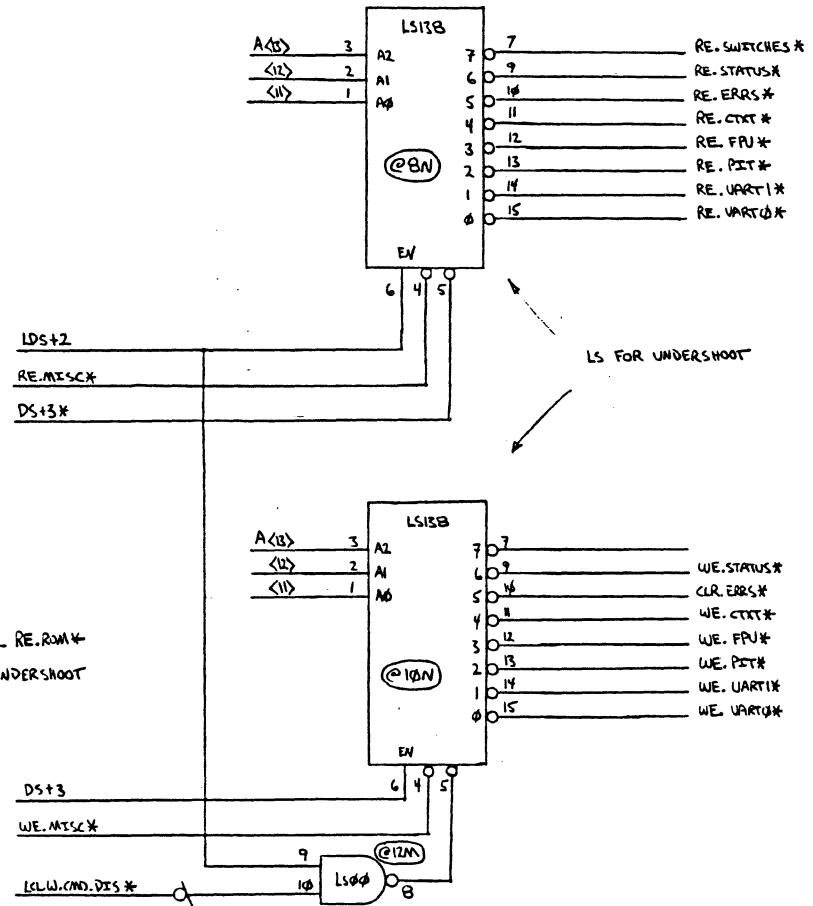
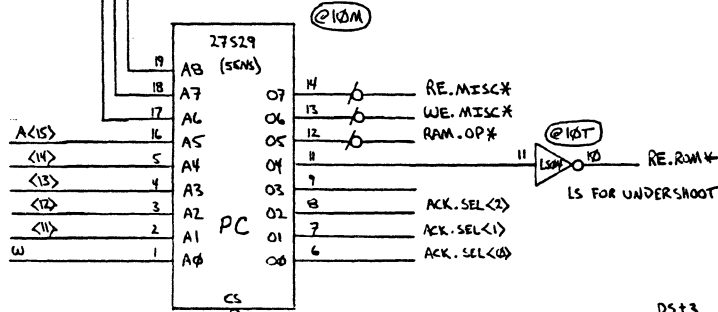
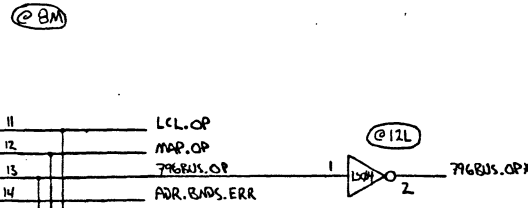
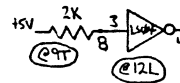
VALID LOGIC SYSTEMS INC.

SCALE:	APPROVED BY:	DRAWN BY: WIDDOES
DATE: 11/5/81		REVISED:
SCHEMATIC CPU VIRTUAL MEMORY		
CODE IDENT NO REV. A	SHEET 6 OF 26	DRAWING NUMBER 710-00035-2

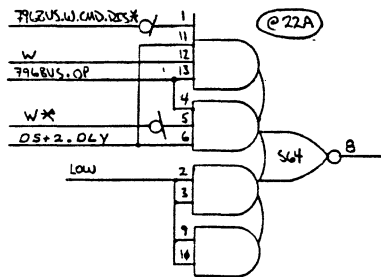
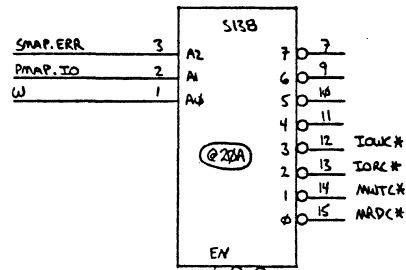


VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY WIDDOES
DATE: 11/5/81		REVISED
SCHEMATIC CPU VIRTUAL MEMORY		
CODE IDENT NO REV A	SHEET 7 OF 26	DRAWING NUMBER 710-00035-2

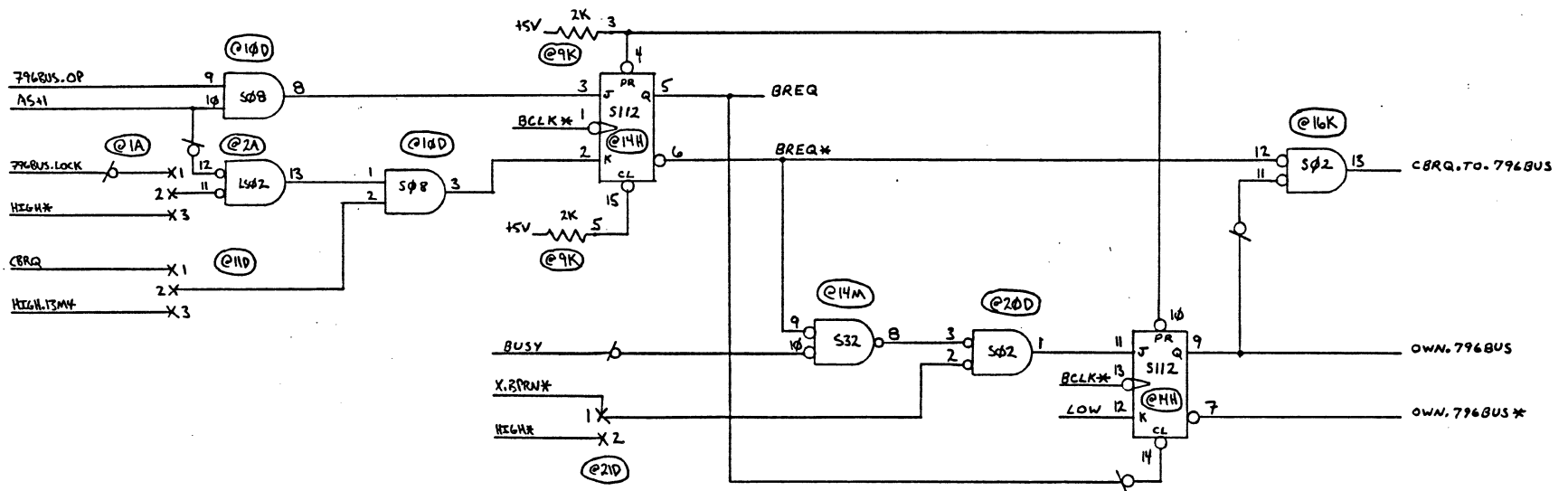
BOOT. STATE #	8	A10
ALLOW. MAP. ACCESS	15	A9 (55NS)
FC<2>	16	A8
A<23>	17	A7
<22>	1	A6
<21>	2	A5
<20>	3	A4
<19>	4	A3
<18>	7	A2
<17>	6	A1
<16>	5	A0



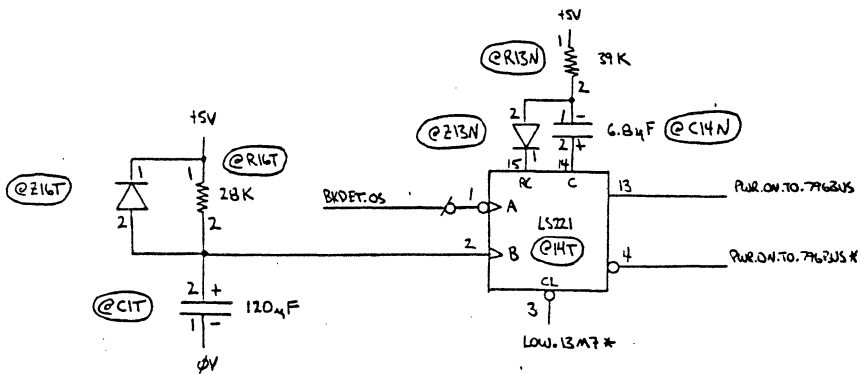
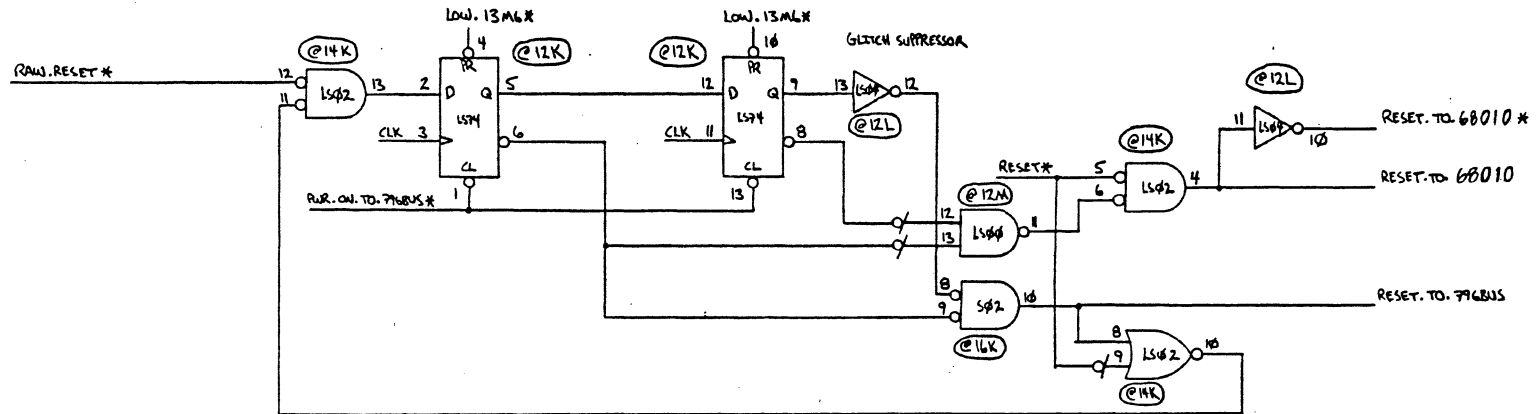
VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY WIDDOES
DATE: 11/5/81		REVISED:
SCHEMATIC CPU VIRTUAL MEMORY		
CODE IDENT NO REV. A	SHEET 8 OF 26	DRAWING NUMBER 710-00035-2



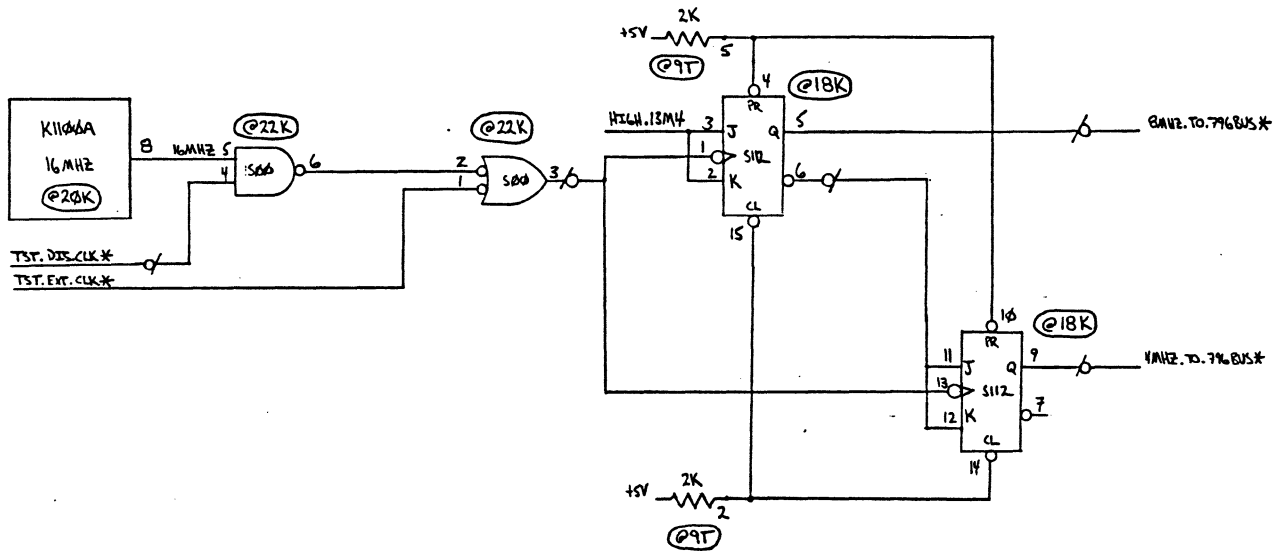
VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY: WIDDOES
DATE: 11/5/81		REVISED:
SCHEMATIC CPU VIRTUAL MEMORY		
CODE IDENT NO REV. A	SHEET 3 OF 26	DRAWING NUMBER 710-00035-2



VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY: WIDDOES
DATE: 3/8/82		REVISED:
<i>SCHEMATIC CPU VIRTUAL MEMORY</i>		
CODE IDENT NO REV. A	SHEET 10 OF 26	DRAWING NUMBER 710-00035-1



VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY: WIDDOES
DATE: 11/5/81		REVISED:
SCHEMATIC CPU VIRTUAL MEMORY		
CODE IDENT NO REV. A	SHEET 11 OF 26	DRAWING NUMBER 710-00035-2



VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY WIDDOES
DATE: 11/15/81		REVISED
SCHEMATIC CPU VIRTUAL MEMORY		
CODE IDENT NO REV. A	SHEET 12 OF 26	DRAWING NUMBER 710-00035-2

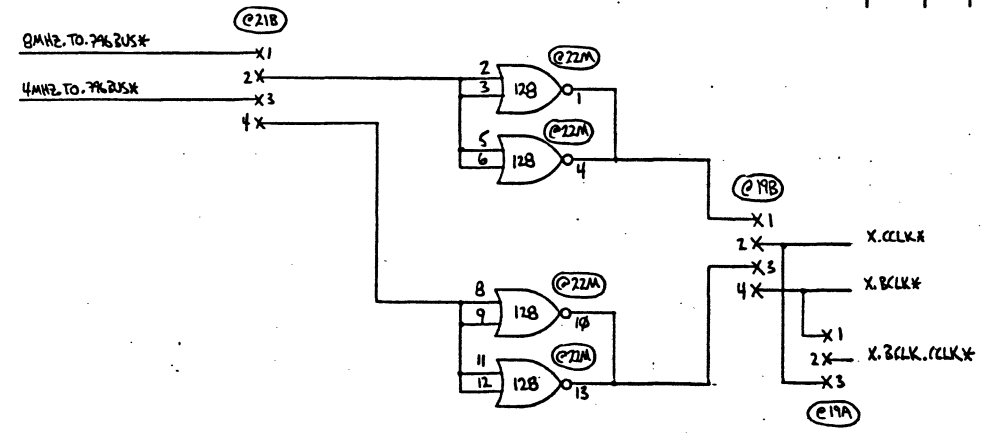
4

3

2

1

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED



D

C

B

A

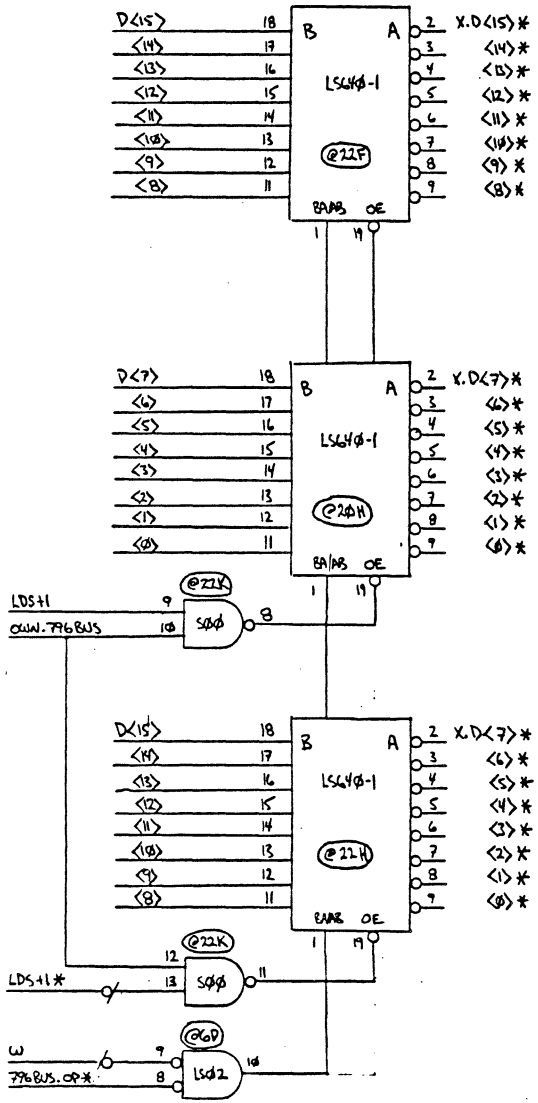
UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE FRACTIONS DECIMALS ANGLES ± .XX ± ± MATERIAL FINISH DO NOT SCALE DRAWING	CONTRACT NO.		
	APPROVALS	DATE	
	DRAWN WIDDOES	3-6-82	SCHMATIC
	CHECKED		CPU VIRTUAL MEMORY
	SIZE B	CODE IDENT NO. REV. A	DRAWING NO. 710-00035-2
	SCALE NONE	SHEET 14 of 26	

4

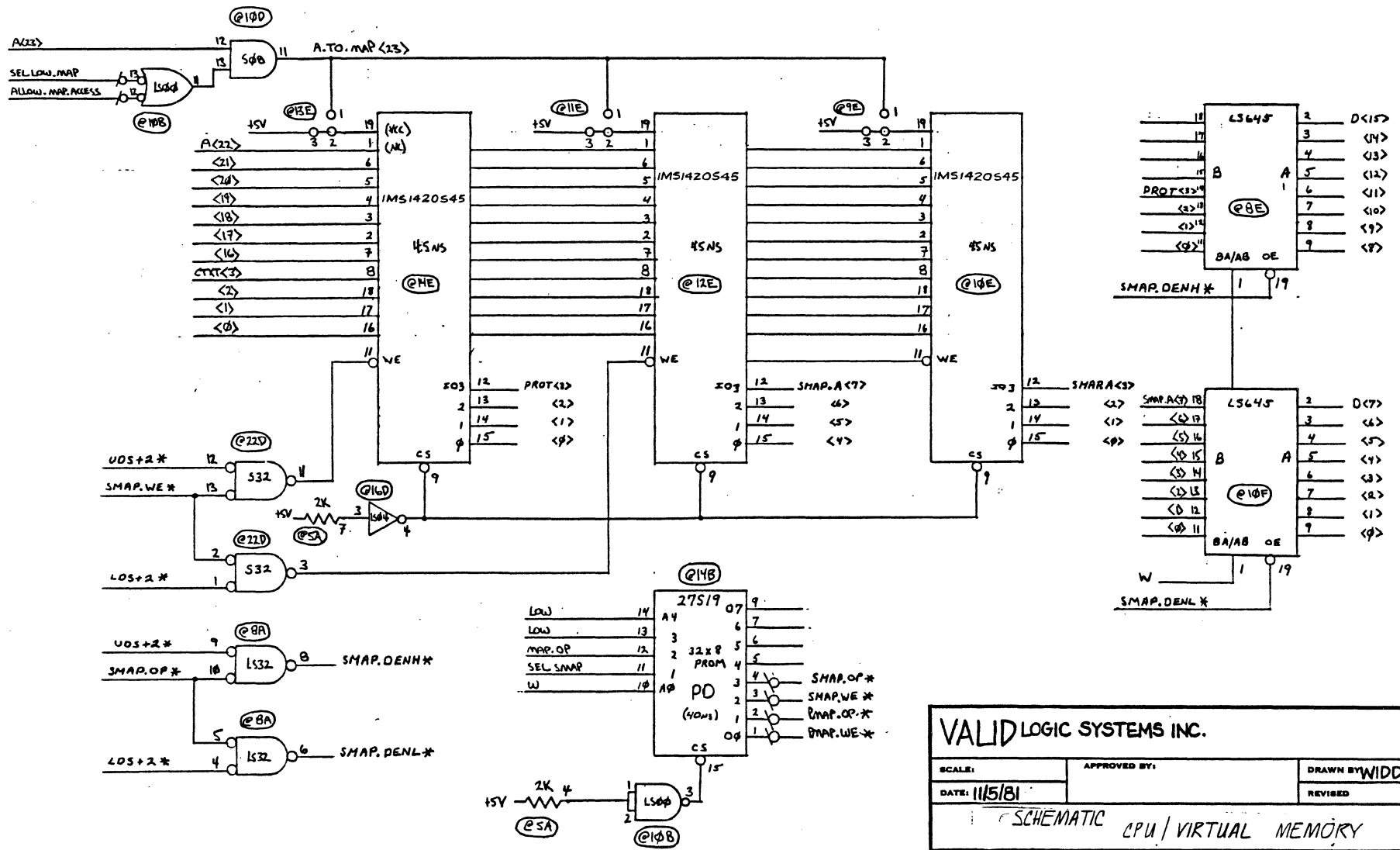
3

2

1

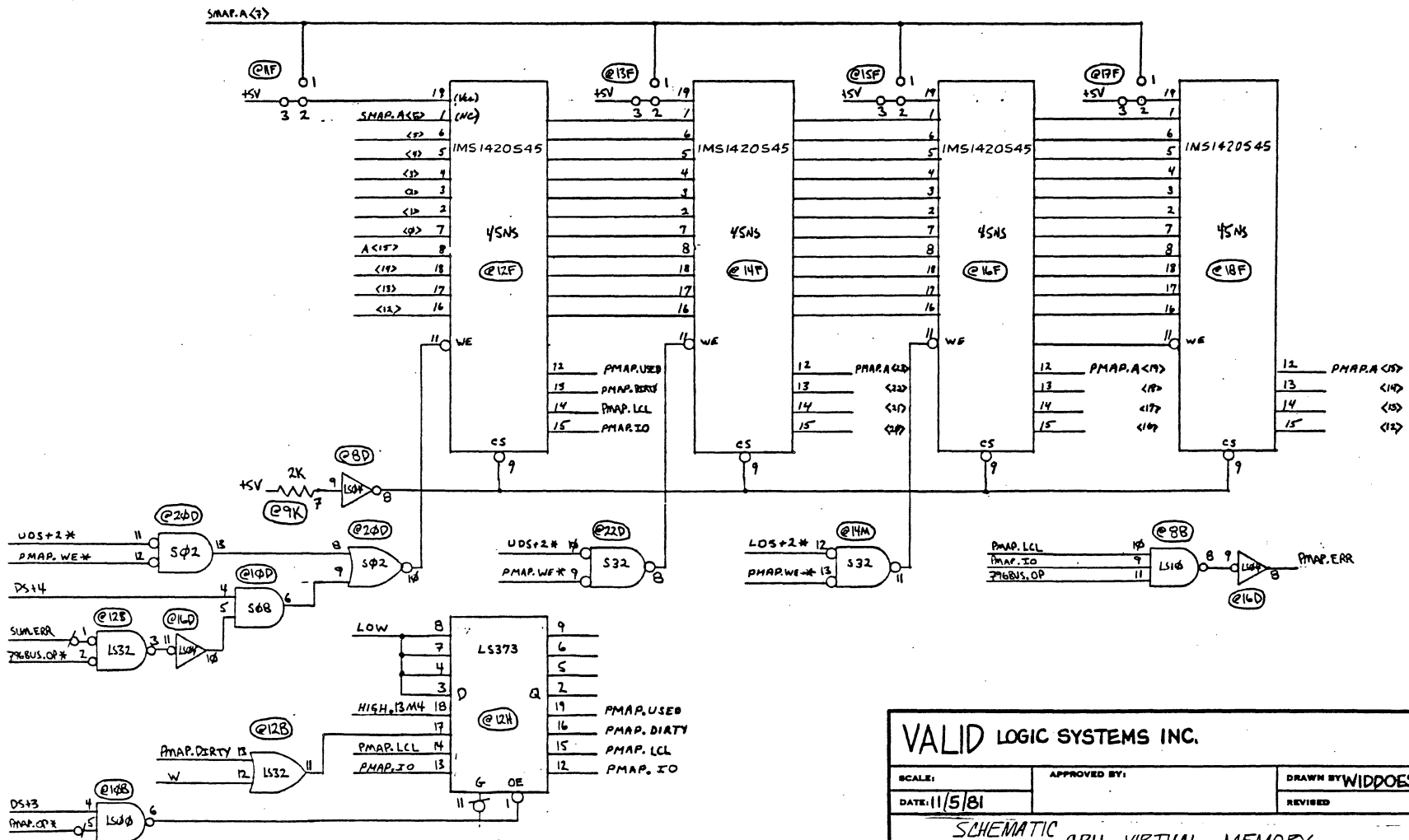


VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY: WIDDOES
DATE: 11/5/81		REVISED:
SCHEMATIC CPU VIRTUAL MEMORY		
CODE IDENT NO REV. A	SHEET 15 OF 26	DRAWING NUMBER 710-00035-2

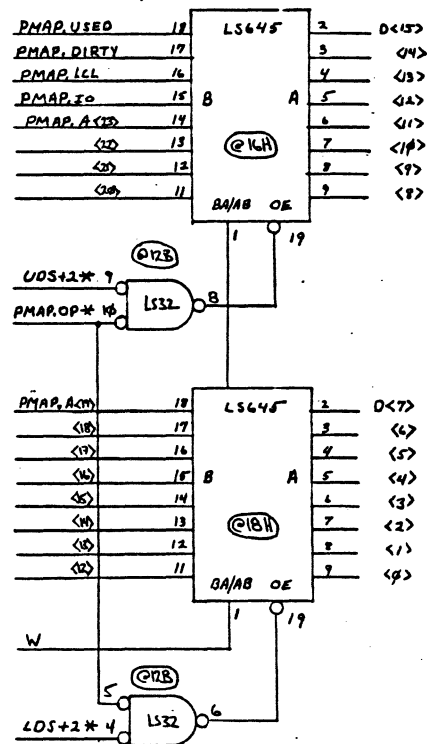


VALID LOGIC SYSTEMS INC.

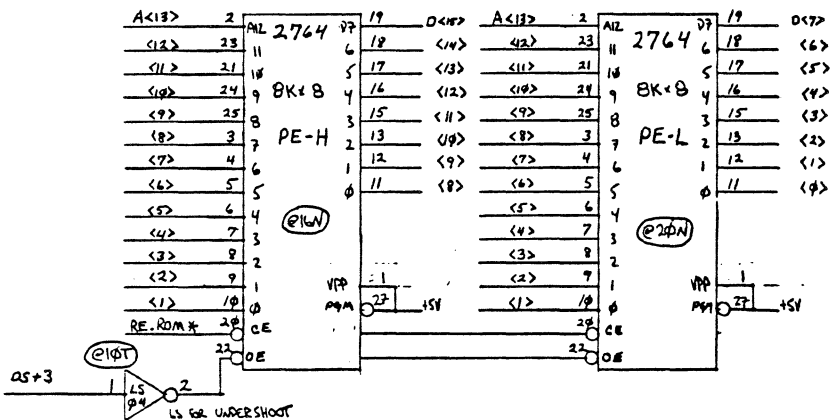
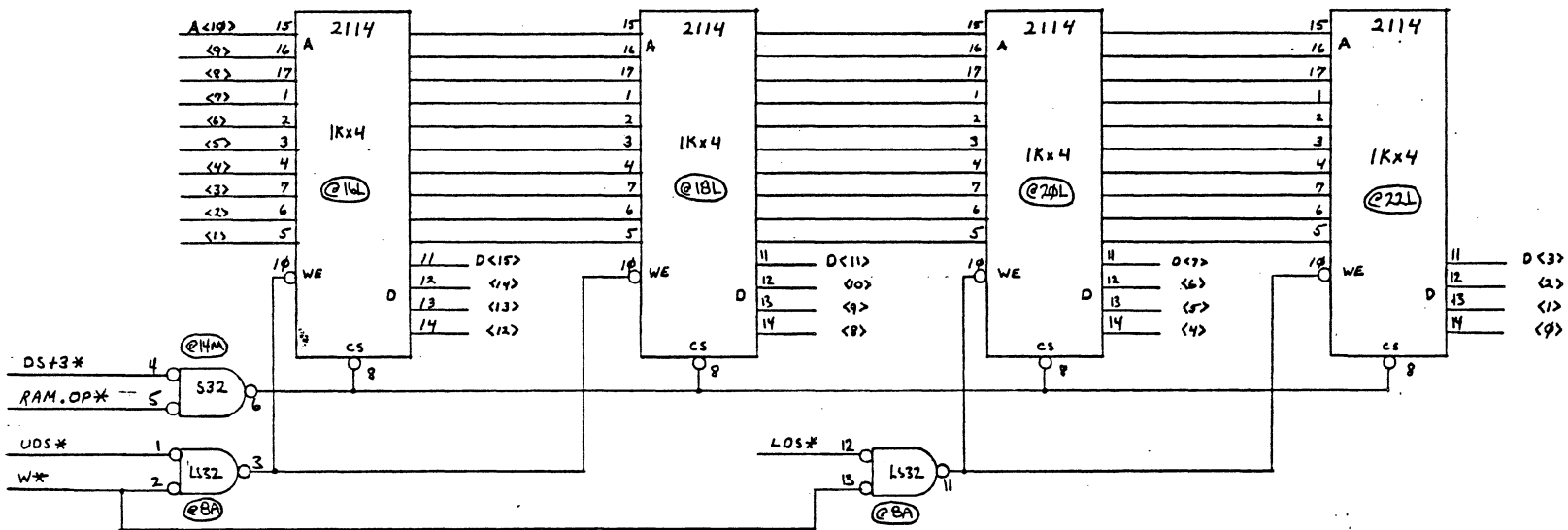
SCALE:	APPROVED BY:	DRAWN BY: WIDDOES
DATE: 11/15/81		REVISED:
SCHEMATIC CPU/VIRTUAL MEMORY		
CODE IDENT NO REV. A	SHEET 16 OF 26	DRAWING NUMBER 710-00035-2



VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY: WIDDOES
DATE: 11/5/81		REVISED:
SCHEMATIC CPU VIRTUAL MEMORY		
CODE IDENT NO REV. A	SHEET 17 OF 26	DRAWING NUMBER 710-00035-2

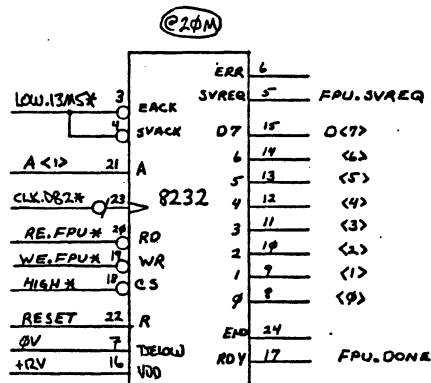
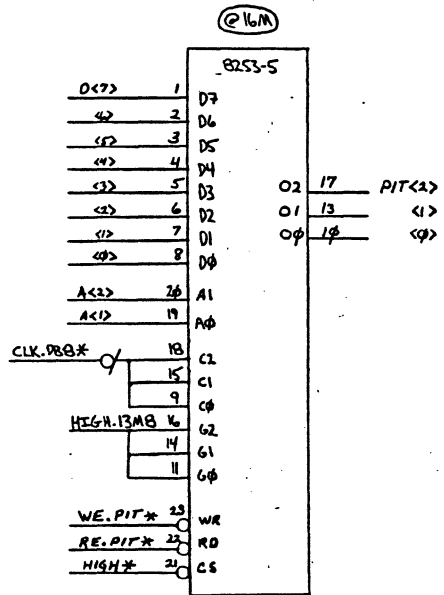


VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY WIDDOES
DATE: 11/5/81		REVISED
<i>SCHEMATIC</i> CPU VIRTUAL MEMORY		
CODE IDENT NO REV. A	SHEET 18 OF 26	DRAWING NUMBER 710-00035-2

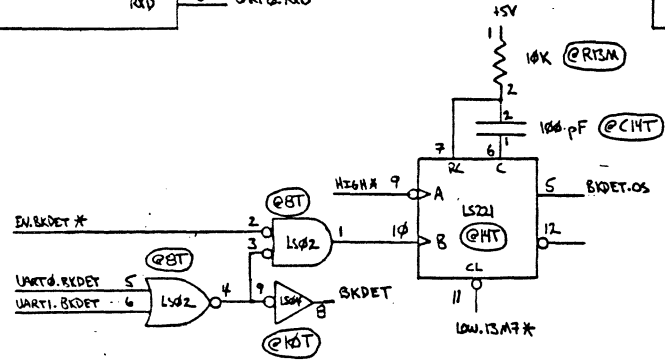
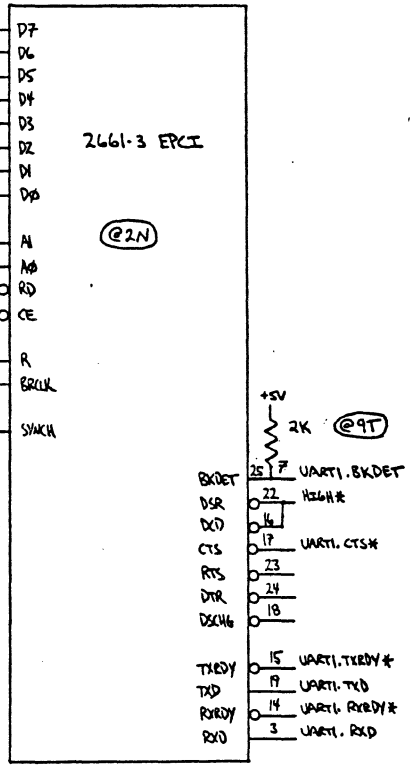
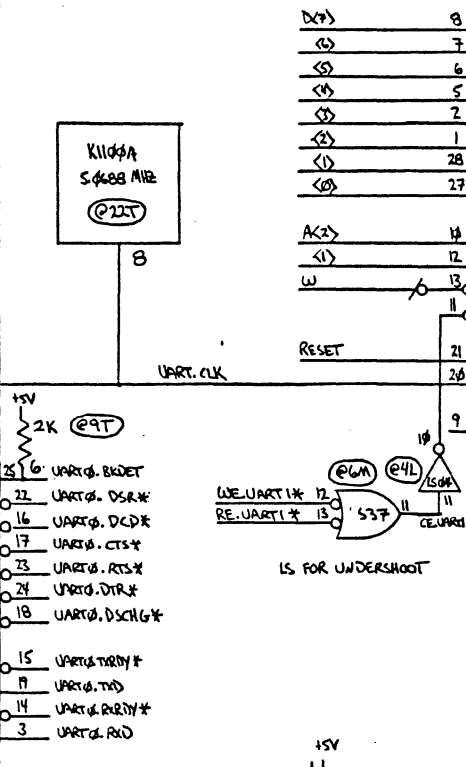
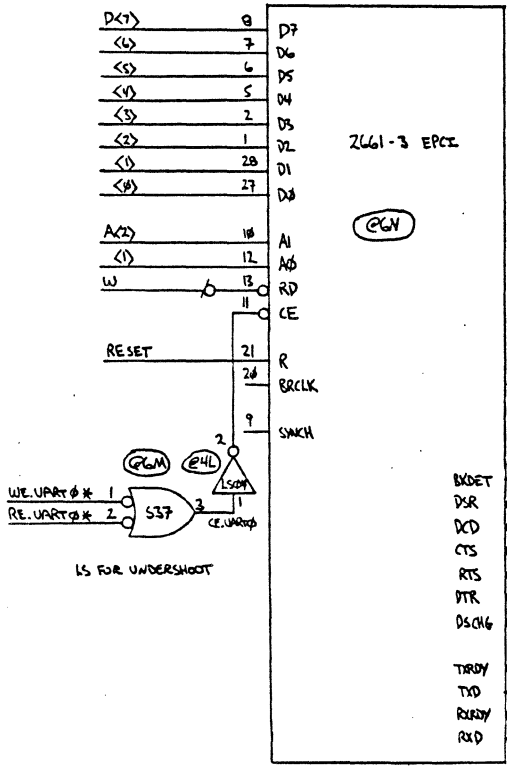


NOTE:
PLUG COMPATIBLE
WITH 2732 EPROM

VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY WIDDOES
DATE: 11/5/81		REVISED
SCHEMATIC CPU VIRTUAL MEMORY		
CODE IDENT NO. REV. A	SHEET 19 OF 26	DRAWING NUMBER 710-00035-Z

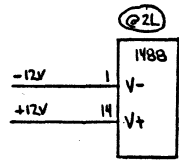
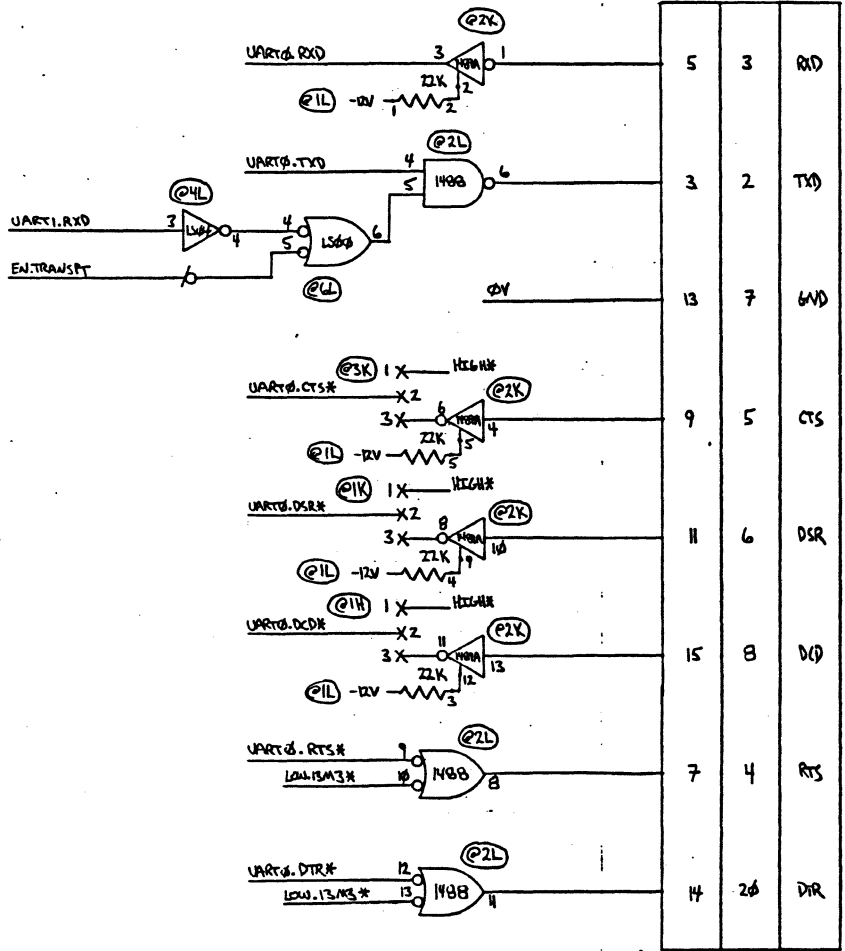


VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY: WIDDOS
DATE: 11/5/81		REVISED:
<i>SCHMATIC</i> CPU VIRTUAL MEMORY		
CODE IDENT NO REV. A	SHEET 20 OF 26	DRAWING NUMBER 710-00035-2

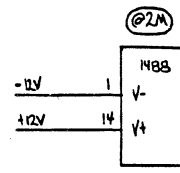
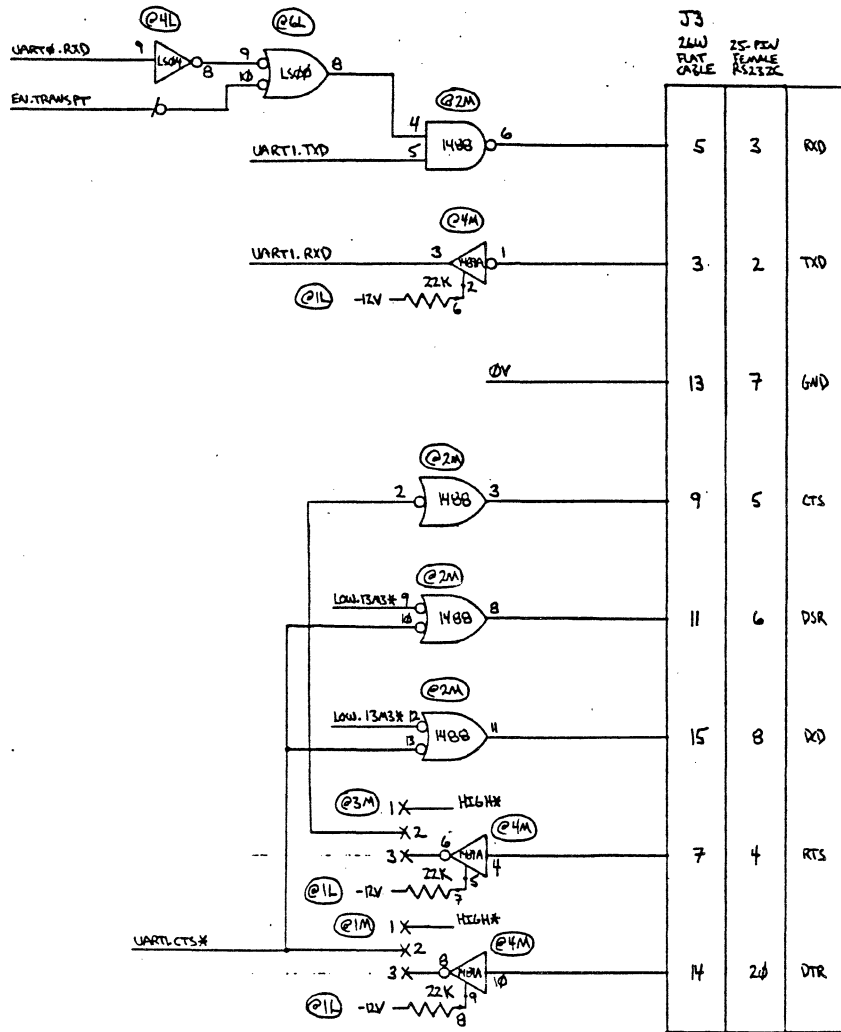


VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY WIDDOWS
DATE: 11/5/81		REVISED
SCHEMATIC CPU VIRTUAL MEMORY		
CODE IDENT NO REV. A	SHEET 21 OF 26	DRAWING NUMBER 710-00035-2

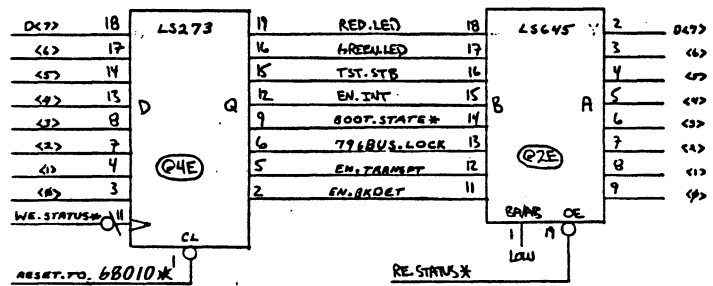
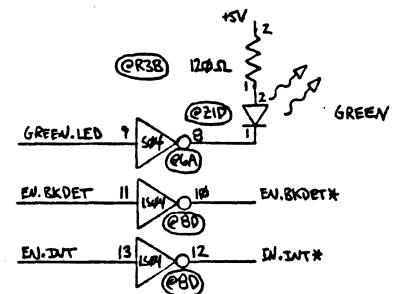
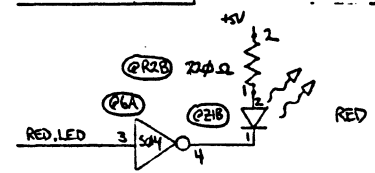
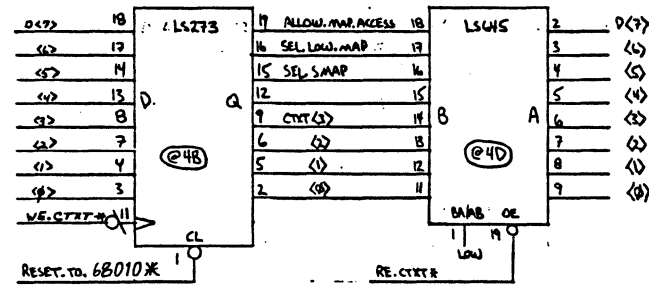
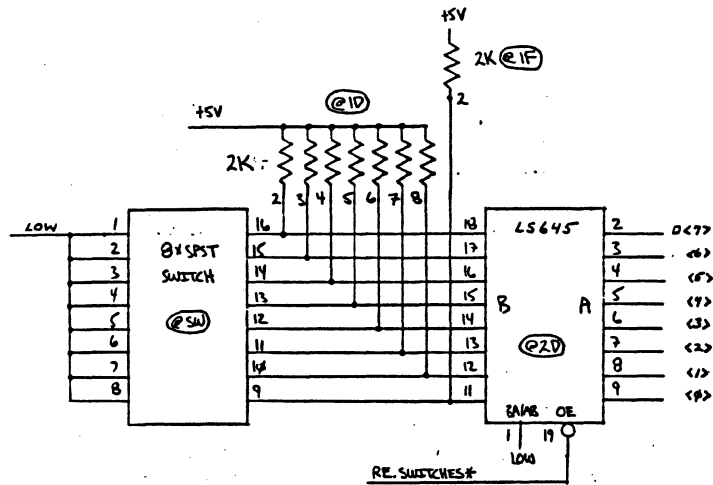
J2
26W
FLAT
CABLE
25-PIN
MALE
DS232C



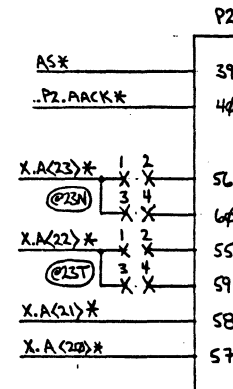
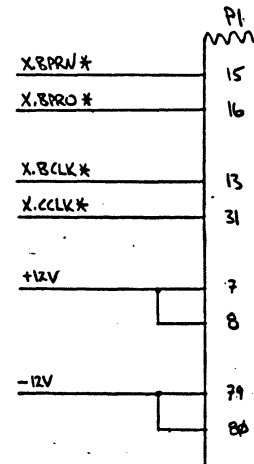
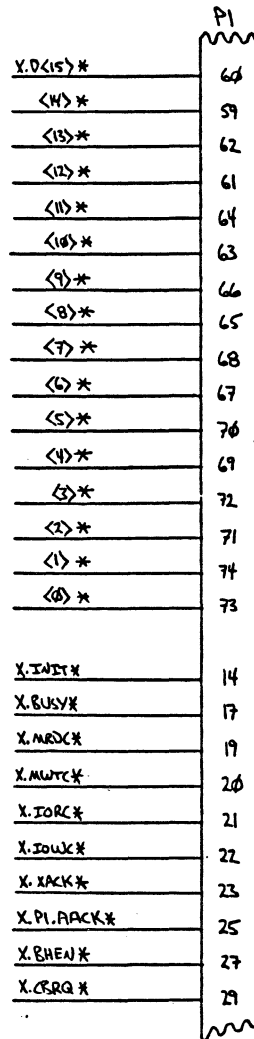
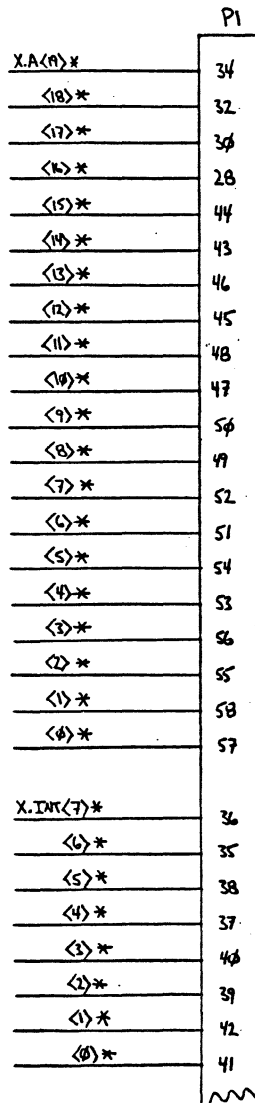
VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY WIDDOES
DATE: 11/5/81		REVISED
SCHEMATIC CPU VIRTUAL MEMORY		
CODE IDENT NO REV. A	SHEET 22 OF 26	DRAWING NUMBER 1710-00035-2



VALID LOGIC SYSTEMS, INC.		
SCALE:	APPROVED BY:	DRAWN BY: WIDDOES
DATE: 11/5/81		REVISED:
SCHEMATIC CPU VIRTUAL MEMORY		
CODE IDENT NO REV. A	SHEET 23 OF 26	DRAWING NUMBER 710-00035-2



VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY WIDDOES
DATE: 11/5/81		REVISED
SCHEMATIC CPU VIRTUAL MEMORY		
CODE IDENT NO REV. A	SHEET 24 OF 26	DRAWING NUMBER 710-00035-2



VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY WIDDOES
DATE: 11/5/81		REVISED
SCHEMATIC CPU VIRTUAL MEMORY		
CODE IDENT NO REV. A	SHEET 26 OF 26	DRAWING NUMBER 710-00035-2

512KBYTE ECM USER'S MANUAL

VED-031682-1 Revision 6-8-82 (LCW)

31 Jul 84

Copyright 1984

Valid Logic Systems Incorporated

This document contains confidential proprietary information which is not to be disclosed to unauthorized persons without the written consent of an officer of Valid Logic Systems Incorporated.

The copyright notice appearing above is included to provide statutory protection in the event of unauthorized or unintentional public disclosure.

TABLE OF CONTENTS

Introduction	1
Multibus Interface	2
High-Speed Mode	4
Addressing	4
Status Register	5
Error Address Register	6
Diagnostic Register	7
Timing	8
Operating System Considerations	9
Appendix A - Strapping Options	
Appendix B - Assembly Drawing	
Appendix C - Specifications	

1.0 INTRODUCTION

The Valid Logic Systems 512KBy Error-Correcting Memory (ECM) features very high performance, high reliability, and high functionality. The board contains 512KBy of read/write memory interfaced to a 16MBy Multibus and accessible as words or bytes. Each access is checked and corrected using a single error correct - double error detect (SECDED) code that corrects any single-bit error and detects any double-bit error (as well as some other multi-bit errors). Advanced pipelining logic allows read accesses from the 8MHz Valid M68000 CPU (CPU) to proceed with only one wait cycle, even with two levels of mapping on the CPU board and full error correction on the ECM board.

Features

- o 512KByte capacity
- o One-wait-cycle corrected read using Valid 8MHz M68000 CPU
- o Full support of the 16MBy IEEE-796 bus (Intel Multibus)
- o Transparent single-error correction and double-error detection
- o Software-enabled single- and double-error interrupts
- o Strappable Multibus interrupt levels
- o Multibus-accessible error-address and syndrome-bit registers
- o Multibus-accessible read/write status and mode registers
- o Diagnostic LEDs display status
- o Supports special diagnostic writes of the error-correction bits
- o Switch-selectable bank address
- o Automatic refresh

The ECM is designed to be used with full-functionality operating systems such as UNIX. Separate error interrupts are available for single- and double-bit errors, and each is individually maskable on the board under software control. Each interrupt can be strapped to any Multibus interrupt level. The least-recent error address and the error syndrome bits are available as Multibus I/O locations. An overflow-error bit in the status register indicates that a second error occurred before the first was cleared.

To facilitate testing, the ECC bits can be written by Multibus devices (using special diagnostic-mode writes). To facilitate maintenance, two diagnostic LEDs display the error status, and two additional LEDs can be accessed in I/O space on the Multibus and are available for use by software diagnostics. For ease in trouble-shooting, the board can be strapped to inhibit error correction and to inhibit refresh.

This manual applies only to Valid ECM PC assembly 710-00005 Rev. B.

2.0 MULTIBUS INTERFACE

The ECM is a slave device on a standard 16MBy Multibus. High-current bus drivers are used for all output signals. Bus receivers are used for all input signals.

Internally, the ECM is organized as 16-bit words with an additional six error-correction-code (ECC) bits per word. However, the board supports both byte and word operations. In the case of a byte write, the board first reads the old word value, merges the new byte, then affixes the final check bits to the result before writing into the RAM array.

The board receives 24 address bits. The top two address bits (A<23..22>) are strappable to be received in either the standard positions (P2-55 and P2-56) or the common alternate positions (P2-59 and P2-60).

The ECM drives the single- and double-bit error interrupts to any of the seven standard Multibus interrupt lines; the assignment is selectable by means of two straps.

The board directly connects BPRN* to BPRO* to facilitate serial bus-priority systems. It also processes INH1* and INH2* according to Multibus specifications. INIT* resets the board to a fixed initial state (without inhibiting automatic refresh). AACK* is strappable to be driven either on P1-25 or P2-40.

The ECM is capable of synchronous operation; it is strappable to run off of the (8MHz) CCLK* or BCLK*. Alternatively, it can be strapped to use its own internal 8MHz clock. "High-speed" mode requires that the ECM be strapped for synchronous operation.

Tables 2-1 and 2-2 show all Multibus signals used by the ECM.

Table 2-1. Valid ECM Multibus Interrupt and Control Signals

Signal	Pin	Signal	Pin
INT<7>*	P1-36	INIT*	P1-14
INT<6>*	P1-35	MRDC*	P1-19
INT<5>*	P1-38	MWTC*	P1-20
INT<4>*	P1-37	IORC*	P1-21
INT<3>*	P1-40	IOWC*	P1-22
INT<2>*	P1-39	XACK*	P1-23
INT<1>*	P1-42	AACK*	P1-25 or P2-40
INT<0>*	P1-41	BHEN*	P1-27
		BPRN*	P1-15
INH1*	P1-24	BPRO*	P1-16
INH2*	P1-26	BCLK*	P1-13
		CCLK*	P1-31
AS*	P2-39		

Table 2-2. Valid ECM Multibus Address and Data Signals

Signal	Pin	Signal	Pin
A<19>*	P1-34	A<23>*	P2-56 or P2-60
A<18>*	P1-32	A<22>*	P2-55 or P2-59
A<17>*	P1-30	A<21>*	P2-58
A<16>*	P1-28	A<20>*	P2-57
A<15>*	P1-44		
A<14>*	P1-43	D<15>*	P1-60
A<13>*	P1-46	D<14>*	P1-59
A<12>*	P1-45	D<13>*	P1-62
A<11>*	P1-48	D<12>*	P1-61
A<10>*	P1-47	D<11>*	P1-64
A<9>*	P1-50	D<10>*	P1-63
A<8>*	P1-49	D<9>*	P1-66
A<7>*	P1-52	D<8>*	P1-65
A<6>*	P1-51	D<7>*	P1-68
A<5>*	P1-54	D<6>*	P1-67
A<4>*	P1-53	D<5>*	P1-70
A<3>*	P1-56	D<4>*	P1-69
A<2>*	P1-55	D<3>*	P1-72
A<1>*	P1-58	D<2>*	P1-71
A<0>*	P1-57	D<1>*	P1-74
		D<0>*	P1-73

3.0 HIGH-SPEED MODE

The ECM can be strapped to operate in high-speed mode, and is then is capable of performing corrected reads of bytes or words while introducing only one wait cycle for the Valid CPU, and corrected writes of bytes or words with only two wait cycles. This high-performance operation is achieved by using two non-standard Multibus signals, advanced acknowledge (AACK*) and address strobe (AS*). Because of dependence on CPU timing, this mode can be used only by the Valid CPU.

The Valid CPU contains advanced two-level (segmentation and paging) memory-mapping hardware. Since it uses a 4KBy page size, the low-order 12 address bits are available earlier than the high-order 12 bits. The CPU asserts AS* to signal that the low-order addresses are ready and (in high-speed mode) the ECM asserts RAS* immediately for one of four 22-bit rows selected by address bits A<10..9>. RAS* is asserted even before the Multibus command strobe occurs. If the high-order address bits fail to select the board at command-strobe time, the ECM inhibits further operation, no CAS occurs, and no state change occurs on the board.

The high-speed mode also uses AACK*. Whenever an access begins with AS*, the ECM asserts AACK* during the cycle preceding data-ready time; this cycle then is used by the CPU in preparing to latch the data. AACK* is strappable to be driven on either P1-25 (LOCK* on some Multibus systems, and AACK* on some others) or on P2-40.

XACK* is normally delivered according to Multibus specifications, independent of AACK*. However, XACK* can be strapped not to occur on accesses that begin with AS* in order to guarantee that the CPU is responding to AACK*.

Normal Multibus devices do not assert AS*. Accesses from these devices do not use high-speed mode and do not begin before the Multibus command strobe occurs.

High-speed mode can be inhibited simply by strapping the ECM so that AS* appears to be always deasserted.

4.0 ADDRESSING

The ECM must be aligned on a 512KBy boundary in Multibus memory space. A bank of five switches (location 28A, switches 4..8) selects the board base address: Switch 8 is the least-significant bit of the board address; "ON" corresponds to a "0," and "OFF" corresponds to a "1".

The ECM also responds to full 16-bit I/O addresses on the Multibus. The base I/O address of the board (i.e., the upper eight bits of I/O function addresses on the board) is simply the value of the eight switches at location 28A: Switch 8 is the least-significant bit; "ON" corresponds to a "0," and "OFF" corresponds to a "1." Note that the low-order five bits of the base I/O address are determined by the same switches as the base memory address.

Within a board, the low-order three bits of the I/O address select the I/O Function as follows:

Table 4-1. I/O Function Offsets

A<2..0>	Direction	I/O Function
0	read/write	STATUS<7..0>
2	read	ERRADR<15..0>
4	read	ERRADR<23..16>
6	write	DIAGREG<15..0>

Only the low-order three bits of the offset (A<2..0>) are significant in selecting the I/O function. Thus each function also is selected by 31 alias addresses within the same 256-byte block. All I/O functions are 16 bits wide; byte devices within the board simply ignore the most-significant byte or leave it undefined, depending on the operation direction. Undefined I/O operations (e.g., a read of the diagnostic register) change no ECM state and also do not time-out.

4.1 Status Register

The Status Register (STATUS) is read or written as the least-significant byte of I/O Function 0. Every bit of STATUS can be read or written from the Multibus. In addition, certain bits are set automatically by the ECM, and the entire register is cleared (set to zero) at RESET time. The assignment of bits is as follows:

Table 4-2. STATUS Format

STATUS Bit	Name	Meaning
7	OFE	overflow error
6	MBE	multi-bit error
5	ERR	some error
4	ENMBEI	enable MBE to cause interrupt
3	ENERRI	enable ERR to cause interrupt
2	DIAG	enable diagnostic writes
1	REDLED	software-controlled red LED
0	GREENLED	software-controlled green LED

OFE is set by the ECM if ERR already is true when a new single- or multi-bit error occurs. MBE is set only when a multi-bit error occurs, whereas ERR is set when either a single- or multi-bit error occurs. OFE, MBE, and ERR are

"sticky" (i.e., they are not reset by subsequent errors). Note that single- or multi-bit errors can occur during write accesses, that are implemented as read-modify-write cycles. ERR and MBE drive the LEDs at the top of the ECM board at locations Z1EA and Z1EB, respectively. Thus, two red lights signify a multi-bit error, while one light indicates a single-bit error.

ENMBEI enables MBE to drive the Multibus interrupt line to which it is strapped. Similarly, ENERRI enables ERR to drive the Multibus interrupt line to which it is strapped. Since STATUS is fully writeable, MBE and ERR can be reset by the error-handling interrupt routine.

DIAG causes all writes to the RAM array to be done in "diagnostic" mode. In this mode, the low-order six bits of DIAGREG (loaded by an I/O function) are substituted for the normally generated check bits during writes. Reads proceed normally, possibly detecting errors. Writes in diagnostic mode also may detect errors during the read portion of the read-modify-write cycle. Diagnostic mode is important for verifying that the ECM is correcting errors properly.

REDLED and GREENLED drive LEDs on the top of the ECM board at locations Z1BA and Z1BB, respectively. These LEDs can be set or reset under software control. Valid diagnostics use these LEDs to display the test results.

4.2 Error Address Register

As long as the ERR bit in STATUS is false, the Error Address Register (ERRADR) latches the address and syndrome bits associated with each read or write to the ECM. When ERR becomes true, ERRADR simply holds the last address and syndrome bits (i.e., those associated with the error). ERRADR thus latches the least-recent error. It is not explicitly writeable via the Multibus.

ERRADR is a three-byte register. The low-order word of ERRADR is accessible as I/O Function 2; it contains bits A<16..11>:A<18..17>:A<8..1> of the access address. The high-order byte of ERRADR is accessible as I/O Function 4. It contains bits A<10..9> of the access address in its low-order bits, and syndrome bits S8:S4:S2:S1:S0: SX in its high-order bits. Since the board is selected by address bits A<23..19>, these bits of the access address are not recorded. Similarly, since every access to the ECM RAM array is word-wide, A<0> is not recorded. The syndrome bits are generated by the AM2960; these bits identify the location of any single-bit error in a 16-bit word. For more information on the AM2920, refer to the vendor's information.

Table 4-3. ERRADR Format

ERRADR Bits	Contents
<23..16>	S8:S4:S2:S1:S0: SX : A<10..9>
<15..0>	A<16..11> : A<18..17> : A<8..1>

Bits A<10..9> of the access address define the "row" containing the error. The individual bits of each row are located on the PC board as shown in the following table:

Table 4-4. RAM Chip Location Designators

A10	A9	D0	D2	D4	D6	D8	D10	D12	D14	CX	C1	C4
0	0	2F	4F	6F	8F	10F	12F	14F	16F	18F	20F	22F
0	1	2K	4K	6K	8K	10K	12K	14K	16K	18K	20K	22K
1	0	2M	4M	6M	8M	10M	12M	14M	16M	18M	20M	22M
1	1	2T	4T	6T	8T	10T	12T	14T	16T	18T	20T	22T

A10	A9	D1	D3	D5	D7	D9	D11	D13	D15	C0	C2	C8
0	0	2H	4H	6H	8H	10H	12H	14H	16H	18H	20H	22H
0	1	2L	4L	6L	8L	10L	12L	14L	16L	18L	20L	22L
1	0	2N	4N	6N	8N	10N	12N	14N	16N	18N	20N	22N
1	1	2U	4U	6U	8U	10U	12U	14U	16U	18U	20U	22U

4.3 Diagnostic Register

The Valid ECM features the capability to fully diagnose its error-correction logic. When the DIAG bit in STATUS is true, the contents of the Diagnostic Register (DIAGREG<5..0>) are substituted for the normally generated check bits during every write to the RAM array. In this way, any arbitrary 22-bit pattern can be forced into any location within the RAM array.

DIAGREG is a write-only register. Its contents at RESET time are undefined. Only the low-order six bits of DIAGREG are defined; these bits correspond to the 16-bit, format-check bits as follows (refer to the AM2960 documentation for additional information):

Table 4-5. DIAGREG Format

DIAGREG Bit	Contents
5	CB8
4	CB4
3	CB2
2	CB1
1	CB0
0	CBX

5.0 TIMING

When a memory operation is initiated, the ECM performs the following basic sequence:

1. Issues RAS* to the selected row (A<10..9>) of the RAM array.
2. Issues CAS* to the RAM array.
3. Latches the 16-bit write data into a holding register.
4. Latches the 22-bit RAM array read data into the AM2960 input latch.
5. Checks and corrects the RAM array read data and sets OFE, MBE, or ERR as appropriate.
6. In case of a read, drives the corrected data to the Multibus.
7. In case of a write, merges the corrected data with the byte to be written or overlays it with the word to be written, generates new ECC bits, and writes the result into the RAM array.

An I/O operation proceeds with the same timing and order, but no CAS* is issued, no error checking, correcting, or data merging is performed, and reading or writing of I/O registers is substituted for reading or writing of the RAM array.

If INH1* or INH2* becomes asserted during an ECM operation, then XACK*, AACK*, and the data-output buffers are all immediately disabled (driven to high-impedance), but otherwise the cycle completes normally.

Refresh occurs automatically with one row being refreshed every 12 us. Refresh takes priority over Multibus accesses, although the ECM always waits until a previous operation has completed before initiating a new operation.

The ECM is designed to operate normally in high-speed mode with a Valid 8MHz M68000 CPU. In high-speed mode, the ECM must be strapped to run off of the same clock (either CCLK* or BCLK*) as the CPU. In this case, we call the synchronous internal CPU/ECM clock "CLK" and "CLK*", where CLK* has the same phase as CCLK* or BCLK*. We note that CLK may have as much as 15 ns skew between the CPU and ECM. If high-speed mode is not used, the ECM may be strapped to run off its own 8MHz internal clock. In this mode we still call the internal ECM clock "CLK" and "CLK*".

In high-speed mode, the CPU asserts AS* on the falling edge of CLK (the beginning of M68000 S3 for a read, S5 for a write), and RAS* is issued to the RAM array on the next rising edge, at which time the CPU has been driving A<11..1> to the bus for at least 70 ns, not including CLK skew or address settling time. If an I/O or memory command (with proper board selection) arrives at all, then it must be stable on the bus before 5 ns after the following falling edge of CLK (the beginning of M68000 S5 for a read, S7 for a write), and it is sampled shortly thereafter. If the sampled board-selected command is a memory operation, then CAS* is asserted, otherwise CAS* is inhibited.

The board-selected memory command may not arrive at all, for example, in case the access is an I/O operation, in case another ECM is being accessed, or in case of a segment or page fault on the CPU board. An I/O or memory command is executed by the ECM if, and only if, the board-selected command arrives promptly as described above. Furthermore, CAS* is issued only for memory commands. If no board-selected command arrives, then the ECM completes its normal cycle without changing any accessible state, in particular, without changing STATUS or the contents of the RAM array.

In case of an AS* access with a prompt board-selected command following, AACK* is asserted approximately 30 ns after the start of M68000 S5 (for a read) or S7 (for a write). This interval allows the data to be latched by the M68000 at the beginning of S9 (for a read) or S11 (for a write). Thus one wait cycle is incurred on reads, and two on writes. This timing applies to both memory and I/O operations. Note that the ECM can be strapped to inhibit XACK* during those operations for which AACK* is asserted in order to ensure that the CPU is responding to AACK*.

Bus masters other than the Valid CPU do not assert AS*. For these devices (and also in the case that AS* is strapped to the de-asserted state on the ECM), the Multibus command is synchronized on the falling edge of CLK. After synchronization, operation proceeds as in high-speed mode except that AACK* is inhibited. For this type of operation, XACK* is asserted four cycles after the first falling edge of CLK which catches the asserted command (i.e., between 4+ and 5 cycles after the command edge itself). The shortest possible interval occurs when the command is asserted on the bus approximately 15 ns before the falling edge of CLK in which case XACK* is asserted four cycles after that falling edge.

6.0 OPERATING SYSTEM CONSIDERATIONS

Logging single-bit errors during normal operations can identify marginal RAM chips that can be replaced during subsequent periodic maintenance. The Valid ECM includes the features necessary for error logging in a full-functionality operating system. With MBE and ERR strapped to drive Multibus interrupt level 0 (and Multibus interrupt level 0 strapped to non-maskable level 7 on the Valid CPU), any error causes a non-maskable interrupt that is serviced after completion of the instruction causing the error.

The memory-error-trap interrupt handler identifies the board responsible for the interrupt as well as the interrupt type (single- or multi-bit error) by reading the each ECM board's STATUS register.

After determining the board responsible for the interrupt, the handler reads the ERRADR register to determine which chip caused the error and then logs the error in system tables. If OFE is set in the STATUS register, more than one error occurred since the last invocation of the interrupt handler, and the ERRADR register has recorded only the first.

If the error is single-bit, it is corrected by the ECM, and the interrupt handler simply returns and continues execution of the process that encountered the error. If the error is multi-bit, it probably is not corrected, and the process is killed. Note that multi-bit errors may indicate bus problems or failed control circuitry.

If any chip fails permanently, a single-bit error may occur on every access. In this case, invoking the interrupt handler on single-bit errors would be prohibitively time-consuming. When the handler detects frequent single-bit errors in a given chip, it logs the chip as defective and inhibits the single-bit-error interrupt on that specific board. It is important to strap both MBE and ERR to Multibus interrupt lines so that multi-bit errors will continue to interrupt after single-bit-error interrupts have been disabled.

APPENDIX A STRAPPING OPTIONS

INTRODUCTION

This appendix describes the full strapping options and switch settings for Valid ECM board 710-00005, Rev B and includes the default factory jumper positions. Jumper straps are either suit case jumpers between terminal posts or, as with interrupt level selection, insulated wire-wrap jumpers.

The following conventions are used in the strapping configuration table:

Default jumper strap positions are noted by an asterisk (*).

An equals sign (=) indicates a single possible connection between the pin listed to the left of the equals sign and one of the pins listed to the right of the equals sign.

Board Location	Strap	Description
16AA	2 = 1*	Enable Refresh
	3	Disable Refresh
26AA	2 = 1*	Enable High-Speed (AS*) Mode
	3	Disable High-Speed (AS*) Mode
12AA	2 = 1*	Inhibit XACK* During AACK* Cycles
	3	Enable XACK* During AACK* Cycles
30BB	2 = 1*	Drive AACK* on P1-25
	3	Drive AACK* on P2-40
29BB	2 = 1*	Use X.BLCK* (if using clock from bus)
	3	Use X.CCLK* (if using clock from bus)
30CC	1 = 30DD1*	MBE Interrupt to Multibus Level 0
	2 = 30DD1*	ERR Interrupt to Multibus Level 0
	= 30DD2	Multibus Level 1
	= 30DD3	Multibus Level 2
	= 30DD4	Multibus Level 3
	= 30DD5	Multibus Level 4
	= 30DD6	Multibus Level 5
	= 30DD7	Multibus Level 6
	= 30DD8	Multibus Level 7

Board Location	Strap	Description
20BB	2 = 1* 3	Drive CLK from bus (inverted) Drive CLK Internally at 8Mhz
20BB	5 = 4* 6	Drive CLK* from bus Drive CLK* Internally at 8MHz
22BB	2 = 1* 3	Correct all Reads and Writes Inhibit all Error Correction
30UB	2 = 1* 3	Receive X.A<23>* on P2-56 Receive X.A<23>* on P2-60
30UA	2 = 1* 3	Receive X.A<22>* on P2-55 Receive X.A<22>* on P2-59

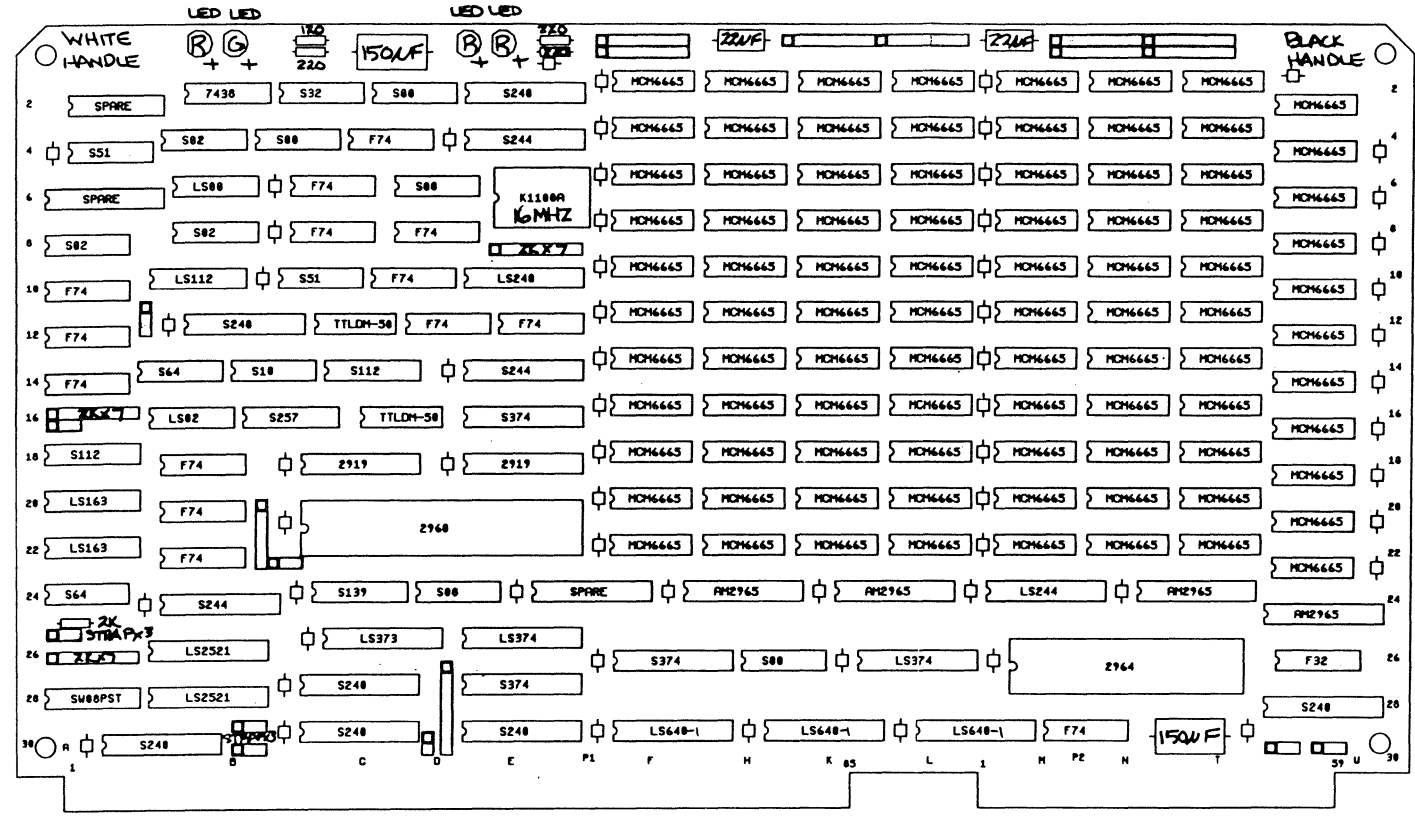
SWITCHES:

The switches at 28A select both the 512K byte memory bank and the 256 byte I/O block. Switch 1 is the most significant bit; "OFF" is a "1," and "ON" is a "0."

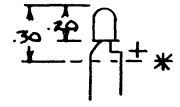
Location	Switches	Description
28A	4..8	Memory Bank Address A<23..19>
28A	1..8	I/O Block Address A<15..8>

APPENDIX B
ASSEMBLY DRAWING AND SILKSCREEN

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED



VALID ECM
873-8885
REVISION-B
ASSEMBLY



* BEND LED 90°
TOWARDS TOP OF
BOARD
MOUNT LED
THIS SET
OF HOLES
VIEW "A"

- NOTES:
1. SEE VIEW "A" FOR LED MOUNTING
 2. COMPONENTS TO BE PLACED IN SOCKETS
 3. INSTALL WIRE WRAP POSTS
 4. PLACE WHITE EJECTOR HANDLE IN UPPER LEFT.
 5. MASK THESE AREAS
 6. ALL CAPS 0.1MF UNLESS OTHERWISE NOTED.
 7. DO NOT STUFF

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± .XXX ±	CONTRACT NO.		ECM 710-00005-B ASSEMBLY DWG.
	APPROVALS	DATE	
MATERIAL	DRAWN	6-10-82	PC VERSION
	CHECKED	6-10-82	
FINISH			SIZE
DO NOT SCALE DRAWING			B
			DRAWING NO.
			REV B
			SCALE NONE
			SHEET OF

APPENDIX C


SPECIFICATIONS

Physical:	Height 6.75 in. Length 12.00 in.
Electrical:	+5V <u>+5%</u> at 3.3A (typical)
Temperature:	0 C to +50 C
Relative Humidity:	10% to 70% (non-condensing)

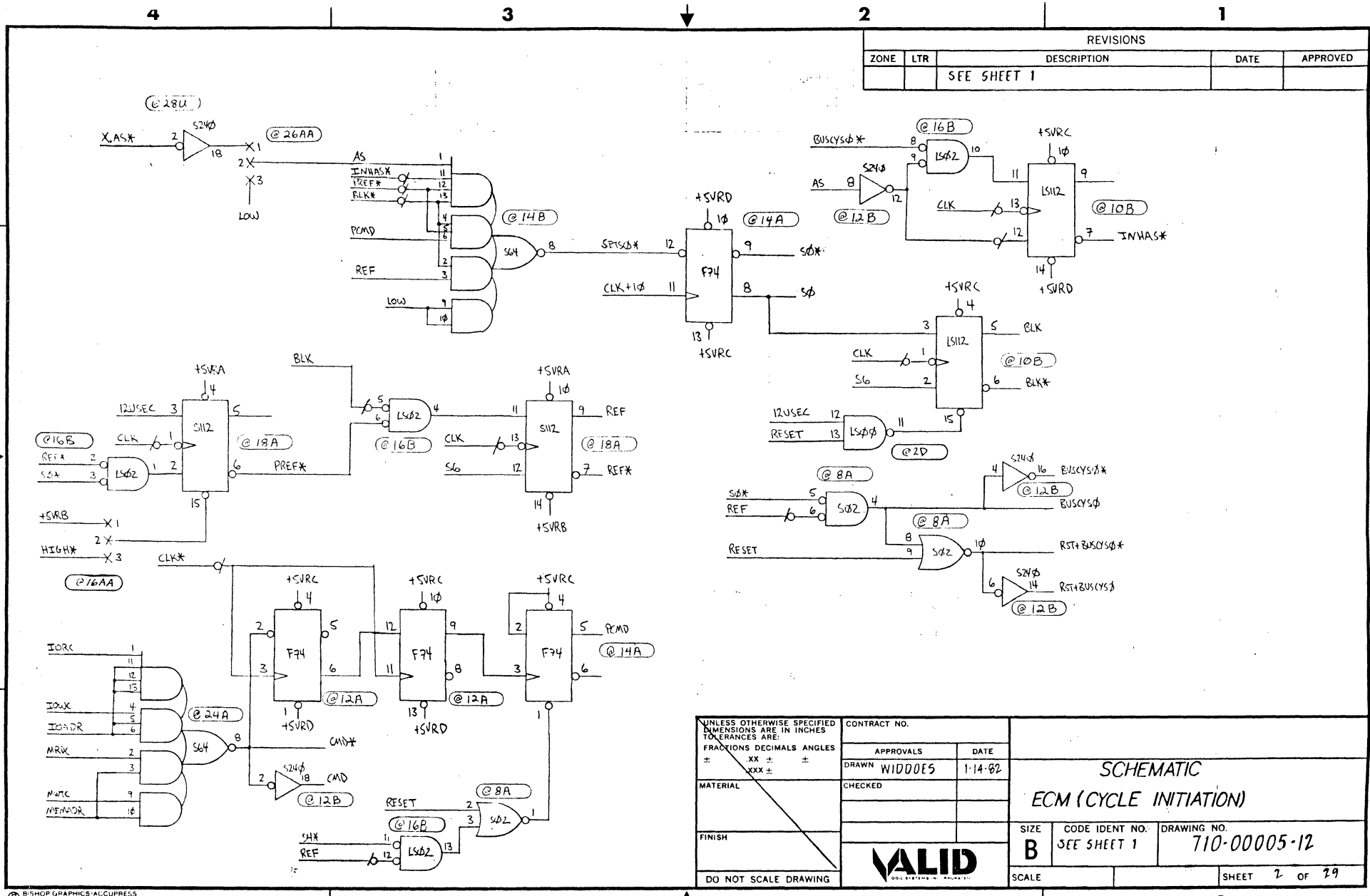
NOTE: THIS PAGE PROVIDES A RUNNING HISTORY OF ALL CHANGES ASSOCIATED WITH THIS DWG

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED

REV	SHEETS	DESCRIPTION	DATE	APPROVED	REV	SHEETS	DESCRIPTION	DATE	APPROVED
A		AS ISSUED PER FCO #104	10-27-82	<i>C. Cogn</i>					

<small>UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ±</small>	CONTRACT NO.		<p style="text-align: center;">SCHEMATIC ERROR CORRECTING MEMORY</p>						
	APPROVALS	DATE							
	DRAWN <i>WIDDOWS</i>	1-14-82							
	CHECKED <i>C. J. [Signature]</i>	10-27-82							
MATERIAL	<i>C. Cogn</i>	10-27-82	<table border="1"> <tr> <td>SIZE</td> <td>CODE IDENT NO.</td> <td>DRAWING NO.</td> </tr> <tr> <td>B</td> <td>REV A</td> <td>710-00005-12</td> </tr> </table>	SIZE	CODE IDENT NO.	DRAWING NO.	B	REV A	710-00005-12
SIZE	CODE IDENT NO.	DRAWING NO.							
B	REV A	710-00005-12							
FINISH									
DO NOT SCALE DRAWING	SCALE	SHEET 1 OF 2							

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± ± .XXX ± ±	CONTRACT NO.		SCHEMATIC ECM (CYCLE INITIATION)
	APPROVALS	DATE	
MATERIAL	CHECKED		SIZE B
FINISH			CODE IDENT NO. SEE SHEET 1
DO NOT SCALE DRAWING			DRAWING NO. 710-00005-12
	SCALE		SHEET 2 OF 29

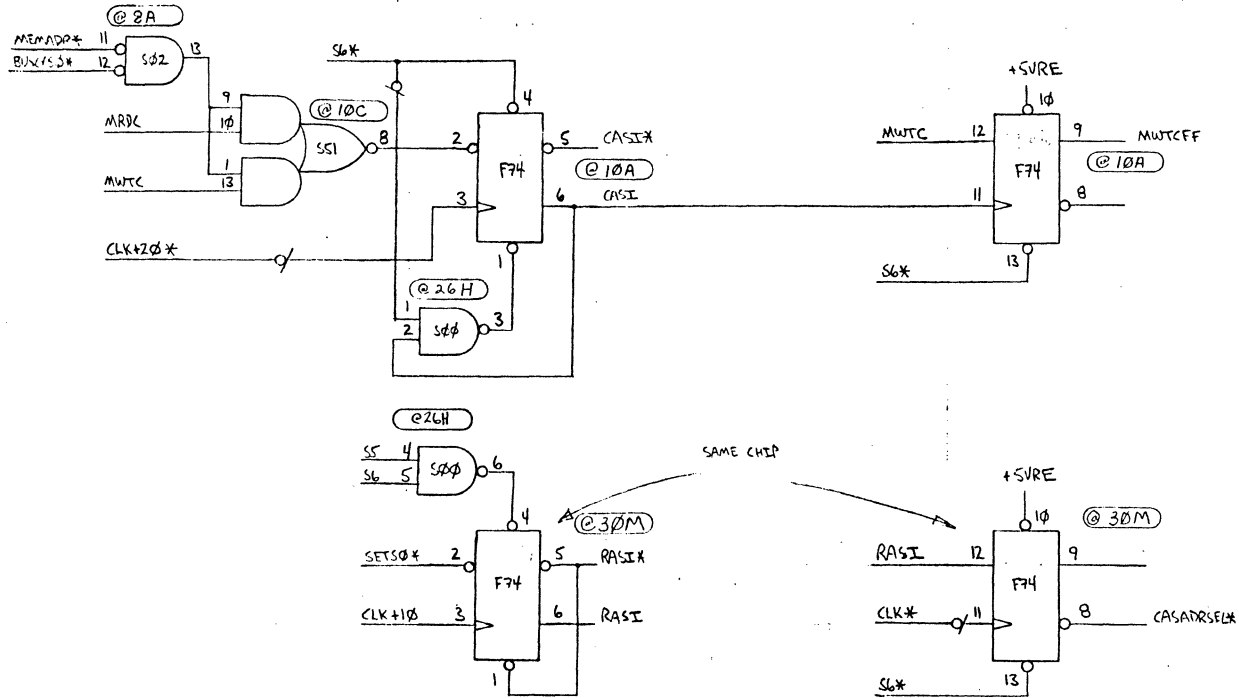
4

3

2

1

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ± ±	CONTRACT NO.		SCHEMATIC ECM (RASI AND CASI CONTROL)		
	APPROVALS	DATE			
	DRAWN WIDDOWS	1-14-82			
	CHECKED				
MATERIAL			SIZE B	CODE IDENT NO. SEE SHEET 1	DRAWING NO. 710-00005-12
FINISH					SHEET 3 OF 29
DO NOT SCALE DRAWING			SCALE		

4

3

2

1

4

3

2

1

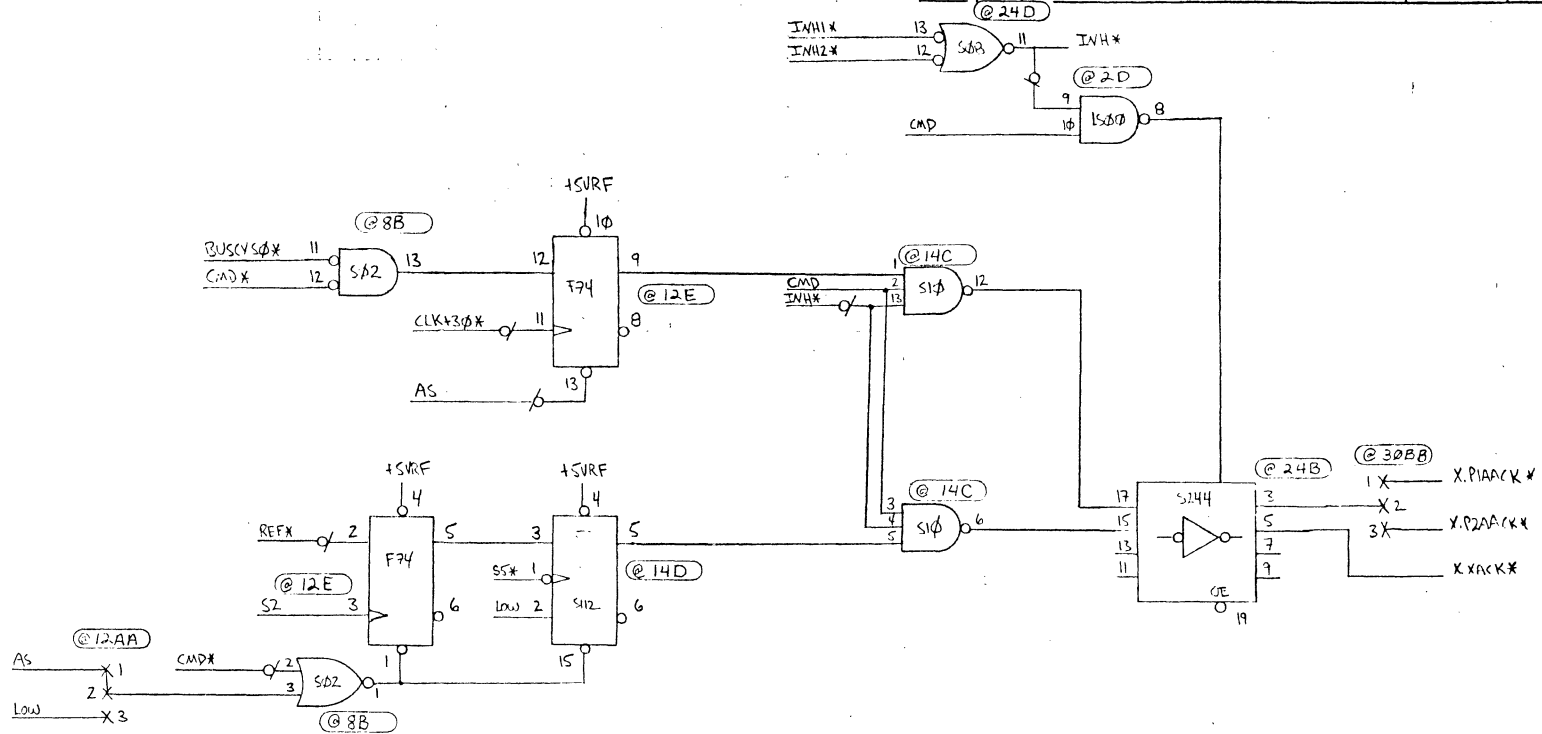
D

C

B

A

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ±	CONTRACT NO.		<p align="center">SCHEMATIC ECM (XACK AND AACK)</p>
	APPROVALS	DATE	
	DRAWN WIDDOWES	1-14 82	
	CHECKED		
MATERIAL			SIZE B CODE IDENT NO. SEE SHEET 1 DRAWING NO. 710-00005-12
FINISH			
DO NOT SCALE DRAWING	VALID		SCALE SHEET 4 OF 29

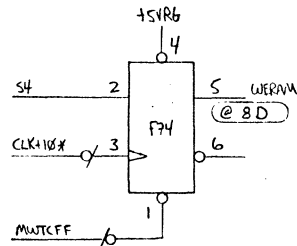
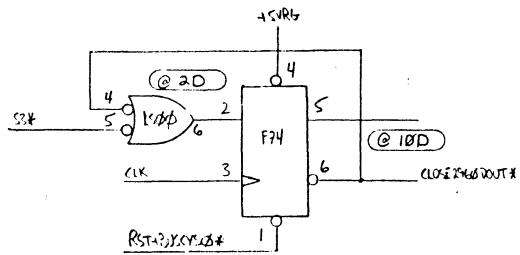
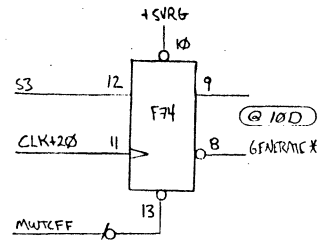
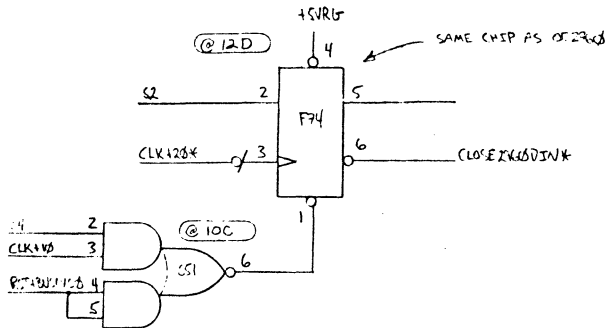
4

3

2

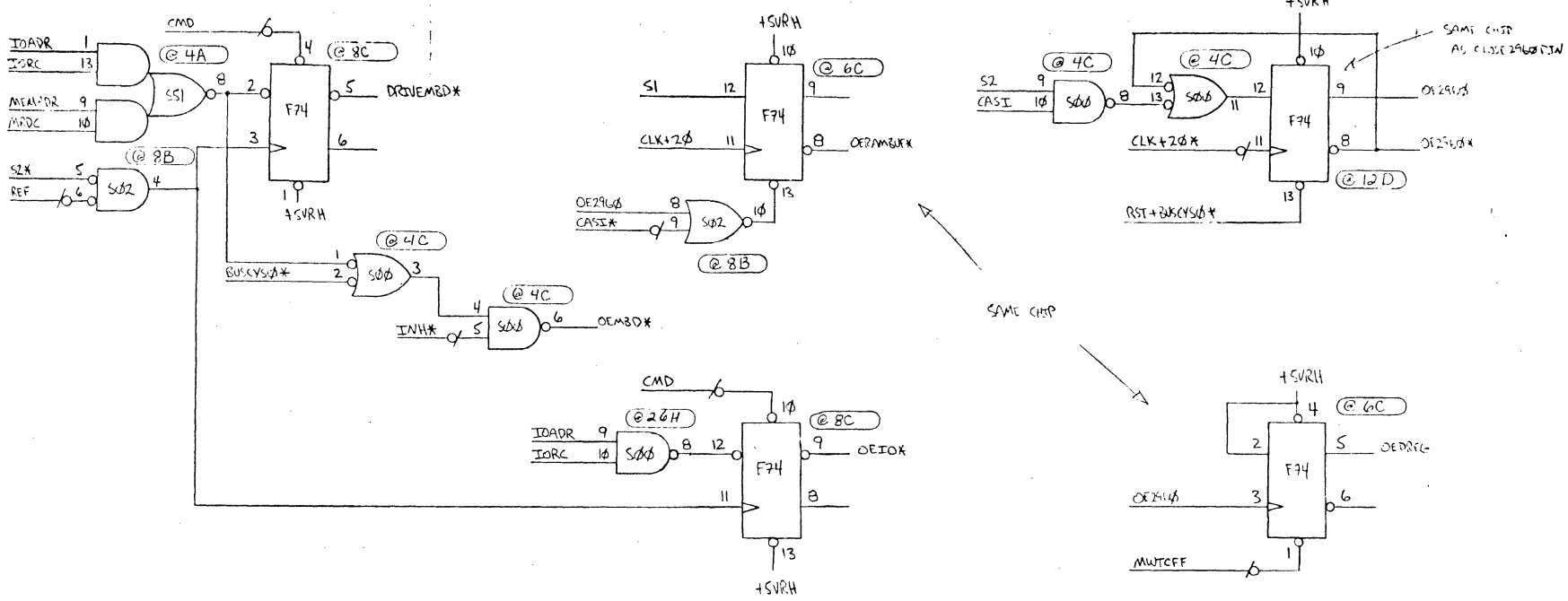
1

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		

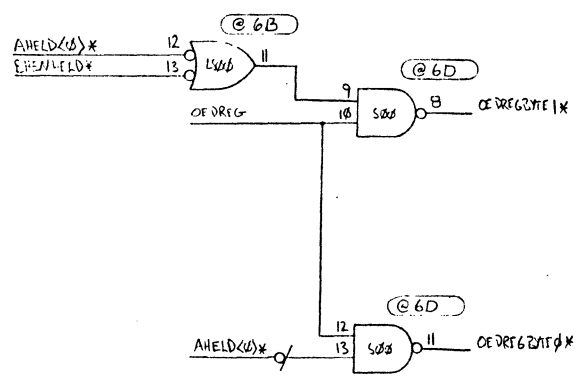
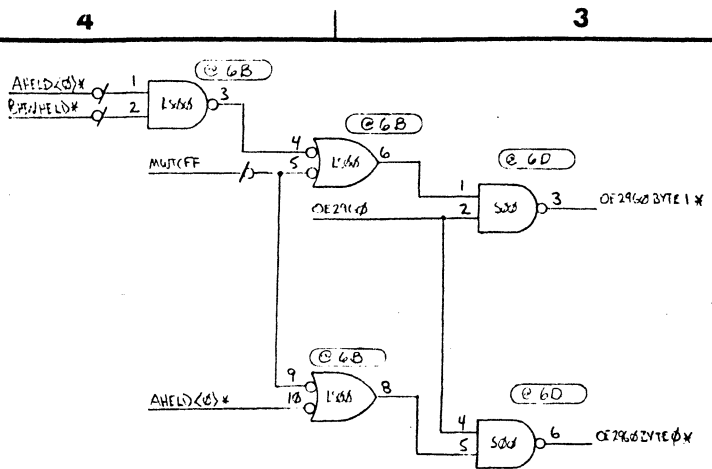


UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ± ±	CONTRACT NO.		SCHEMATIC			
	APPROVALS	DATE				
	MATERIAL	DRAWN WIDDOES	1-14-82	ECM (MISCELLANEOUS ENABLE 1/3)		
	FINISH	CHECKED		SIZE B	CODE IDENT NO. SEE SHEET 1	DRAWING NO. 710-00005-12
DO NOT SCALE DRAWING	VALID		SCALE	SHEET 5 OF 29		

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES. TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± ± .XXX ± ±	CONTRACT NO.		SCHEMATIC ECM(MISCELLANEOUS ENABLE 2/3)
	APPROVALS	DATE	
	DRAWN WIDDDES	1-14-82	
	CHECKED		
MATERIAL			
FINISH			
DO NOT SCALE DRAWING	VALID		
	SIZE B	CODE IDENT NO. SEE SHEET 1	DRAWING NO. 710-00005-12
	SCALE		SHEET 6 OF 29



REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ±	CONTRACT NO.		SCHEMATIC		
	APPROVALS	DATE			
	MATERIAL	DRAWN WIDDOWES	1-14-82	ECM (MISCELLANEOUS ENABLE 3/3)	
	FINISH	CHECKED		SIZE B	CODE IDENT NO. SEE SHEET 1
DO NOT SCALE DRAWING	VALID		SCALE	SHEET 7 OF 29	

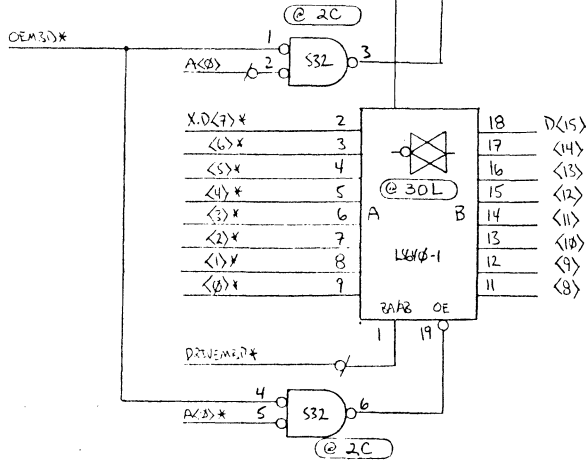
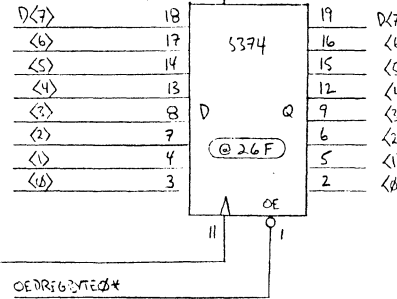
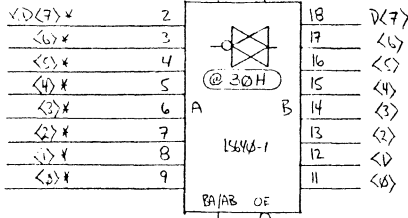
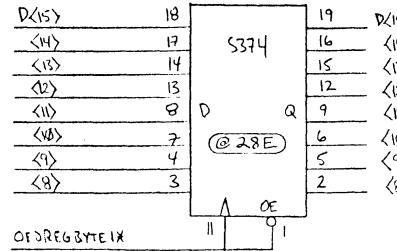
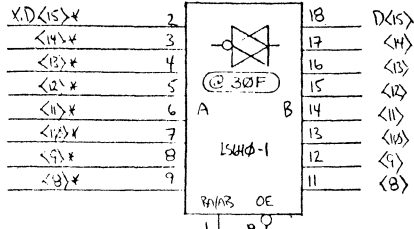
4

3

2

1

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ±	CONTRACT NO.		SCHEMATIC ECM (DATA BUFFERS)	
	APPROVALS	DATE		
	MATERIAL	CHECKED	DRAWN WIDDOWS	1-14-82
	FINISH			
DO NOT SCALE DRAWING	VALID	SIZE B	CODE IDENT NO. SEE SHEET 1	DRAWING NO. 710-00005-12
		SCALE		SHEET 8 OF 29

4

3

A

2

1

A

B

C

I

D

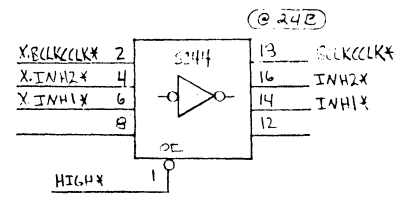
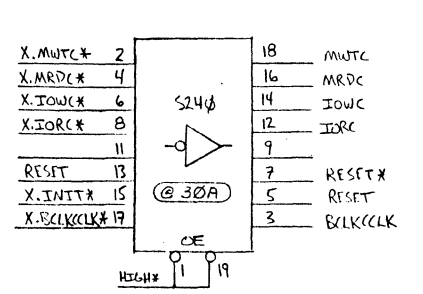
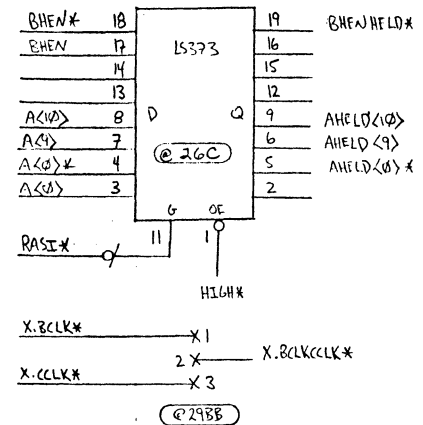
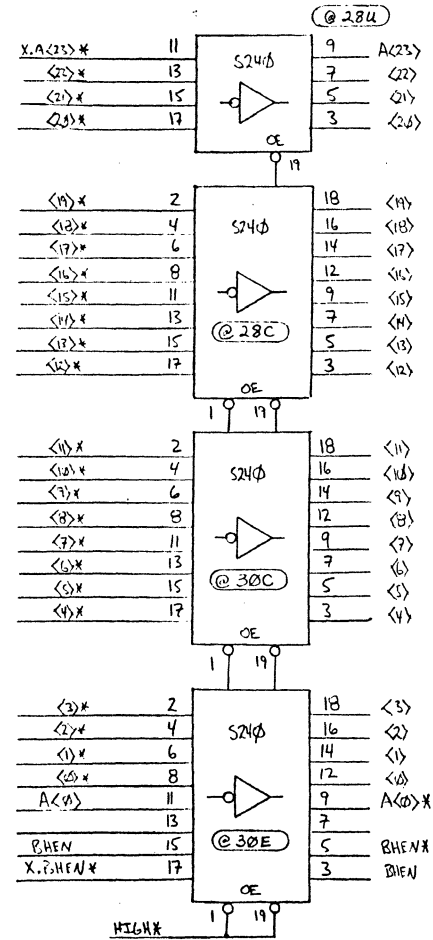
C

A

B

A

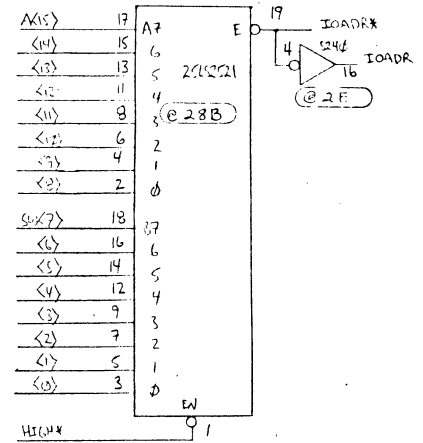
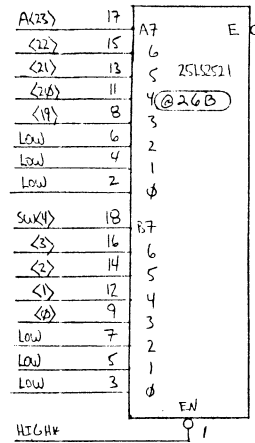
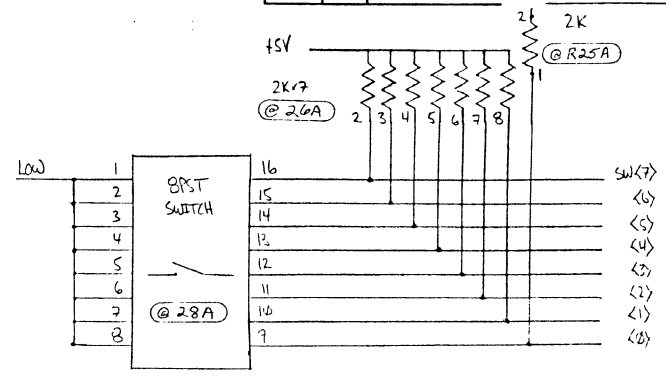
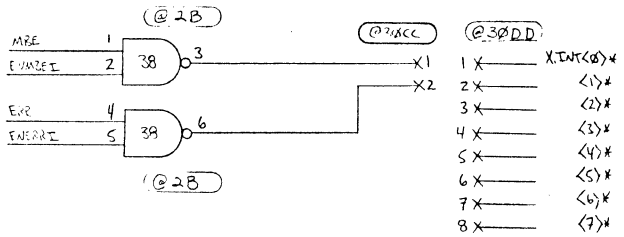
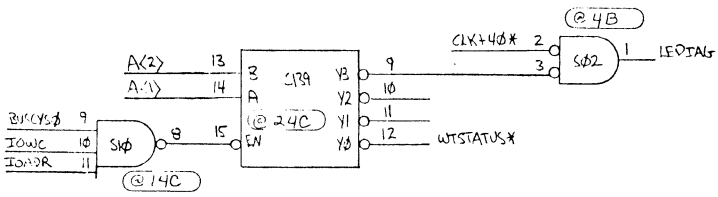
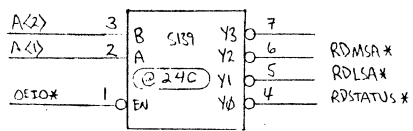
REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± ± .XXX ± ±	CONTRACT NO.			
	APPROVALS	DATE		
	DRAWN WIDDOWES	1-14-82	SCHEMATIC ECM (ADDRESS & CONTROL BUFFER)	
	CHECKED			
MATERIAL		SIZE B	CODE IDENT NO. SEE SHEET 1	DRAWING NO. 710-00005-12
FINISH		SCALE		SHEET 9 OF 29

4 3 2 1

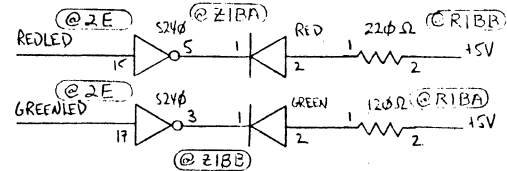
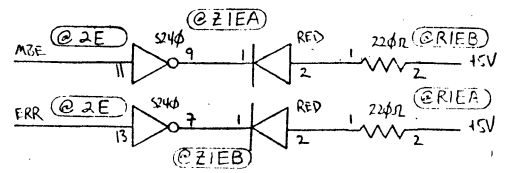
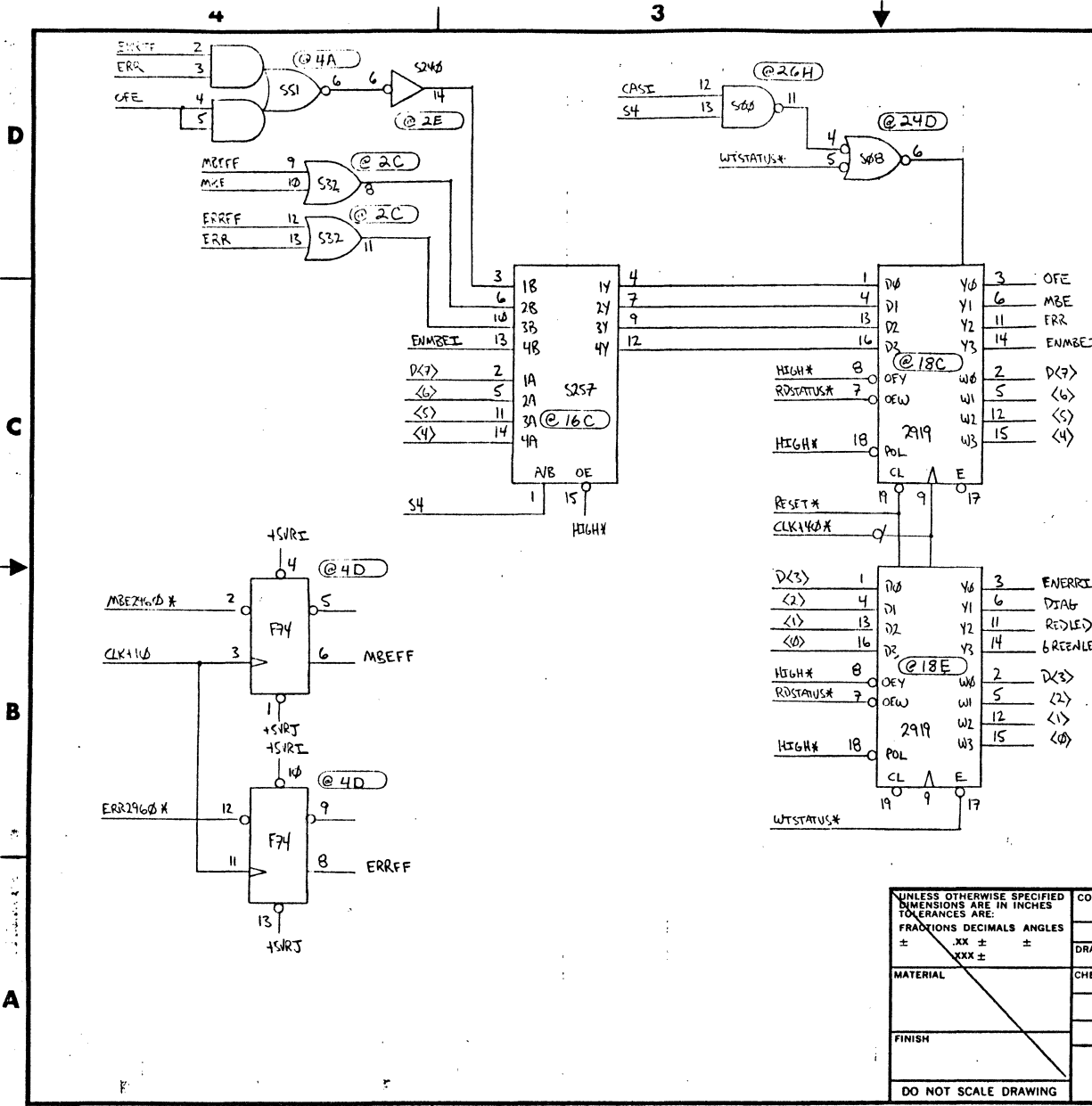
REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ± ±	CONTRACT NO.		SCHEMATIC ECM (ADDRESS DECODE)		
	APPROVALS	DATE			
	MATERIAL	DRAWN WIDDOS	1-14-82	SIZE B CODE IDENT NO: SEE SHEET 1 DRAWING NO. 710-00005-12	
	FINISH	CHECKED			
DO NOT SCALE DRAWING	VALID		SCALE	SHEET 10 OF 29	

4 3 2 1

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ± ±	CONTRACT NO.		SCHEMATIC ECM (STATUS REGISTER)
	APPROVALS	DATE	
	DRAWN WIDDOES	1-14-82	
	CHECKED		
MATERIAL			SIZE B CODE IDENT NO. SEE SHEET 1 DRAWING NO. 710-00005-12
FINISH			
DO NOT SCALE DRAWING	VALID <small>GOOD BY DESIGN</small>		SCALE
			SHEET 11 OF 29

4

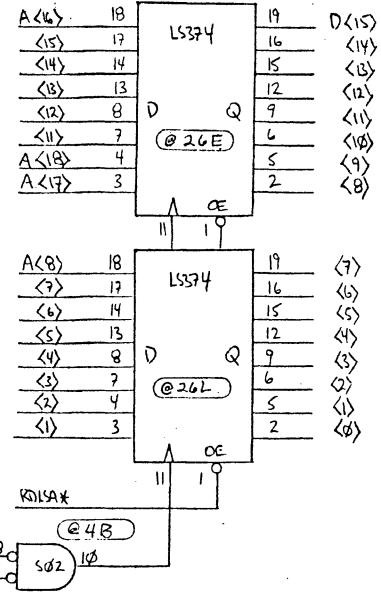
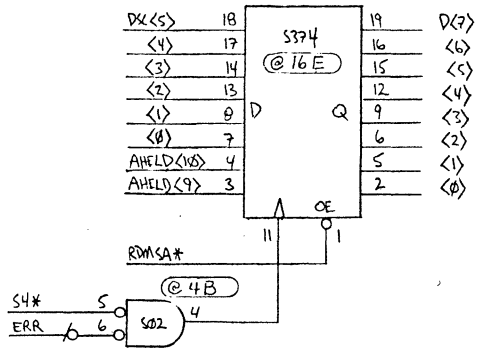
3

2

1

REVISIONS

ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± MATERIAL FINISH DO NOT SCALE DRAWING	CONTRACT NO.		APPROVALS		DATE	SCHMATIC ECM (ERRADA REGISTER)
	DRAWN WIDDOWES				1-14-82	
	CHECKED					
	FINISH					
SIZE		CODE IDENT NO.	DRAWING NO.			
B		SEE SHEET 1	710-00005-12			
SCALE				SHEET 12 OF 29		

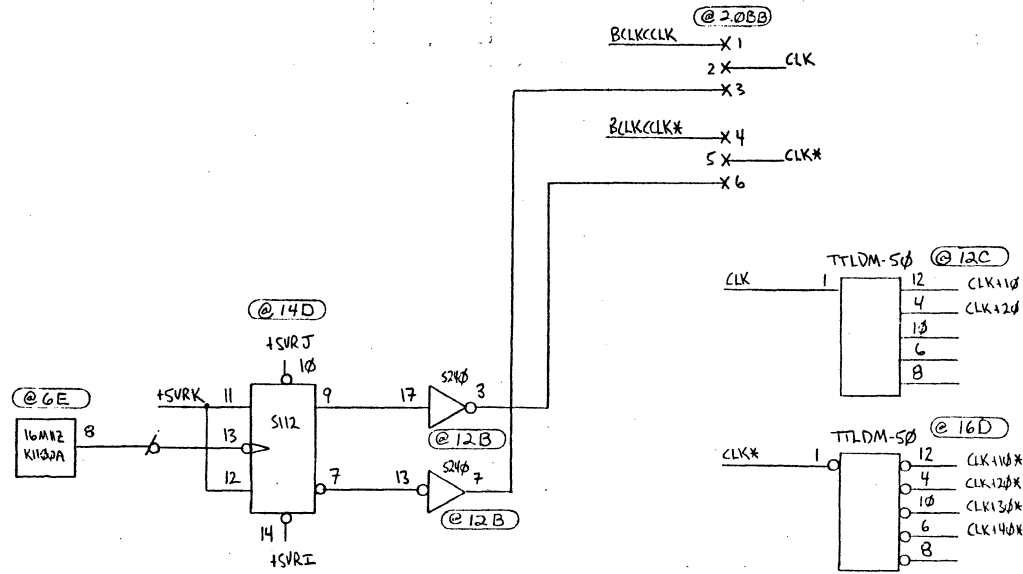
4

3

2

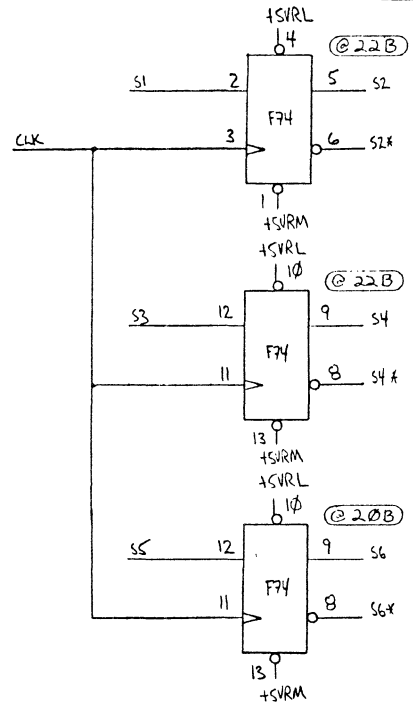
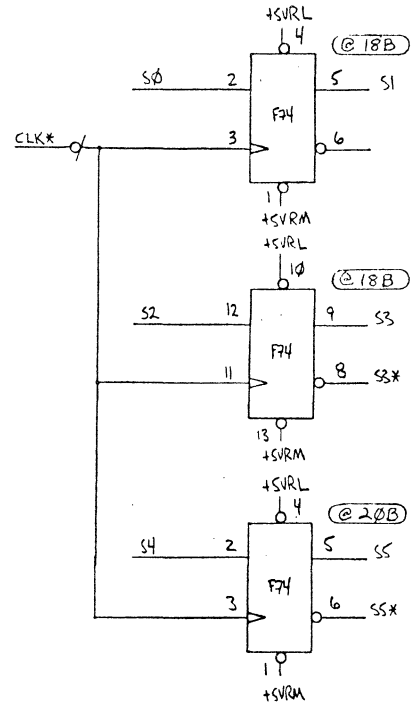
1

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ±	CONTRACT NO.		SCHEMATIC ECM (CLK GENERATOR)		
	APPROVALS	DATE			
	DRAWN WIDDOWS	1-14-82			
	CHECKED				
MATERIAL			SIZE B	CODE IDENT NO. SEE SHEET 1	DRAWING NO. 710-00005-12
FINISH			SCALE	SHEET 13 OF 29	
DO NOT SCALE DRAWING					

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ±	CONTRACT NO.		SCHEMATIC ECM (STATE COUNTER)	
	APPROVALS	DATE		
	DRAWN WIDDOES	1-14-82		
	CHECKED			
MATERIAL			SIZE B	CODE IDENT NO. SEE SHEET 1
FINISH			DRAWING NO. 710-00005-12	
DO NOT SCALE DRAWING	VALID		SCALE	SHEET 14 OF 29

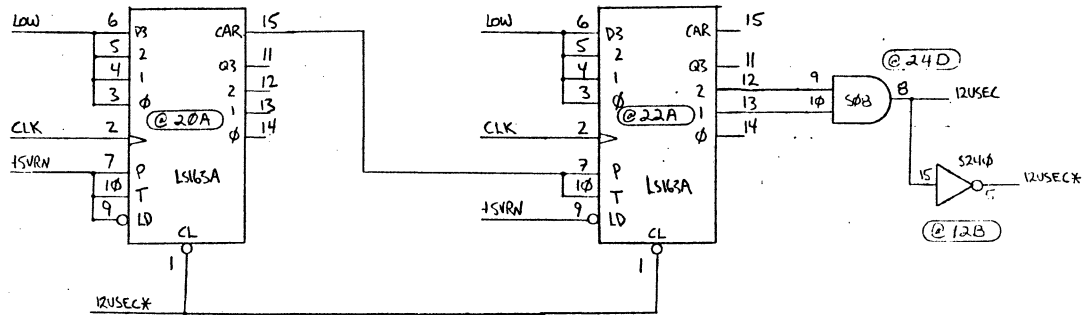
4

3

2

1

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ±	CONTRACT NO.		SCHEMATIC ECM (REFRESH TIMER)
	APPROVALS	DATE	
	DRAWN WIDDOES	1-14-82	
	CHECKED		
MATERIAL			
FINISH			
DO NOT SCALE DRAWING	VALID LOGIC SYSTEMS TECHNOLOGICAL	SIZE B	CODE IDENT NO. SEE SHEET 1
			DRAWING NO. 710-00005-12
		SCALE	SHEET 15 OF 29

4

3

2

1

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		


D<15>	2	DIS	SC6	25	
<14>	3	14	5	29	DSL<5>
<11>	4	13	4	26	<4>
<12>	5	12	3	28	<3>
<11>	9	11	2	27	<2>
<10>	10	10	1	24	<1>
<9>	11	9	0	30	<0>
<8>	12	8			
<7>	14	7	DEC	31	HELT*
<6>	15	6	MRE	33	MRE 2960*
<5>	16	5	ERR	32	ERR 2960*
<4>	17	4			
<3>	20	3			
<2>	21	2			
<1>	22	1			
<0>	23	0			
LOW	41	CB6			
DCP<5>	39				@ 20C
<4>	38	5			
<3>	37	4			
<2>	35	3			
<1>	34	2			
<0>	40	1			
LOW	45	0			
EXP<5>	19	LEIN			
EXP<5>	18	LENT			
EXP<5>	8	OE1			
EXP<5>	18	OE0			
LOW	45	CID2			
LOW	44	CID1			
LOW	43	CID0			
LOW	47	MODE1			
DIAG	46	MONTE0			
GENERATE*	42	GENERATE			
LOW	48	DIAG			
LOW	2X	DIAG			
LOW	X3	DIAG			
DIAG	7	DIAG			

AM2960

@ 20C

TSVRN X 1
 LOW 2X
 X 3

@ 2288

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ± ±	CONTRACT NO.		SCHEMATIC ECM (AM2960)
	APPROVALS	DATE	
	DRAWN WIDDOES	1-14-82	
	CHECKED		
MATERIAL			SIZE B CODE IDENT NO. SEE SHEET 1 DRAWING NO. 710-00005-12
FINISH			
DO NOT SCALE DRAWING			SCALE SHEET 16 OF 29

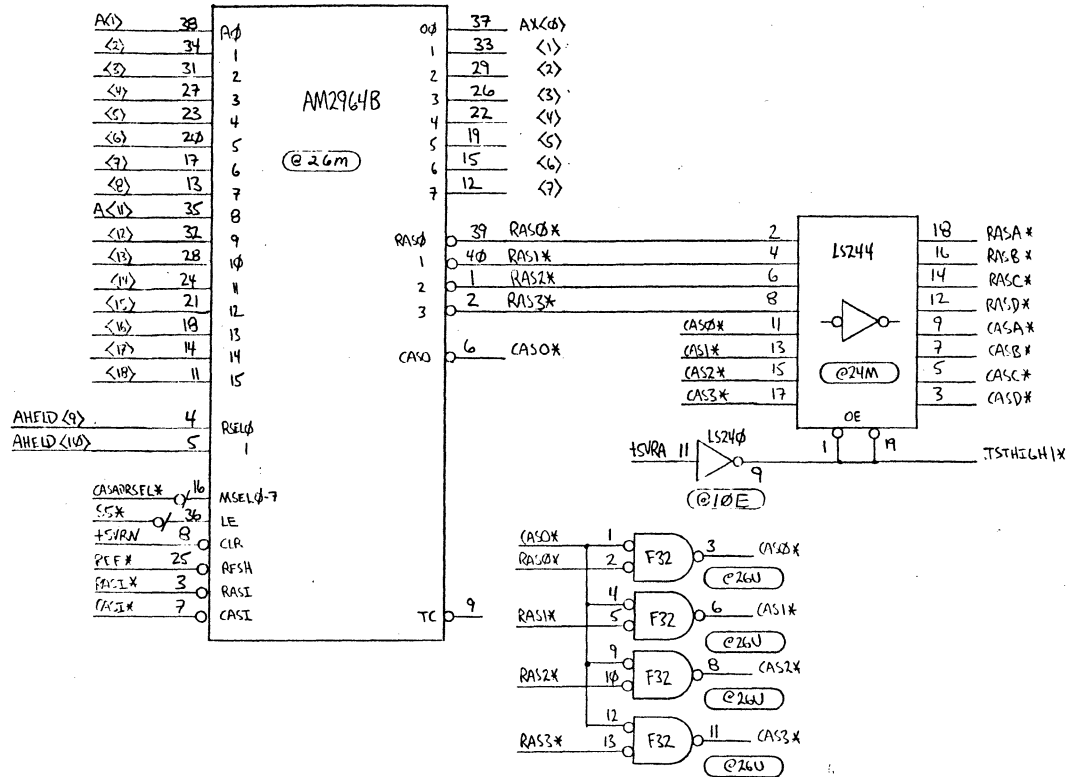
4

3

2

1

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± MXX ±	CONTRACT NO.		SCHMATIC ECM (AM2964B)		
	APPROVALS	DATE			
	DRAWN WIDDDES	1-14-82			
	CHECKED				
MATERIAL			SIZE	CODE IDENT NO.	DRAWING NO.
FINISH			B	SEE SHEET 1	710-00005-12
DO NOT SCALE DRAWING			SCALE	SHEET 17 OF 29	

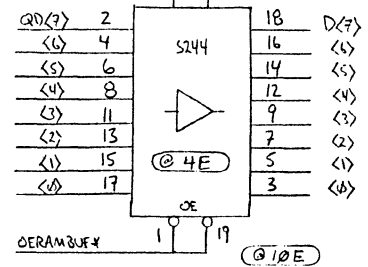
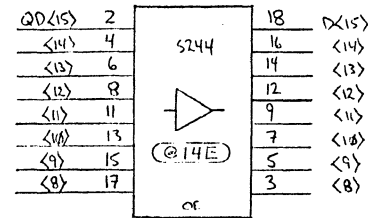
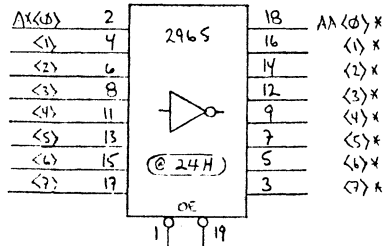
4

3

2

1

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		




UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ±	CONTRACT NO.		SCHEMATIC ECM (RAM DRIVERS)		
	APPROVALS	DATE			
	DRAWN WIDDOES	1-14-82			
	CHECKED				
MATERIAL			SIZE	CODE IDENT NO.	DRAWING NO.
FINISH			B	SEE SHEET 1	710-00005-12
DO NOT SCALE DRAWING	VALID LOGIC SYSTEMS INCORPORATED		SCALE	SHEET 18 OF 29	

AA <7> *
 AA <6> *
 AA <5> *
 AA <4> *
 AA <3> *
 AA <2> *
 AA <1> *
 AA <0> *
 WEA *
 CASA *
 RASA *

		4	15	3	5	7	6	12	11	10	13	9	NC	1	
D<0>	2	666SA-15											14	QD<0>	
D<2>	2	Ø2F											14	QD<2>	
D<4>	2	Ø4F											14	QD<4>	
D<6>	2	Ø6F											14	QD<6>	
D<8>	2	Ø8F											14	QD<8>	
D<10>	2	Ø10F											14	QD<10>	
D<12>	2	Ø12F											14	QD<12>	
D<14>	2	Ø14F											14	QD<14>	
DS<0>	2	Ø16F											14	QB<0>	
DS<2>	2	Ø18F											14	QB<2>	
DS<4>	2	Ø20F											14	QB<4>	
DS<6>	2	Ø22F											14	QB<6>	

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		


UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ±	CONTRACT NO.		SCHEMATIC ECM (A-EVEN RAMS)		
	APPROVALS	DATE			
	MATERIAL	DRAWN WIDDDES	1:14-82	SIZE B	CODE IDENT NO. SEE SHEET 1
	FINISH	CHECKED		DRAWING NO. 710-00005-12	
DO NOT SCALE DRAWING			SCALE	SHEET 19 OF 29	

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		

AA<7>*
 AA<6>*
 AA<5>*
 AA<4>*
 AA<3>*
 AA<2>*
 AA<1>*
 AA<0>*
 WEA*
 CAA*
 RASA*

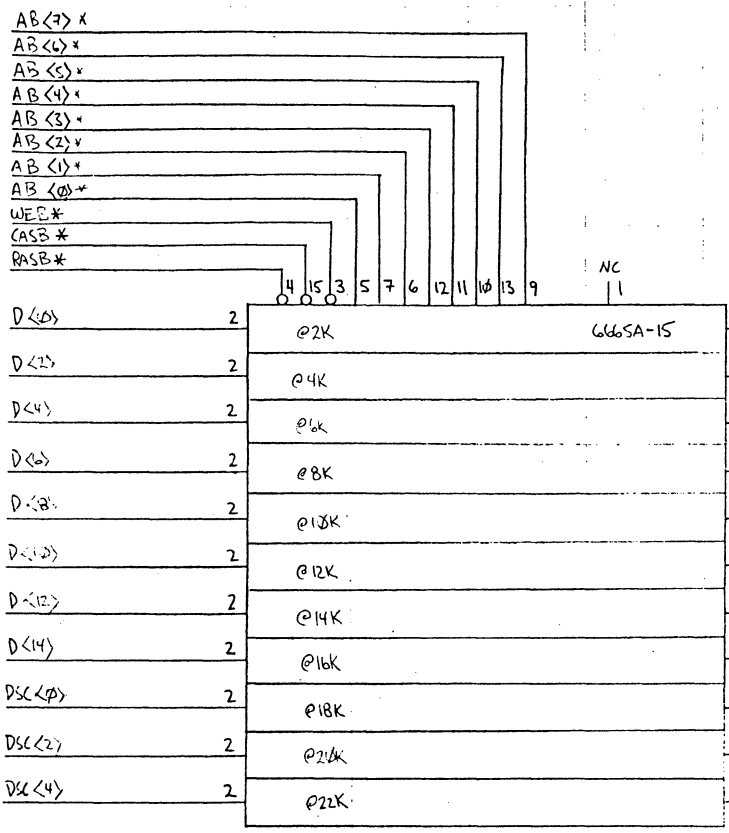
4 15 3 5 7 6 12 11 16 13 9 NC 11

D<1>	2	Ø2H	666SA-15	14	ØD<1>
D<3>	2	Ø4H		14	ØD<3>
D<5>	2	Ø6H		14	ØD<5>
D<7>	2	Ø8H		14	ØD<7>
D<9>	2	Ø10H		14	ØD<9>
D<11>	2	Ø12H		14	ØD<11>
D<13>	2	Ø14H		14	ØD<13>
D<15>	2	Ø16H		14	ØD<15>
DSC<1>	2	Ø18H		14	ØCB<1>
DSC<3>	2	Ø21H		14	ØCB<3>
DSC<5>	2	Ø22H		14	ØCB<5>

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES $\pm .XX \pm \pm$ $\pm .XXX \pm \pm$	CONTRACT NO.		SCHEMATIC ECM (A-ODD RAMS)	
	APPROVALS	DATE		
	MATERIAL	DRAWN WIDDOES	1-14-82	SIZE B CODE IDENT NO. SEE SHEET 1 DRAWING NO. 710-00005-12
	FINISH	CHECKED		
DO NOT SCALE DRAWING			SCALE	SHEET 20 OF 29

D
C
B
A

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		



UNLESS OTHERWISE SPECIFIED
DIMENSIONS ARE IN INCHES
TOLERANCES ARE
FRACTIONS DECIMALS ANGLES
± .XX ± ±
XXX ±

MATERIAL

FINISH

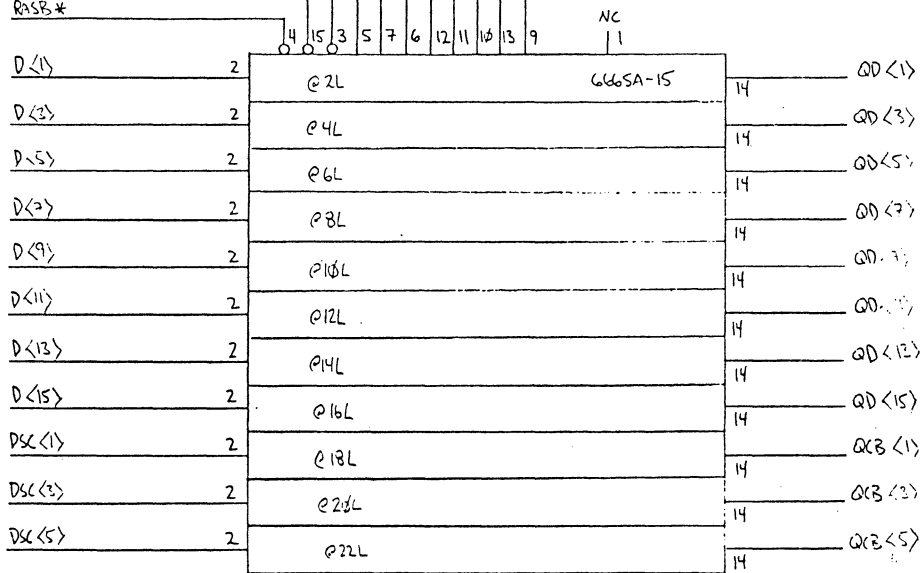
DO NOT SCALE DRAWING

CONTRACT NO.	
APPROVALS	DATE
DRAWN WIDDOES	1-14-82
CHECKED	

SCHEMATIC ECM (B-EVEN RAMS)		
SIZE B	CODE IDENT NO. SEE SHEET 1	DRAWING NO. 710-00005-12
SCALE	SHEET 21 OF 29	

D
C
B
A

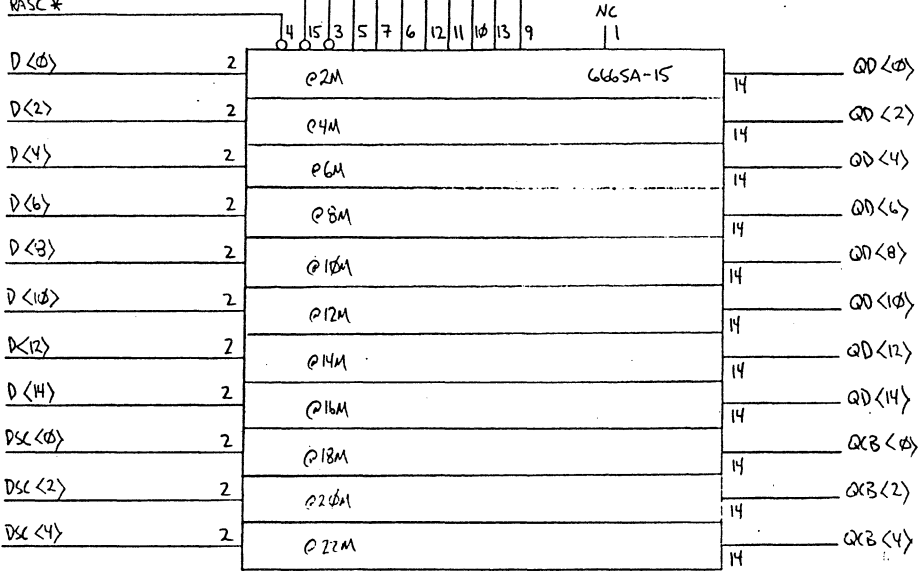
AB <7> *
 AB <6> *
 AB <5> *
 AB <4> *
 AB <3> *
 AB <2> *
 AB <1> *
 AB <0> *
 WEB *
 CASB *
 RASB *



REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± ± .XXX ±	CONTRACT NO.		SCHEMATIC ECM (B-ODD RAMS)	
	APPROVALS	DATE		
	DRAWN WIDDOWES	1-14-82		
	CHECKED			
MATERIAL			SIZE	CODE IDENT NO.
FINISH			B	SEE SHEET 1
DO NOT SCALE DRAWING		SCALE		DRAWING NO. 710-00005-12
				SHEET 17 OF 29

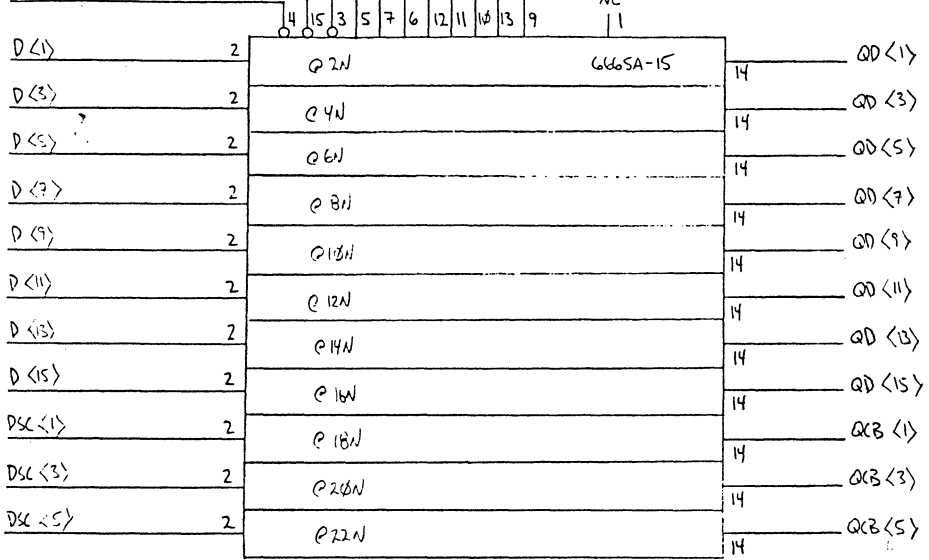
AC <7>*
 AC <6>*
 AC <5>*
 AC <4>*
 AC <3>*
 AC <2>*
 AC <1>*
 AC <Ø>*
 WEC*
 CASC*
 RASC*



REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± ± .XXX ± ±	CONTRACT NO.		SCHEMATIC ECM (G-EVEN RAMS)		
	APPROVALS	DATE			
	DRAWN WIDDOES	1-14-82			
	CHECKED				
MATERIAL			SIZE	CODE IDENT NO.	DRAWING NO.
FINISH			B	SEE SHEET 1	710-00005-12
DO NOT SCALE DRAWING	VALID ONE SYSTEMS INTERNATIONAL		SCALE	SHEET 13 OF 29	

AC <7> *
 AC <6> *
 AC <5> *
 AC <4> *
 AC <3> *
 AC <2> *
 AC <1> *
 AC <0> *
 WEC *
 CASC *
 RASC *



REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		

<small>UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ±</small>	CONTRACT NO.		SCHEMATIC ECM (C-ODD RAMS)		
	APPROVALS	DATE			
	DRAWN WIDDOWS	1-14-02			
	CHECKED				
MATERIAL			SIZE	CODE IDENT NO.	DRAWING NO.
FINISH			B	SEE SHEET 1	710-00005-12
DO NOT SCALE DRAWING			SCALE	SHEET 24 OF 29	

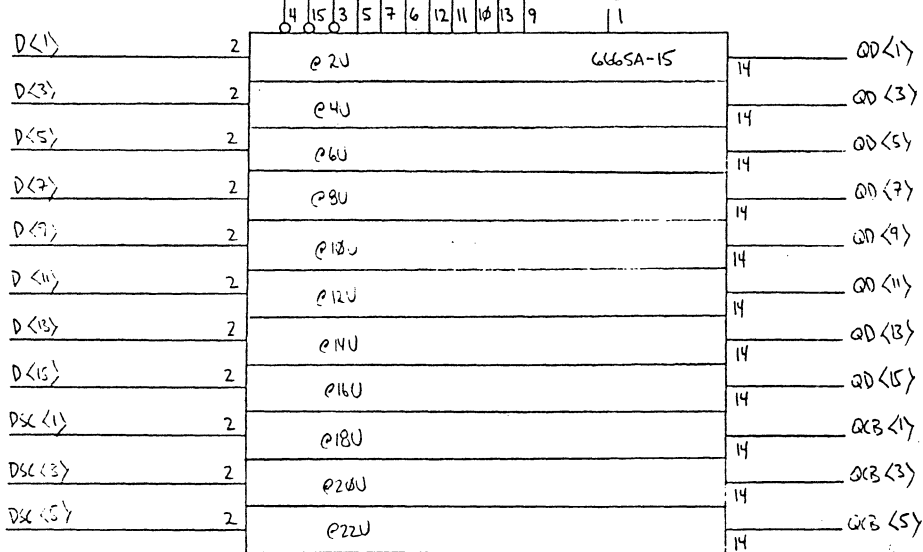
AD <7> *
 AD <6> *
 AD <5> *
 AD <4> *
 AD <3> *
 AD <2> *
 AD <1> *
 AD <0> *
 WED *
 CASD *
 RASD *

		4	15	3	5	7	6	12	11	10	13	9	NC			
D <0>	2													6665A-15	14	OD <0>
D <2>	2														14	OD <2>
D <4>	2														14	OD <4>
D <6>	2														14	OD <6>
D <8>	2														14	OD <8>
D <10>	2														14	OD <10>
D <12>	2														14	OD <12>
D <14>	2														14	OD <14>
D <16>	2														14	OD <16>
D <18>	2														14	OD <18>
D <20>	2														14	OD <20>
D <22>	2														14	OD <22>

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ±	CONTRACT NO.		APPROVALS		DATE
	DRAWN WIDDOES				1-14-82
	MATERIAL	CHECKED		SCHEMATIC ECM (D-EVEN RAMS)	
	FINISH				
DO NOT SCALE DRAWING		SIZE B	CODE IDENT NO. SEE SHEET 1	DRAWING NO. 710-00005-12	
		VALID		SHEET 25 OF 29	

AD<7>*
 AD<6>*
 AD<5>*
 AD<4>*
 AD<3>*
 AD<2>*
 AD<1>*
 AD<Ø>*



REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .xx ± ± xxx ±	CONTRACT NO.		SCHEMATIC ECM (D-ODD RAMS)	
	APPROVALS	DATE		
MATERIAL	DRAWN WIDDDES	1-14-82	DRAWING NO. 710-00005-12	
FINISH	CHECKED			
DO NOT SCALE DRAWING	VALID LOGIC SYSTEMS INCORPORATED		SIZE B	CODE IDENT NO. SEE SHEET 1
			SCALE	SHEET 26 OF 29

4

3

2

1

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		

* R.162/266x6

AN<0>* 2	[]	@IFA
<2>* 3		
<1>* 4		
<3>* 5		
<4>* 6		
<5>* 7		

* R.162/266x6

AB<0>* 2	[]	@IK
<2>* 3		
<1>* 4		
<3>* 5		
<4>* 6		
<5>* 7		

* R.162/266x6

AC<0>* 2	[]	@INB
<2>* 3		
<1>* 4		
<3>* 5		
<4>* 6		
<5>* 7		

* R.162/266x6

PAS0* 2	[]	@ITB
AD<2>* 3		
AD<1>* 4		
<3>* 5		
<4>* 6		
<5>* 7		

* R.162/266x6

CPSAX 2	[]	@IFB
WEA* 3		
NC 4		
PA<6>* 5		
PA<7>* 6		
PASAX* 7		

* R.162/266x6

AR<7>* 2	[]	@IL
CASB* 3		
WEB* 4		
AR<6>* 5		
PASR* 6		
NC 7		

* R.162/266x6

AC<6>* 2	[]	@INA
<7>* 3		
PASC* 4		
CASC* 5		
WEC* 6		
NC 7		

* R.162/266x6

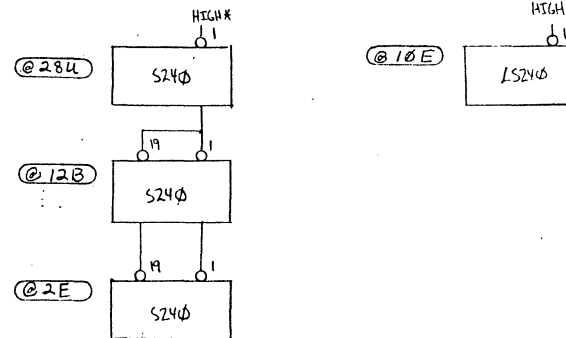
NC 2	[]	@ITA
AD<0>* 3		
AD<1>* 4		
CASD* 5		
WED* 6		
AD<7>* 7		

R.2Kx7

+SVRA 2	[]	@16A
+SVRB 3		
+SVRC 4		
+SVRD 5		
+SVRE 6		
+SVRF 7		
+SVRG 8		

R.2Kx7

+SVRI 2	[]	@8E
+SVRII 3		
+SVRIII 4		
+SVRIV 5		
+SVRIV 6		
+SVRII 7		
+SVRII 8		



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ±	CONTRACT NO.		SCHEMATIC ECM (TERMINATORS)		
	APPROVALS	DATE			
	DRAWN WIDDONES	1-14-82			
	CHECKED				
MATERIAL			SIZE	CODE IDENT NO.	DRAWING NO.
FINISH			B	SEE SHEET 1	710-00005-12
DO NOT SCALE DRAWING	VALID		SCALE	SHEET 27 OF 29	

4

3

2

1

D

C

B

A

B

A

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		

	P1
X.A<19>*	34
<19>*	32
<17>*	30
<16>*	28
<15>*	44
<14>*	43
<13>*	46
<12>*	45
<11>*	48
<10>*	47
<9>*	50
<8>*	49
<7>*	52
<6>*	51
<5>*	54
<4>*	53
<3>*	56
<2>*	55
<1>*	58
<0>*	57
X.INT<7>*	36
<6>*	35
<5>*	38
<4>*	37
<3>*	40
<2>*	39
<1>*	42
<0>*	41

	P1
X.D<15>*	60
<14>*	59
<15>*	62
<12>*	61
<11>*	64
<10>*	63
<9>*	66
<8>*	65
<7>*	68
<6>*	67
<5>*	70
<4>*	69
<3>*	72
<2>*	71
<1>*	74
<0>*	73
X.INIT*	14
NC	17
X.MRD*	19
X.MWTC*	20
X.TORC*	21
X.TOWC*	22
X.XACK*	23
X.PIACK*	25
X.BHEN*	27
NC	29

	P1
X.BPRN*	15
	16
X.BCLK*	13
X.CCLK*	31
X.INH1*	24
X.INH2*	26

	P2
X.AS*	39
X.P2ACK*	40
@ 30UB	
X.A<23>*	56
1X	60
3X	60
@ 30UA	
X.A<22>*	55
1X	59
3X	59
X.A<21>*	58
X.A<20>*	57

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ± ±	CONTRACT NO.		SCHEMATIC ECM (MULTIBUS CONNECTORS)	
	APPROVALS	DATE		
MATERIAL	DRAWN	WIDDOES	1-14-82	SIZE B
FINISH	CHECKED			
DO NOT SCALE DRAWING	VALID CIRCUIT SYSTEMS INCORPORATED		CODE IDENT NO. SEE SHEET 1	DRAWING NO. 710-00005-12
	SCALE		SHEET 20 OF 29	

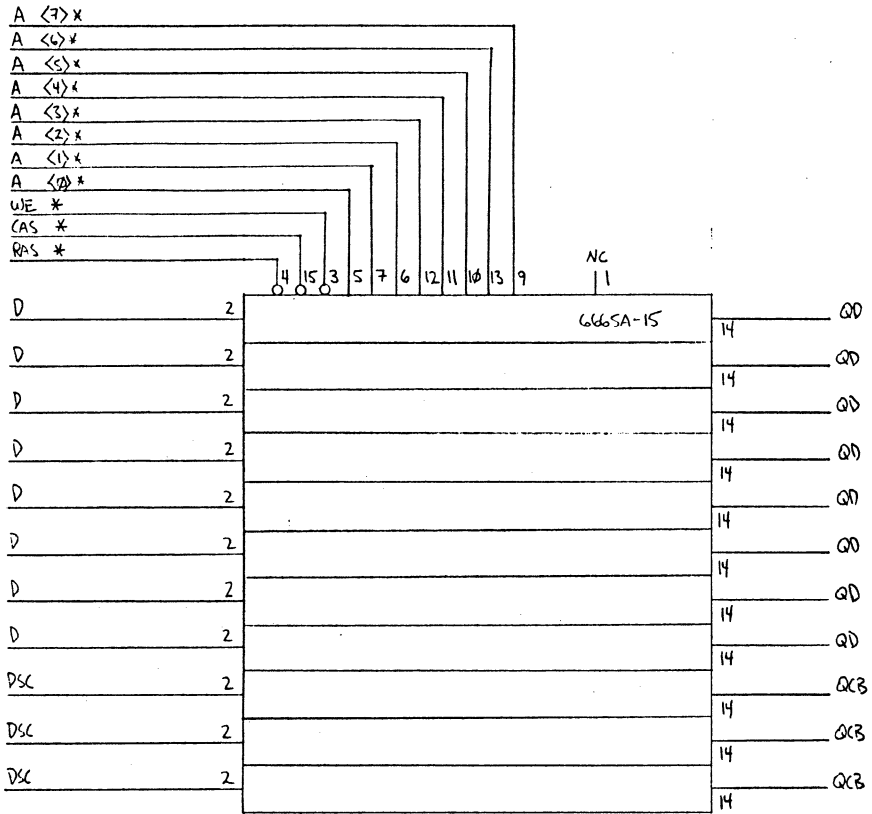
4

3

2

1

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES \pm $XX \pm$ \pm $XXX \pm$	CONTRACT NO.		SCHEMATIC ECM (RAM MASTER)		
	APPROVALS	DATE			
	MATERIAL	DRAWN WIDDOES	1-14-82	SIZE B	CODE IDENT NO. SEE SHEET 1
	FINISH	CHECKED		DRAWING NO. 710-00005-12	
DO NOT SCALE DRAWING			SCALE	SHEET 29 OF 29	

Video Graphics Board User Manual

VED-041582-1 Revision 4-15-82 (GSM)

01 Aug 84

Copyright 1984

Valid Logic Systems Incorporated

This document contains confidential proprietary information which is not to be disclosed to unauthorized persons without the written consent of an officer of Valid Logic Systems Incorporated.

The copyright notice appearing above is included to provide statutory protection in the event of unauthorized or unintentional public disclosure.

1.0 INTRODUCTION

The Valid Logic Systems Video Graphics Board (VGB) is a Multibus-compatible board that is used as the graphics engine for SCALDsystem work station. The work station itself consists of:

- o A high resolution, high performace, raster-scanned, CRT-based graphics display. The graphics display is refreshed at a rate of 60 frames per second from an on-board frame buffer. The image is 800 lines by 1024 pixels, each pixel at one of four intensity levels. The board generates a single composite video signal compatible with a variety of standard CRT monitors.
- o A keyboard and a graphics tablet/puck. These input devices communicate with the VGB over two RS-232C serial ports at any standard baud rate up to 38.4Kbaud.

The VGB offers the following features:

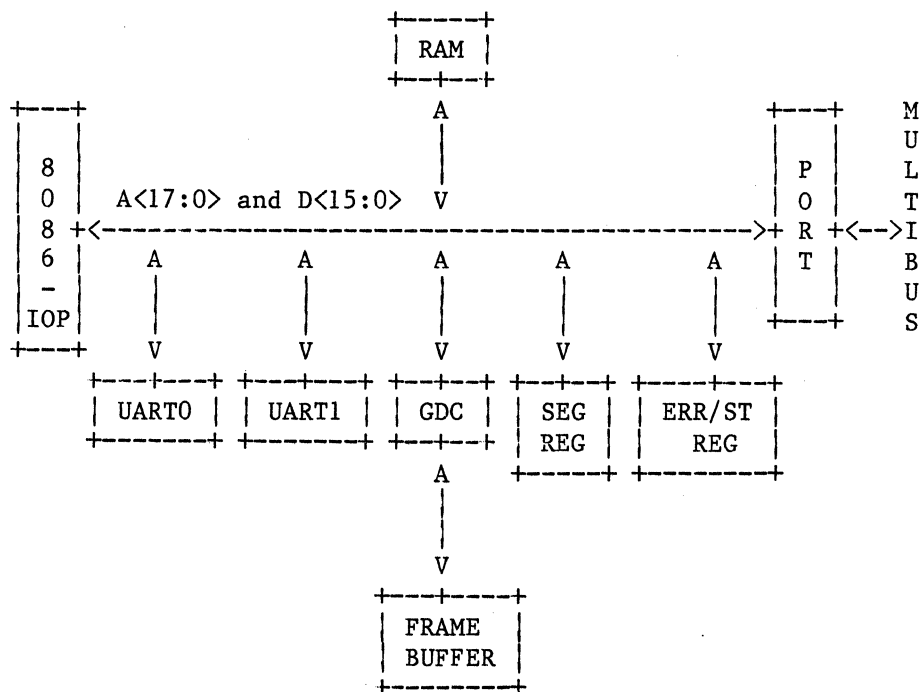
- o 800- by 1024-pixel display
- o 60Hz refresh rate
- o Composite video output
- o Four levels of intensity per pixel
- o Line, arc, circle and area-fill hardware
- o 224- by 1024-pixel graphics scratch pad memory
- o On-board control (input/output) processor
- o 2KBy of local RAM for the IOP, upgradeable to 16KBy
- o On-board status, error and memory-relocation registers
- o Full interface to 16M byte IEEE-796 bus (Intel Multibus)
- o RS-232C keyboard interface port
- o RS-232C graphics-tablet interface port
- o Maintenance port for testing and diagnosis

In order to achieve fast response at the work station without over-burdening the CPU, the VGB includes two microprocessors. The first microprocessor is a graphics display controller or "GDC" (an NEC uPD7220). This device controls the refresh of the CRT from the frame buffer and provides hardware assistance for rapidly drawing key graphic primitives such as lines and arcs. The second

microprocessor is an Intel 8086 that functions as an input/output processor or "IOP." This device handles high-frequency local computations such as scanning of display lists held in Multibus memory space, character generation, windowing, scaling, clipping, and the low-level processing associated with the keyboard and tablet. The IOP executes programs that reside either in Multibus memory or in on-board RAM (or both). Communication between the IOP and the CPU use shared memory and a simple interrupt scheme.

2.0 VIDEO GRAPHICS BOARD ARCHITECTURE

The architecture of the VGB allows the board to operate as autonomously as possible by shifting a great deal of the load associated with the work station away from the CPU and the Multibus. The overall structure consists of the IOP driving an on-board bus with six devices attached as shown in the following illustration.



All devices are memory mapped with the following address assignments (in IOP address space):

- 00000 ... 03FFF : on-board RAM
- 04000 ... 04007 : UART0
- 04008 ... 0400F : UART1
- 04010 : segment register (SREG)
- 04018 : status and error register (ERR/ST)
- 06000 : GDC status
- 06800 : GDC command
- 07000 : GDC DMA port
- 10000 ... 1FFFF : Multibus IO access
- 30000 ... 3FFFF : Multibus memory access

During Multibus access, a full 24-bit Multibus address is formed by concatenating SEGREG<7..0> with A<15..0>. The high-order address bits (A<23..20>) are driven on connector P2. Two different standards exist for the positions of A<22> and A<23> on P2. These bits are therefore driven according to both standards to allow the use of memory or peripheral boards that conform to either standard. The positions of the non-standard Multibus signals are as follows:

Pin(s)	Signal
56, 60	A<23>
55, 59	A<22>
58	A<21>
57	A<20>

The Multibus is acquired and released using precisely the same bus arbitration logic that is used on the CPU board. This arbitration logic has two strappable options. One option allows locking of the the bus, and the other selects between giving the bus away after each bus cycle or only surrendering the bus when it is requested by another bus master.

Video Graphics Board User Manual

The VGB also functions as a Multibus slave. Devices on the VGB can be accessed by the CPU using Multibus reads and writes. For this purpose, the VGB occupies a 64K-byte window of Multibus memory address space. This window can be located on any 64K-byte boundary using on-board switches; each work station on a cluster must be assigned a unique 64K-byte window. Within this window, the CPU address space is the same as the IOP, except that A<17..15> of the on-board address bus are forced to zero during CPU accesses. Should the IOP generate a Multibus reference to its own 64K-byte window, a timeout will occur.

Devices on the VGB may not be accessed from the Multibus while the IOP is running. However, a memory write operation to board address 008000 with D<5> set clears the DEV.EN bit of the on-board status register whether or not the IOP is running. Clearing this bit, in turn, forces the IOP into its reset state and forces the board into slave mode.

To facilitate CPU/IOP communication, "door bell" interrupts are provided. There are two door bell bits in the status register. When DOORBELLO is set, an interrupt is sent to the IOP. When DOORBELL1 is set, an interrupt is sent to the host at a strappable priority level. In addition, straps are provided so that the IOP can sense one Multibus interrupt. A memory write operation to board address 008000 with D<1> set, sets the DOORBELLO bit of the on-board status register (see section 3.2) whether or not the IOP is running.

3.0 VIDEO GRAPHIC BOARD CAPABILITIES

The VG board has a great deal of local processing power and powerful local devices that can be used in a variety of ways. The main functions of these devices are outlined below.

3.1 The IOP

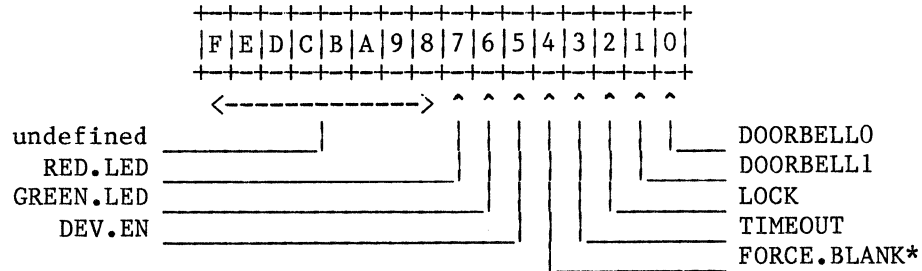
The IOP is a general-purpose microcomputer; its role on the board is determined mainly by its programming. (Further information on the 8086 used as the IOP can be found in Intel publications The 8086 Family User's Manual and the Component Data Catalog.) The IOP is intended to perform simple, frequently occurring tasks such as:

- o Reading a display list prepared by the CPU in Multibus memory.
- o Taking objects from the list and the current view specification, transforming the objects to screen coordinates, and then drawing the objects. These operations require windowing and clipping the objects and converting the resultant objects into GDC command sequences.
- o Low-level display-list processing such as locating the objects nearest the cursor.
- o Buffering characters from the keyboard and passing them to the CPU only when required (i.e., when <return> is pressed).
- o Receiving the pointer position from the tablet, smoothing position movement, converting the position from tablet to screen coordinates, and updating the cursor image on the screen.
- o Character generation from ASCII codes.
- o DMA transfers of frame-buffer images to and from Multibus memory.
- o Running local diagnostics.

Local on-board RAM contains most of the code and constant data required by the IOP. By using on-board RAM, most IOP wait states are eliminated and much of the VGB traffic from the Multibus is removed.

3.2 The Status Register

An on-board status register contains important board state information, some of which may be written. The register has eight fields (unless otherwise noted a field is read/write):



- o RED.LED lights the on-board red diagnostic LED if, and only if, it is set.
- o GREEN.LED lights the on-board green diagnostic LED if, and only if, it is set.
- o DEV.EN is normally strapped to the reset pin of the IOP. If clear, the IOP is forced to the reset state. Independent of the strapping, a second on-board red LED lights if, and only if, DEV.EN is clear. DEV.EN is not only read/write, but may be "asynchronously" cleared from the Multibus as described in section 2.0.
- o FORCE.BLANK blanks the CRT of the work station if, and only if, is clear.
- o TIMEOUT is set if no response to a bus cycle (on or off board) is received by the IOP after 16 microseconds. A timeout causes a non-maskable interrupt to be sent to the IOP that stacks some status information and executes the appropriate interrupt routine. Note: If the timeout occurs on a intra-segment jump instruction, the stacked IP is wrong. If the timeout occurs on an inter-segment jump, both the stacked IP and CS are wrong. TIMEOUT is a clear only bit. Writing the status register clears TIMEOUT.
- o If LOCK is set, the bus is not released after the next Multibus access; if LOCK is clear, the bus is released to any requestor regardless of priority.
- o If DOORBELL1 is set, an interrupt request is made to the host at a strappable priority level.
- o If DOORBELLO is set, an interrupt request is made to the IOP, at a strappable priority. DOORBELLO is not only read/write, but may be asynchronously set from the Multibus as described in section 2.0.

The status register is cleared on reset.

3.3 The Serial Ports

The VGB contains two serial ports implemented with 2661-2 UARTs and associated RS-232C drivers and receivers. Both ports are RS-232C compatible and capable of sending and receiving serial data up to 38.4KBaud.

One port (the keyboard port) is a minimal port that simply sends and receives serial data. The second port (the tablet port) sends and receives the most commonly used subset of modem control signals to communicate with the graphics data tablet. (The actual signals driven and received are - RxD, TxD, CTS, DSR, RTS, and DTR; RTS and DTR may optionally be strapped high.)

Each 2661 UART contains four 8-bit registers accessed by low-order byte addresses; A<2..1> selects the internal function. For additional information on the 2661 UART, refer to the vendor's documentation.

3.4 The GDC

The graphics display processor has three primary functions:

- o Creating images in the frame-buffer given a suitable set of commands from the IOP. The GDC primitives include line, arc, circle, and rectangle drawing and area-fill.
- o Generating sync, blanking and video signals (to be combined by the DAC).
- o Refreshing the dynamic RAMs of the frame-buffer.

The GDC is extremely fast and capable of drawing lines at a rate of approximately 100 microseconds per inch.

A portion (224x1024 pixels) of the GDC's frame-buffer is not displayed and is available as a scratch-pad memory for the GDC. Some uses of the scratch-pad memory are for "caching" characters that then can be drawn by simply performing block transfers within the frame-buffer, and "rasterizing" vector lists for Versatec plots. A full description of the GDC (uPD7220) can be found in the vendor's documentation.

4.0 MANUFACTURING AND MAINTENANCE

The VGB has been designed to operate with a number of RAM devices (for the frame buffer) from various manufacturers. Control signals for the RAMs are developed from delay lines. Each manufacturer's RAM requires a different delay line (all of which come from the same vendor however). Independent of the choice of RAMs, two control signals must be tuned to occur at the correct time relative to one clock edge.

Maintenance is enhanced by several features, for example, the IOP's ability to run local diagnostics. The results of these diagnostics are indicated via two on-board LEDs - a red LED and a green LED. Additionally, a maintenance connector is included on the board. This connector includes critical signals to be monitored for diagnosis faults as well as the signals that must be driven to single-step the IOP. Finally, since the VGB can operate as a Multibus slave, the system CPU itself can diagnose all on-board functions (except for correct operation of the IOP and the GDC).

The prototype board has the following strappable options:

- o The IOP may be reset either by the 796Bus reset signal or DEV.EN.
- o Relinquishing Multibus mastership on each clock cycle or only when there is an outstanding request.
- o Priority level of the DOORBELL1 interrupt.
- o Disabling DOORBELL1 interrupts.
- o Priority level of the Multibus interrupt (if any) monitored by the IOP.
- o RTS and DTR of the tablet port may be strapped high rather than received.

APPENDIX A

STRAPPING OPTIONS

INTRODUCTION

This appendix describes the full strapping options and switch setting for the VGB, assembly 710-00002, Rev B and includes the default factory jumper strap positions. Jumper straps are insulated wire-jumpers installed between the designated terminal posts.

The following conventions are used in the strapping configuration table:

Default jumper strap positions are noted by an asterisk (*).

An equals sign (=) indicates a single possible connection between the pin listed to the left of the equals sign and one of the pins listed to the right of the equals sign.

NC indicates that no jumper strap is installed.

JUMPER STRAPS

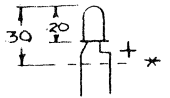
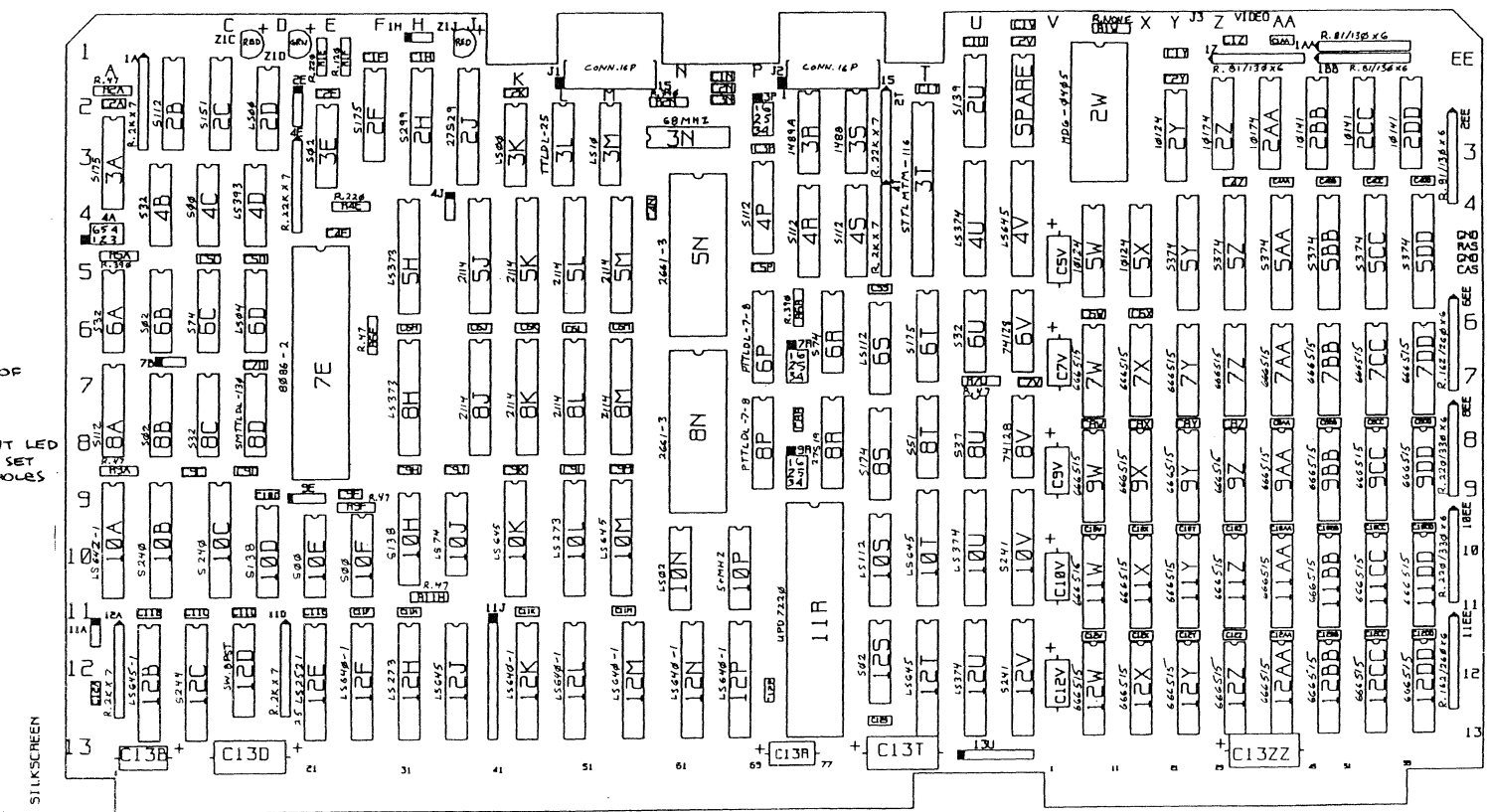
Board Location	Strap	Description
11C	2 = 3	VGB front end runs from CCLK
	1*	VGB front end runs from BCLK
2E	2 = 3	Fast ACK time for on-board cycles
	1*	Slow ACK time for on-board cycles
1H	2 = 1	Select 4K-byte on-board RAM
	NC*	Select 16K-byte on-board RAM
4A	2 = 3*	Disable software Multibus Lock
	1	Enable software Multibus Lock
	5 = 4*	Disable CBRQ
4J	6	Enable CBRQ
	2 = 1*	Select 16K-byte local RAM.
	NC	Select 4K-byte local RAM
9E	2 = 1*	Enable 8086 reset from status register
	3	Enable 8086 reset from bus INIT only

Video Graphics Board User Manual

7B	2 = 1 NC*	Configure VGB for head of BPRN chain Configure VGB to receive BPRN from higher-priority bus master
11A	2 = 1* NC	Interrupt 8086 each vertical sync time Disable vertical sync interrupt to 8086
11J	1 = 2* 3* 4 5 6 7 8 9 10	Doorbell interrupt from host Multibus Interrupt Level 7 Multibus Interrupt Level 6 Multibus Interrupt Level 5 Multibus Interrupt Level 4 Multibus Interrupt Level 3 Multibus Interrupt Level 2 Multibus Interrupt Level 1 Multibus Interrupt Level 0
3P	2 = 1 3* 5 = 4 6*	Assert CTS on tablet port Assert CTS when RTS is asserted Assert DSR on tablet port Assert DSR when DTR is asserted
8P	1 = 6 2 = 5 3 = 4	Advance RAS falling edge by 8ns Advance RAS falling edge by 16ns Advance RAS falling edge by 32ns
6P	1 = 6 2 = 5 3 = 4	Advance 2nd CAS rising edge by 8ns Advance 2nd CAS rising edge by 16ns Advance 2nd CAS rising edge by 32ns
13U	2 = 1* 3 5 = 4* 6	Multibus address A<22> on P2-59 Multibus address A<22> on P2-55 Multibus address A<23> on P2-60 Multibus address A<23> on P2-56

APPENDIX B
ASSEMBLY DRAWING

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED



* BEND LEAD 30°
TOWARDS TOP OF BOARD

MOUNT LEAD THIS SET OF HOLES

VIEW 'A'

SILKSCREEN
HALL LOGIC SYSTEM INC
3000 E. GRAND AVE
VAL10925

- NOTES:
1. INSTAL SOCKETS AT THE FOLLOWING LOC.
 2. INSTAL WIRE WRAP POSTS
 3. INSTAL SPECIAL SOCKET ON 3T (SMTH...)
 4. MASK THESE AREAS
 5. DO NOT INSTAL CONNECTOR AT J1
 6. SPARE DXX @ 2W .10 X 2 OF BOARD

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES # XX - # XXX - #	CONTRACT NO.		VIDEO GRAPHICS BOARD ASSEMBLY DRAWING 710-00002	
	APPROVALS		DATE	
MATERIAL	DRAWN P. JAMESON		3-1-82	
	CHECKED			
FINISH	SIZE		CODE IDENT NO.	
	DO NOT SCALE DRAWING		DRAWING NO.	
		B		SCALE NONE
		SHEET		OF

4

3

2

1

NOTE: THIS PAGE PROVIDES A RUNNING HISTORY OF ALL CHANGES ASSOCIATED WITH THIS DWG

REVISIONS			
ZONE	LTR	DESCRIPTION	DATE

REV	SHEETS	DESCRIPTION	DATE	APPROVED	REV	SHEETS	DESCRIPTION	DATE	APPROVED
A		AS ISSUED PER ECO #102	10-15-82	<i>G. Aguirre</i>					
A ₁		CORRECT SCHEMATIC ON SHT. 5 & 11 PER DCO#154	6-21-84	<i>G. Aguirre</i>					

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± ± .XXX ± ±	CONTRACT NO.		SCHEMATIC VIDEO GRAPHICS
	APPROVALS	DATE	
MATERIAL	DRAWN <i>TILLEMANN</i>	10-12-82	SIZE B CODE IDENT NO. REV A₁ DRAWING NO. 710-00002-2
FINISH	CHECKED <i>G. Aguirre</i>	10-15-82	
DO NOT SCALE DRAWING	VALID LOGIC SYSTEMS INCORPORATED		SCALE

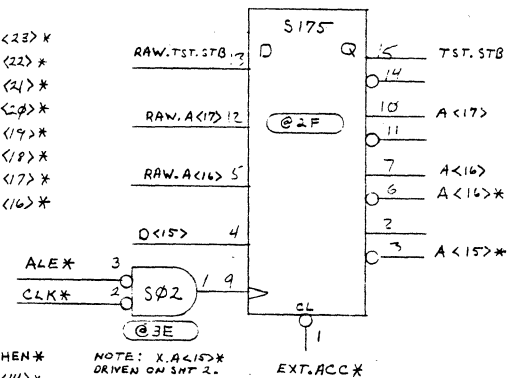
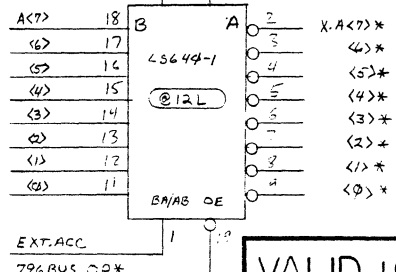
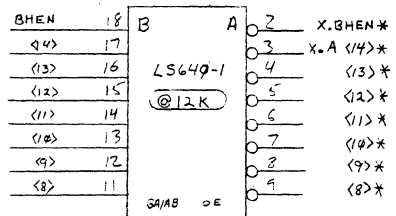
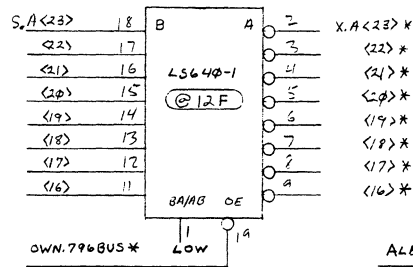
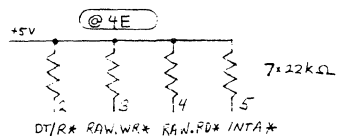
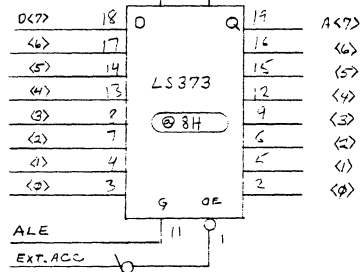
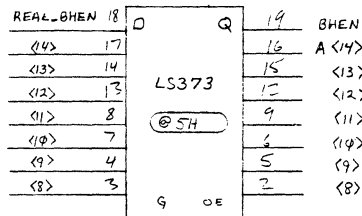
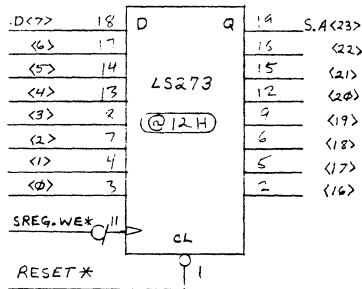
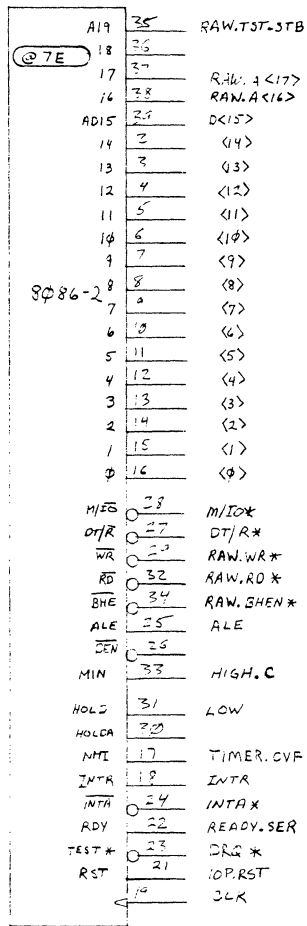
4

3

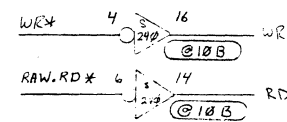
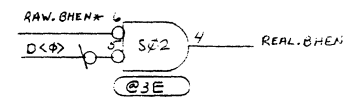
2

1

D
C
B
A

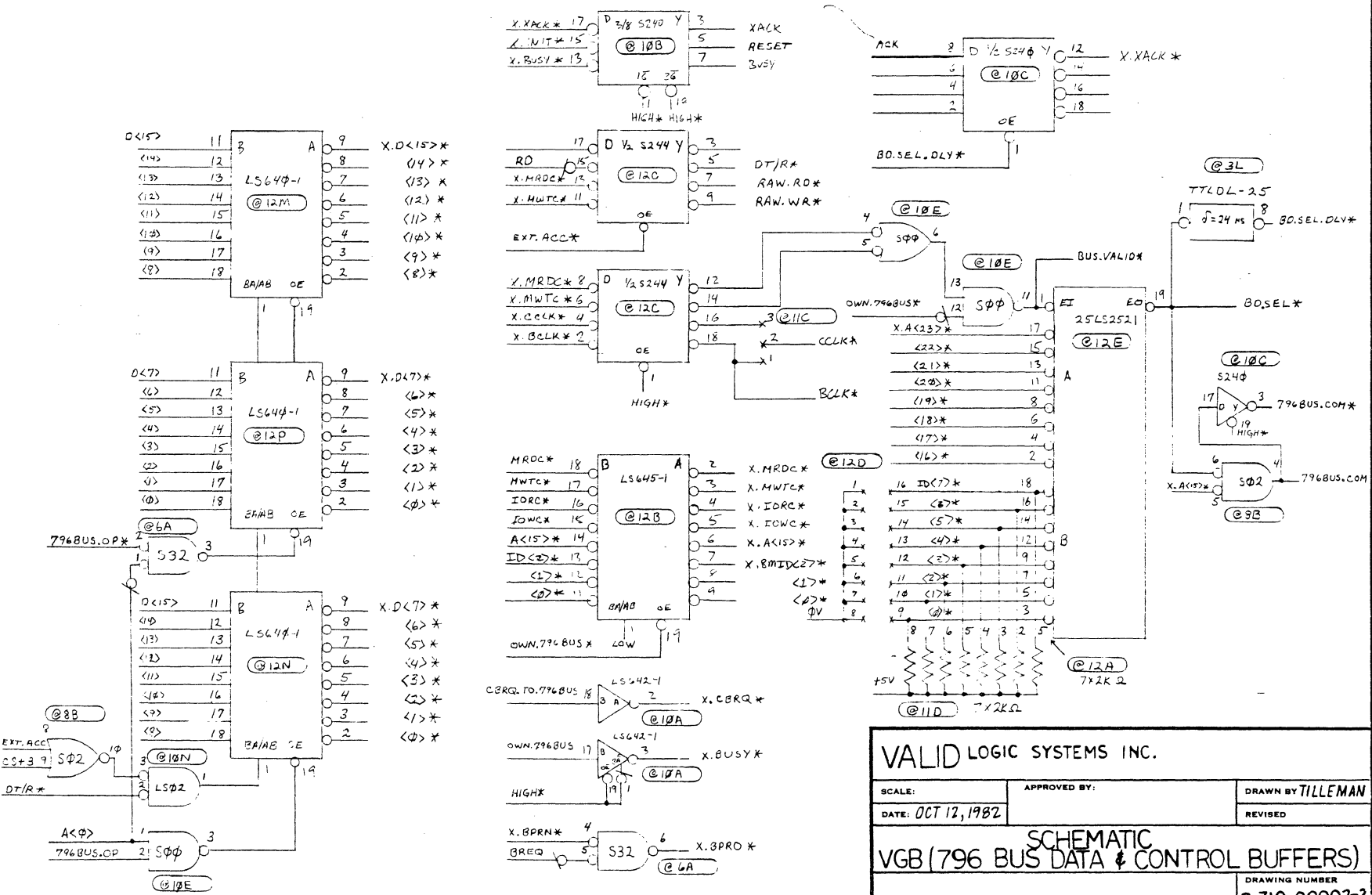


NOTE: X.A<15>* DRIVEN ON SHF 2.



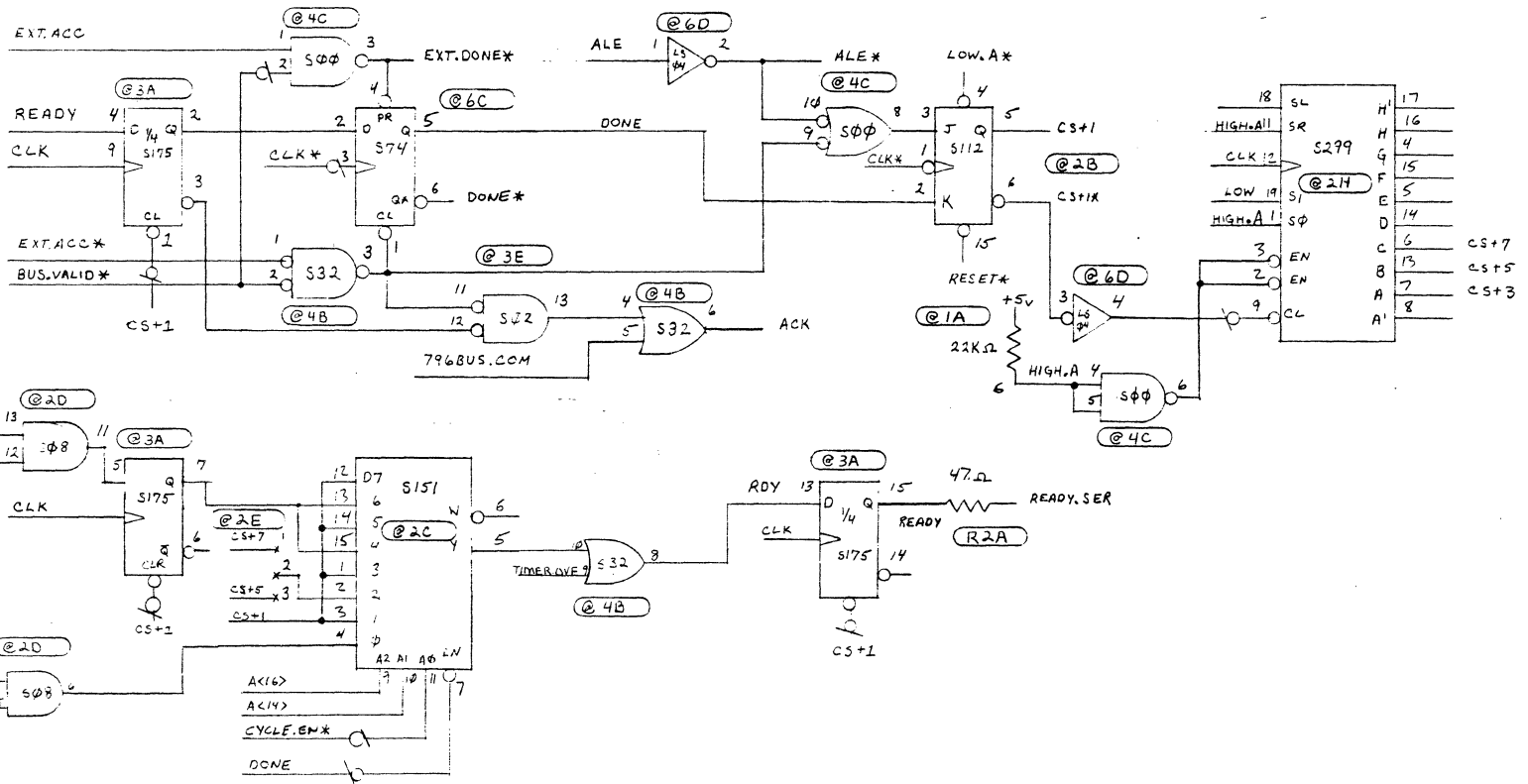
VALID LOGIC SYSTEMS INC.

SCALE:	APPROVED BY:	DRAWN BY TILLEMANN
DATE: OCT 12, 1982		REVISED
SCHEMATIC VGB (8086, ADDRESS BUFFERS)		
DRAWING NUMBER		B-710-C0002-2



VALID LOGIC SYSTEMS INC.

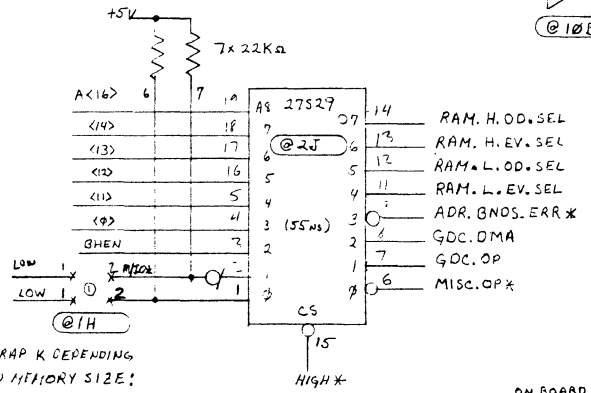
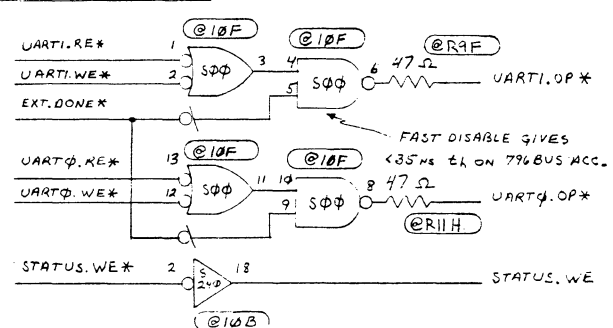
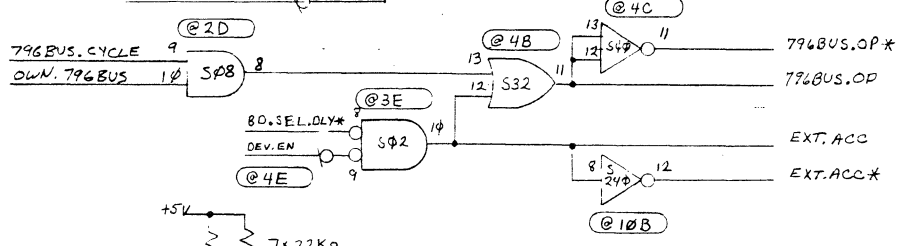
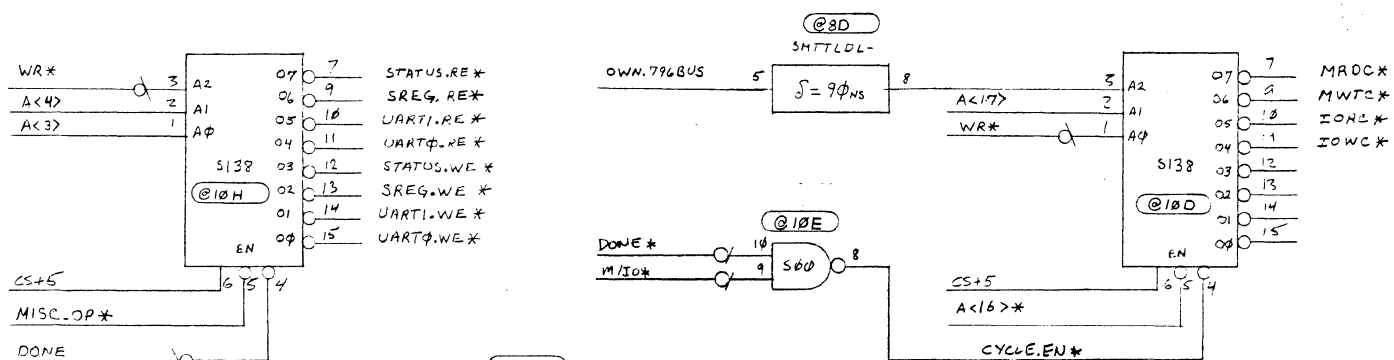
SCALE:	APPROVED BY:	DRAWN BY TILLEMANN
DATE: OCT 12, 1982		REVISED
SCHEMATIC		
VGB (796 BUS DATA & CONTROL BUFFERS)		
DRAWING NUMBER		
B-710-00002-2		



A<17>	A<16>	A<15>	A<14>	A<13>	A<12>	A<11>	A<10>	A<9>	A<8>	A<7>	A<6>	A<5>	A<4>	A<3>	A<2>	A<1>	A<0>	ACCESS TYPE
X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	LOCAL RAM
X	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	LOCAL RAM
X	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	LOCAL DEV
X	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	SDC DMA
0	1	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	796BUS (LDB)
0	1	X	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	796BUS (SD)
1	1	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	796BUS (MEM)
1	1	X	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	796BUS (MEM)

VALID LOGIC SYSTEMS INC.

SCALE:	APPROVED BY:	DRAWN BY TILLEMAN
DATE: OCT 12, 1982		REVISED
SCHEMATIC VGB (LOCAL BUS AND ACK TIMING)		
DRAWING NUMBER		B-710-00002-2



① STRAP K DEPENDING ON MEMORY SIZE:

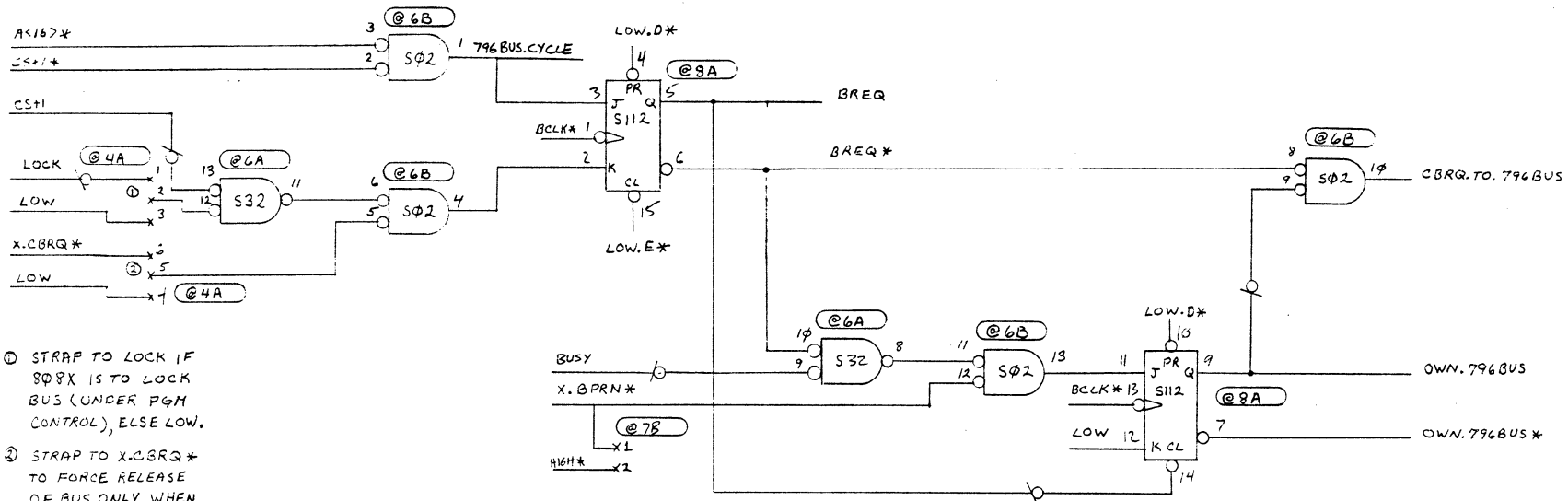
8K BY ⇒ K=0
16K BY ⇒ K=1

NOTE:

A<16>=1 ⇒ 796BUS REF
 =0 ⇒ ON BOARD
 A<14>=0 ⇒ MEM. REF. A<17>=1 ⇒ MEM. REF. } OFF BOARD
 =1 ⇒ 80. REF. =0 ⇒ IO REF.
 A<13>=1 ⇒ GDC OP
 A<13>=0 & A<12>=1 ⇒ GDC.DMA

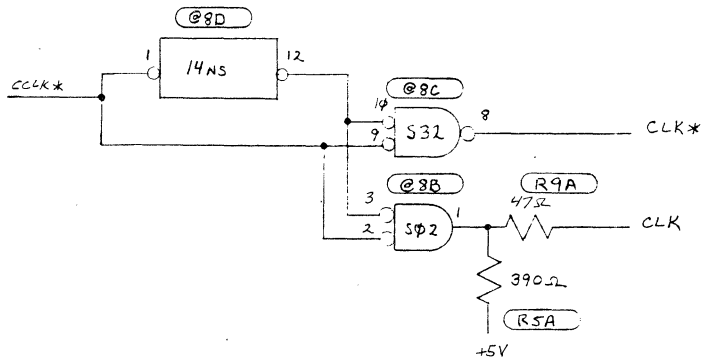
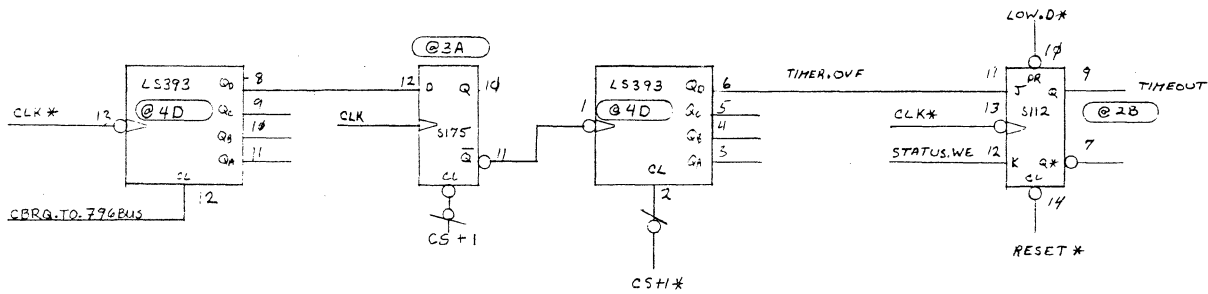
NON-EXISTANT MEMORY

VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY TILLEMANN
DATE: OCT 12, 1982		REVISED
SCHEMATIC VGB (LOCAL/796BUS COMMAND STROBES)		
DRAWING NUMBER B-710-00002-2		



- ① STRAP TO LOCK IF SΦX IS TO LOCK BUS (UNDER PGM CONTROL), ELSE LOW.
- ② STRAP TO X.CBRQ* TO FORCE RELEASE OF BUS ONLY WHEN ANOTHER MASTER IS REQUESTING. LOW FORCES RELEASE AFTER EVERY CYCLE.
- ③ STRAP TO DELAY FOR 5 MHz OPERATION. STRAP TO CS+1 FOR 4 MHz.

VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY TILLEMANN
DATE: OCT 12, 1982		REVISED
SCHEMATIC VGB (796 BUS ARBITRATION LOGIC)		
		DRAWING NUMBER B-710-00002-2



VALID LOGIC SYSTEMS INC.

SCALE:
DATE: OCT 12, 1982

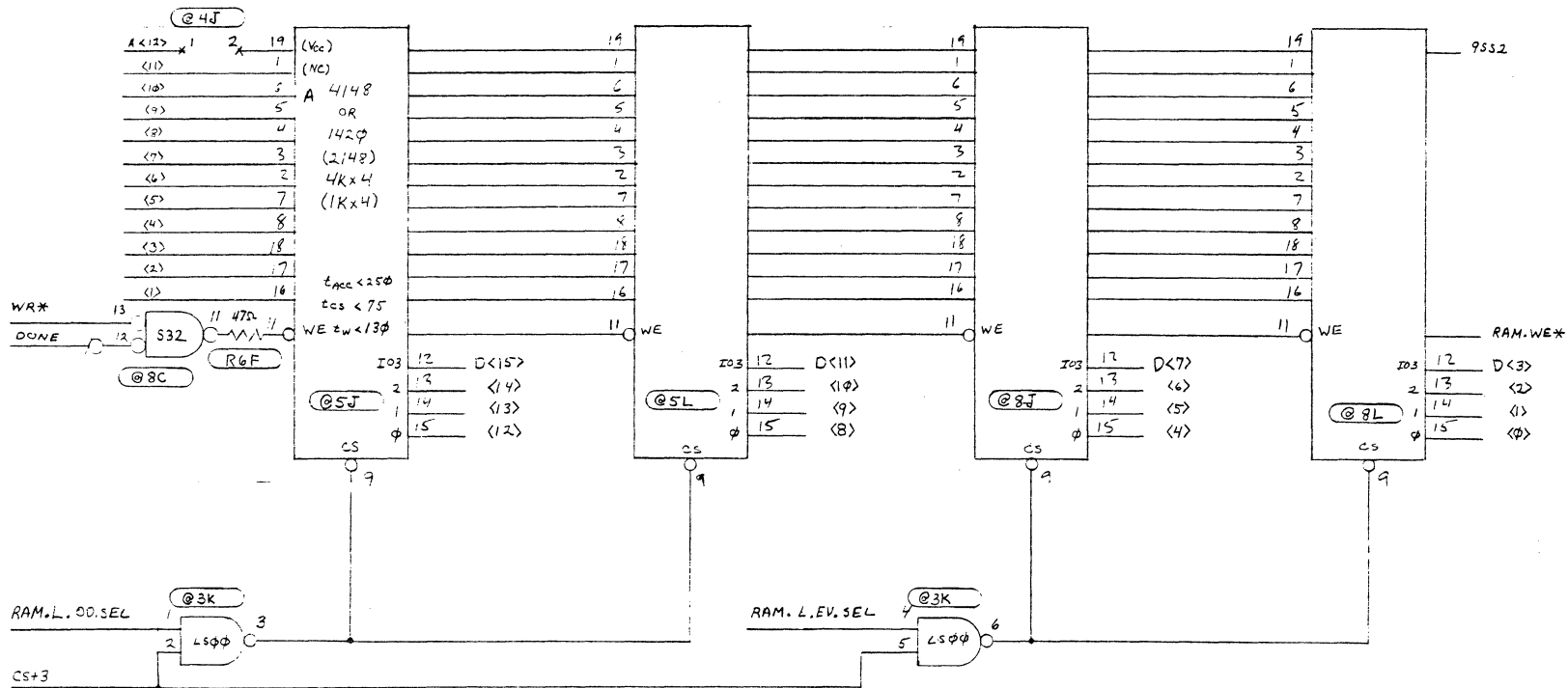
APPROVED BY:

DRAWN BY TILLEMANN

REVISED

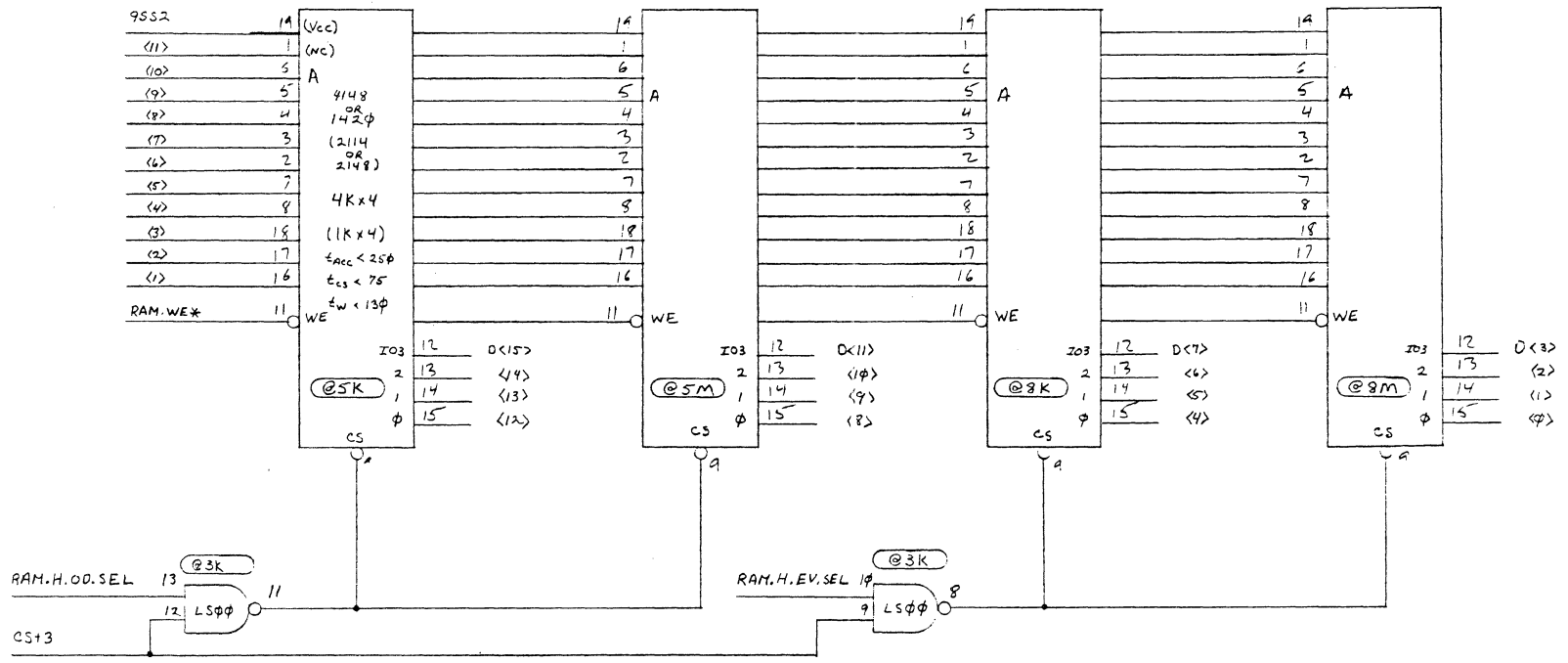
SCHEMATIC
VGB (TIMEOUT AND CLOCK)

DRAWING NUMBER
B-710-00002-2



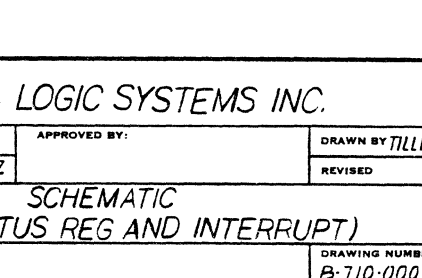
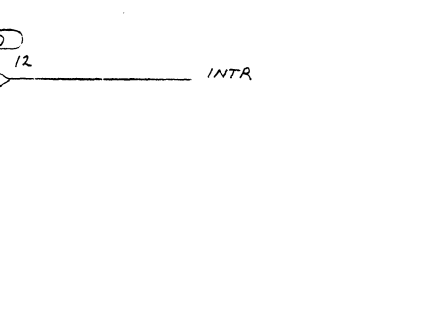
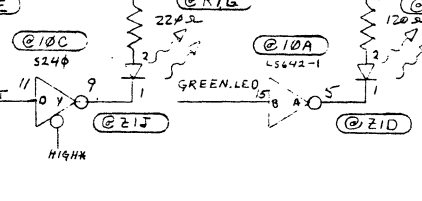
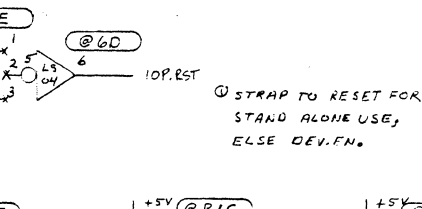
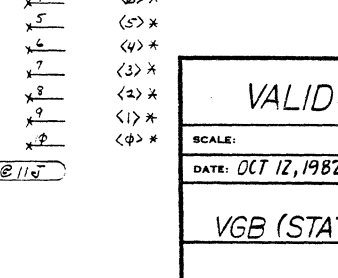
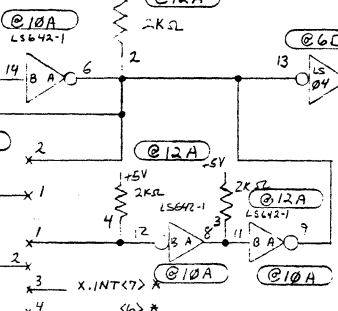
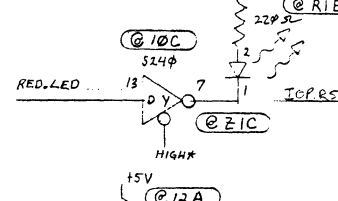
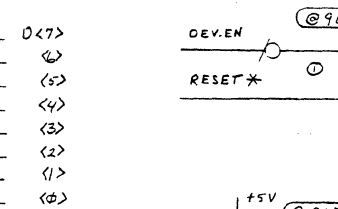
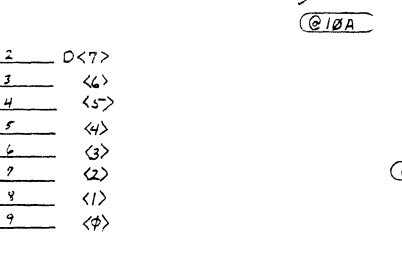
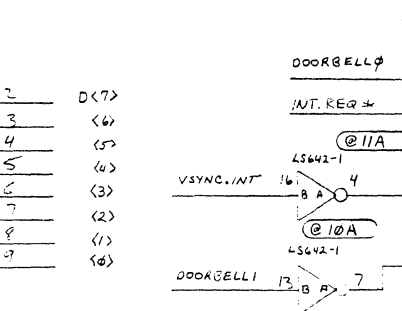
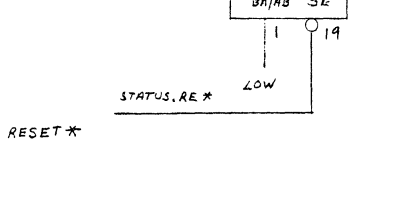
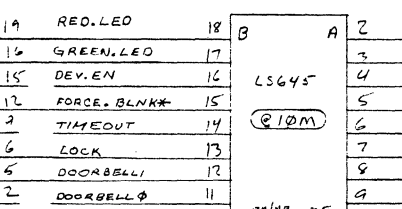
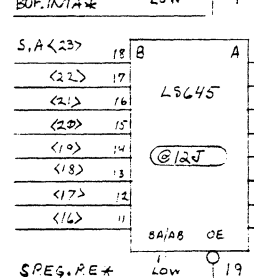
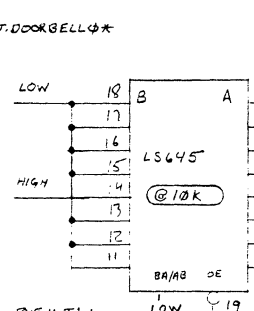
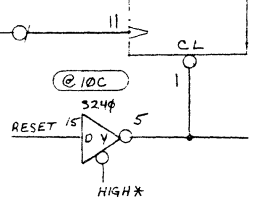
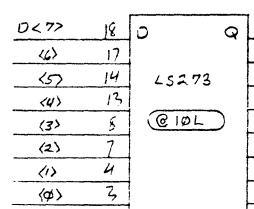
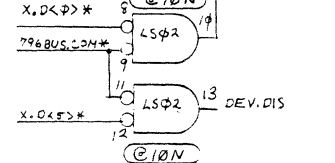
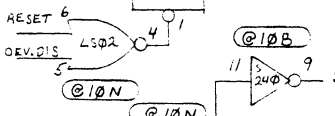
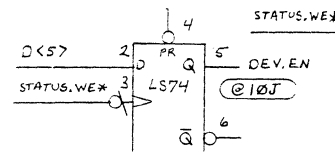
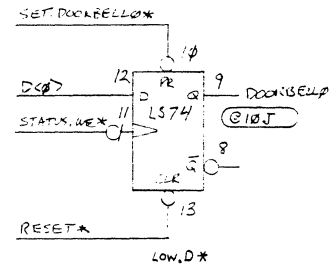
* NOTE: EACH RAM HAS A CUTTABLE TRACE FROM PIN 19 TO PIN 2φ. TRACE IS CUT AND JUMPER INSTALLED FOR 4Kx4 RAMS.

VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY TILLEMANN
DATE: OCT 12, 1982		REVISED
SCHEMATIC VGB (LOCAL MEMORY 1 OF 2)		
		DRAWING NUMBER B-710-00002-2



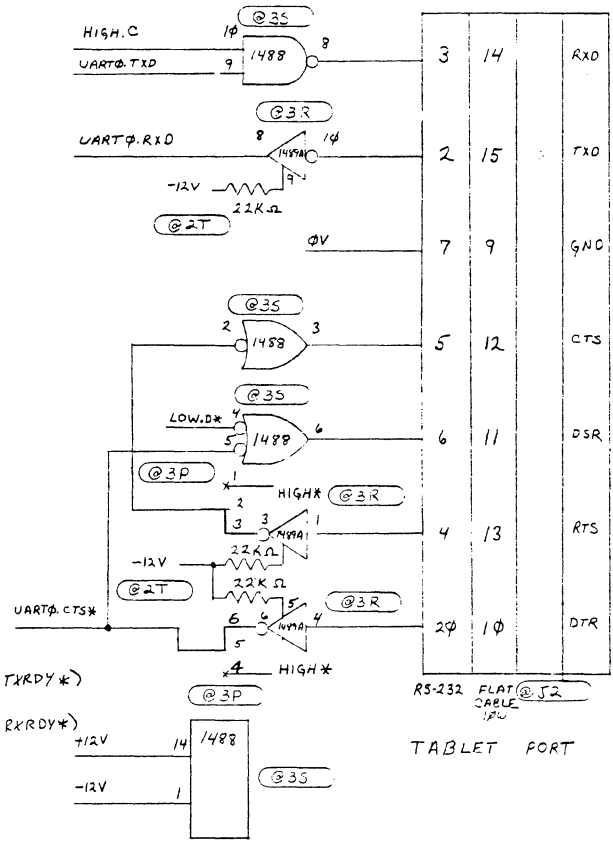
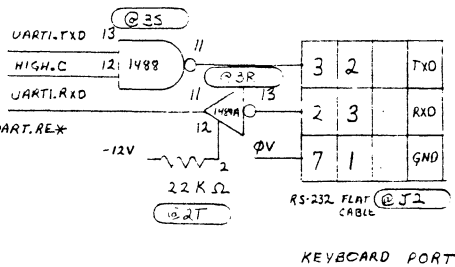
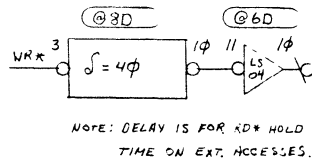
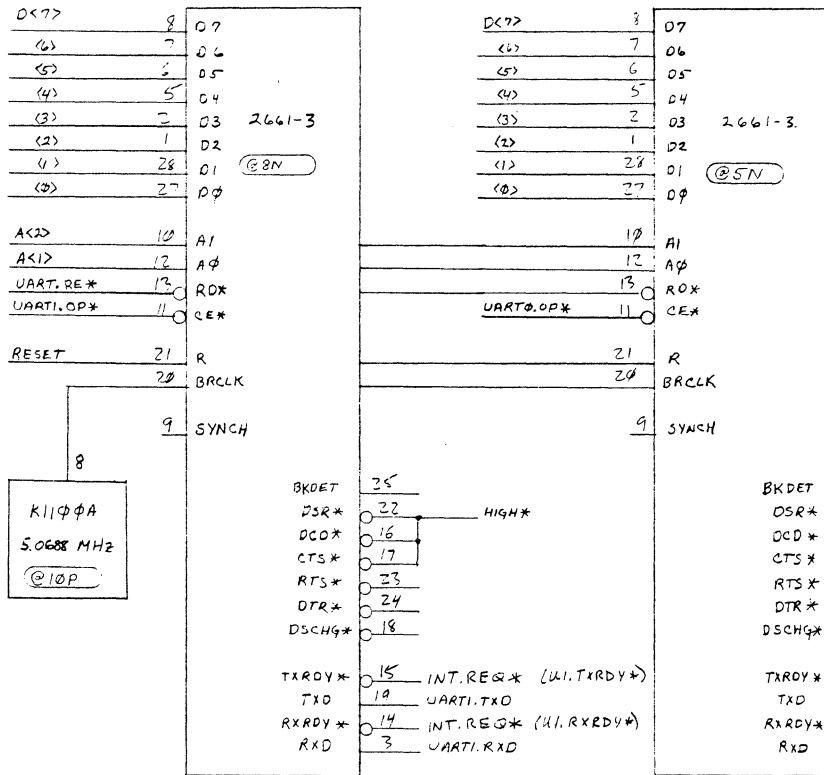
* NOTE: EACH RAM HAS A CUTTABLE TRACE FROM PIN 19 TO PIN 2φ. TRACE IS CUT AND SOLDER INSTALLED FOR 4Kx4 RAMS.

VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY TILLEMAN
DATE: OCT 12, 1982		REVISED
SCHEMATIC VGB (LOCAL MEMORY 2 OF 2)		
DRAWING NUMBER		B-710-00007-2



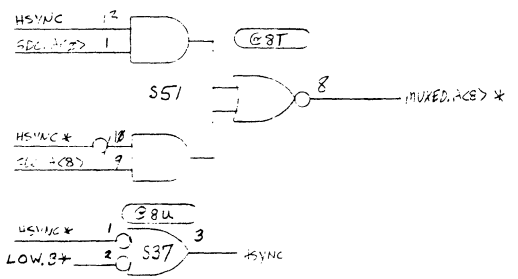
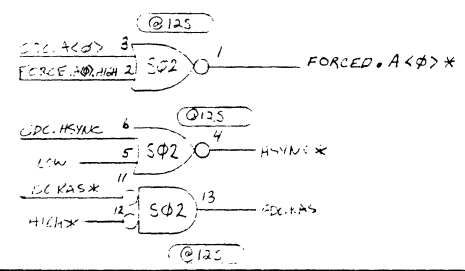
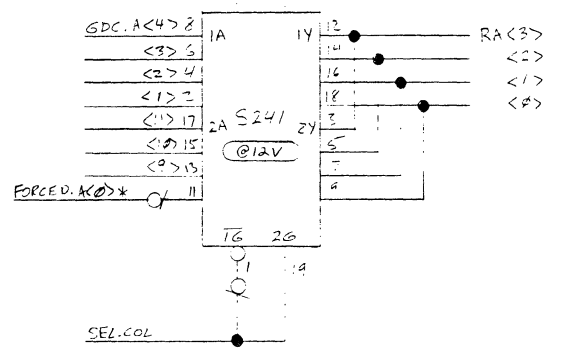
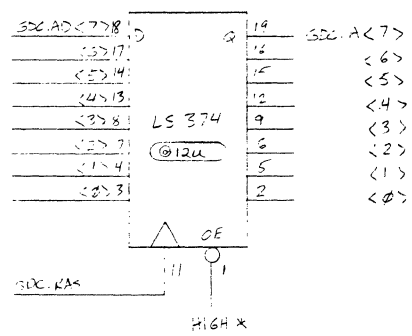
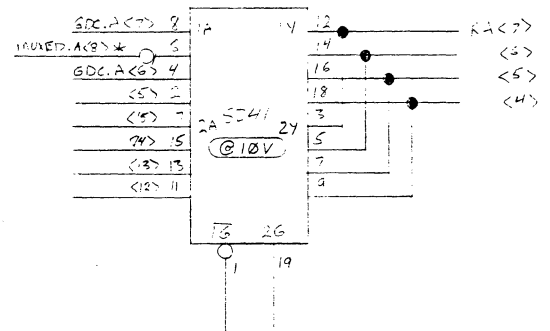
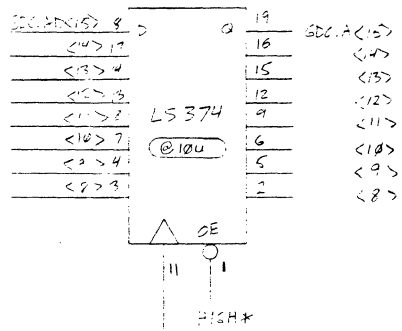
VALID LOGIC SYSTEMS INC.

SCALE:	APPROVED BY:	DRAWN BY TILLEMAN
DATE: OCT 12, 1982		REVISED
SCHEMATIC VGB (STATUS REG AND INTERRUPT)		
		DRAWING NUMBER B-710-00002-2

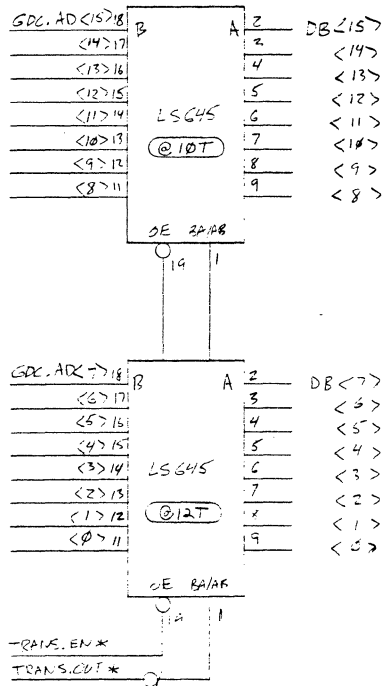


VALID LOGIC SYSTEMS INC.

SCALE:	APPROVED BY:	DRAWN BY TILLEMANN
DATE: OCT 12, 1982		REVISED
SCHMATIC VGB (UARTS)		
		DRAWING NUMBER B-710-00002-2



VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY TILLEMANN
DATE: OCT 12, 1982		REVISED
SCHEMATIC VGB (ADDRESS LATCH AND MUX)		
		DRAWING NUMBER B-710-00002-2

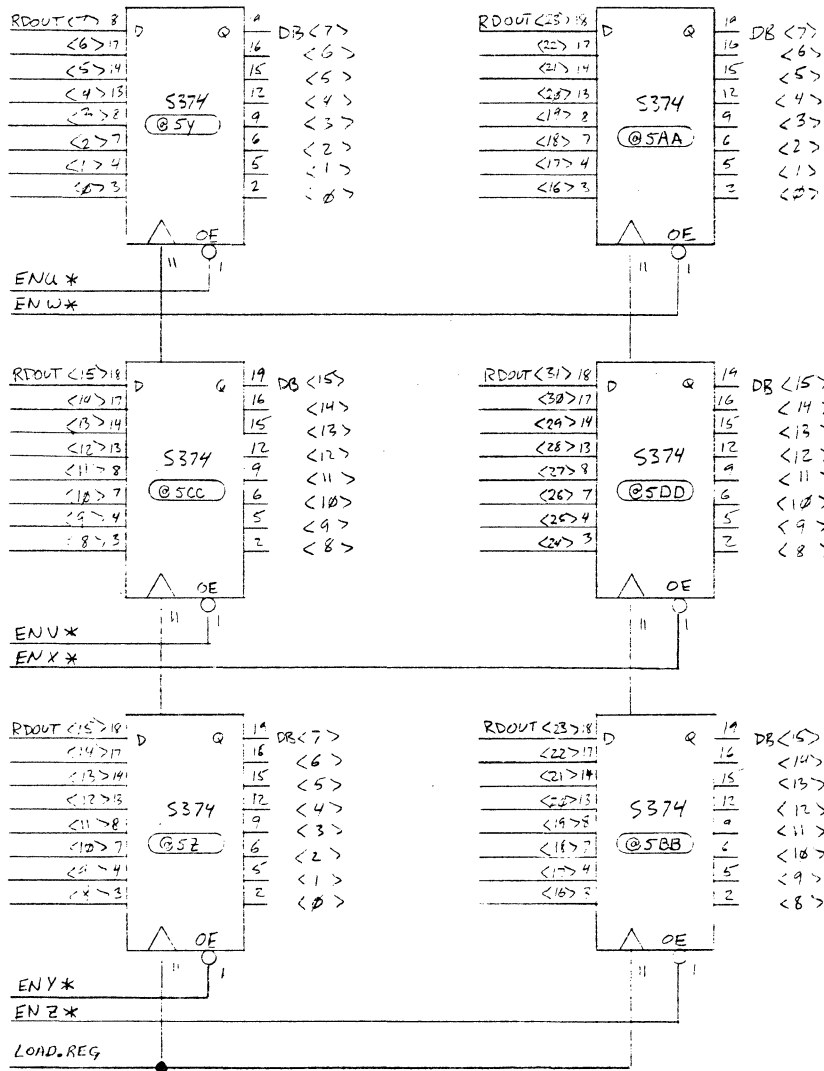


VALID LOGIC SYSTEMS INC.

SCALE:	APPROVED BY:	DRAWN BY TILLEMAN
DATE: OCT 12, 1982		REVISED

SCHMATIC
VGB (DATA TRANSCEIVER)

DRAWING NUMBER B-710-00002-2

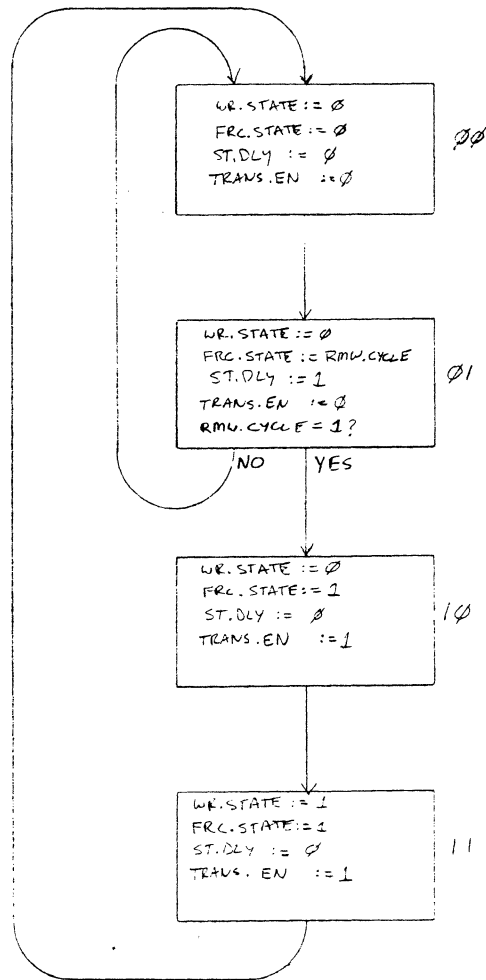


VALID LOGIC SYSTEMS INC.

SCALE:	APPROVED BY:	DRAWN BY TILLEMAN
DATE: OCT 12, 1982		REVISED

SCHMATIC
VGB (DATA REGISTERS)

DRAWING NUMBER B-710-00002-2



NOTE: ALL ASSIGNMENTS OCCUR
ON ENTRY TO STATE.

NOTE: GDX.RAS LOW FORCES FSM
INTO STATE 00

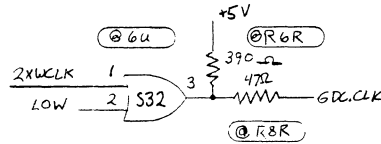
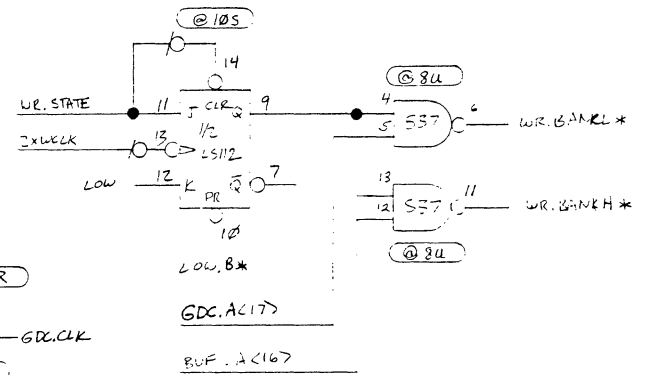
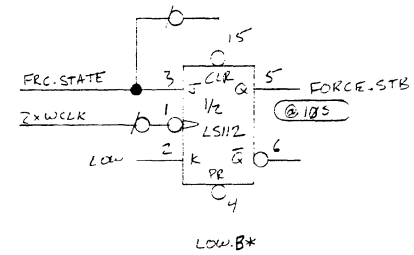
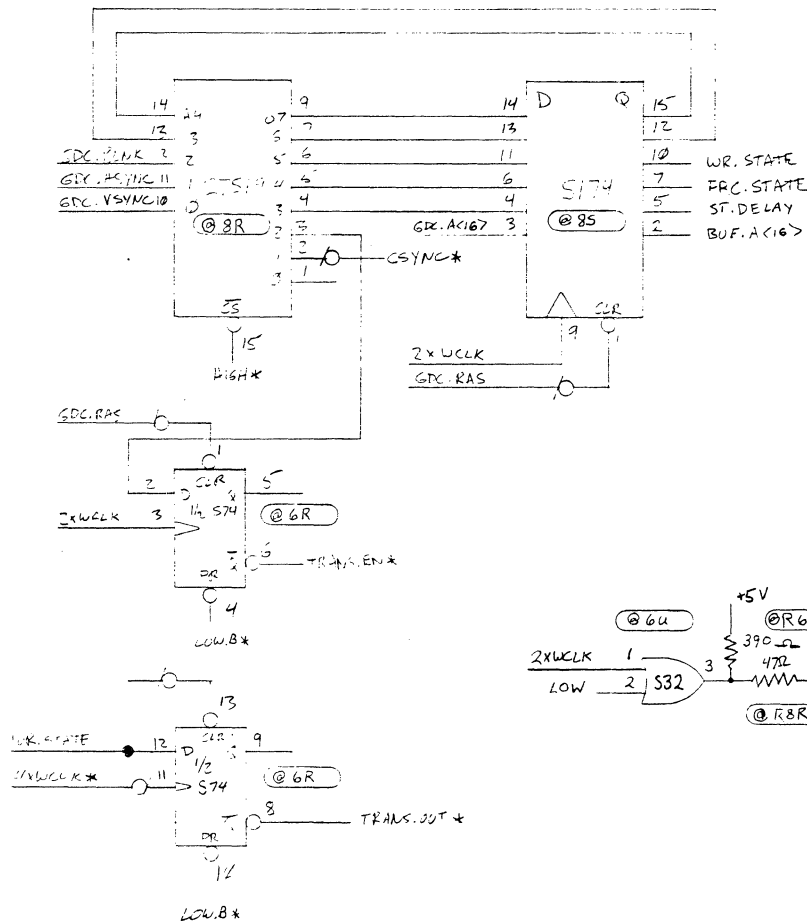
NOTE: RMW.CYCLE := BANK A - HSYNC

VALID LOGIC SYSTEMS INC.

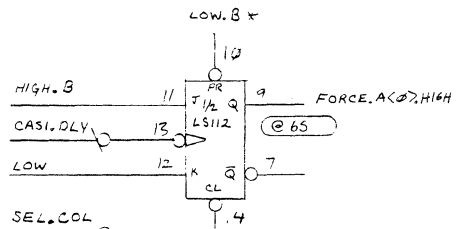
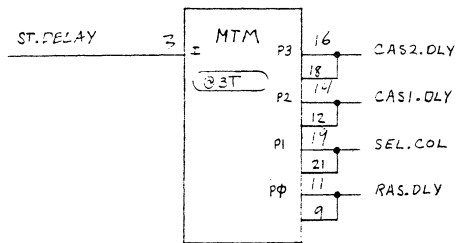
SCALE:	APPROVED BY:	DRAWN BY TILLEMANN
DATE: OCT 12, 1982		REVISED

SCHMATIC
VGB (MEMORY CNTRL FSM STATE DIAGRAM)

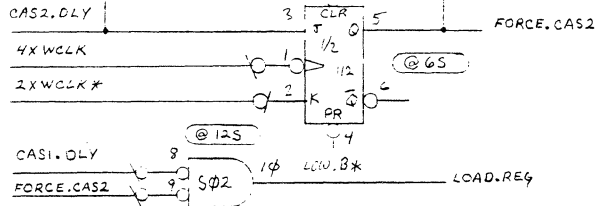
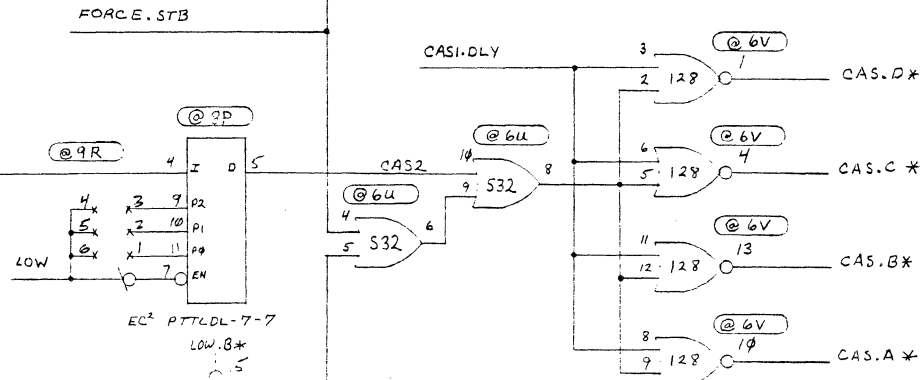
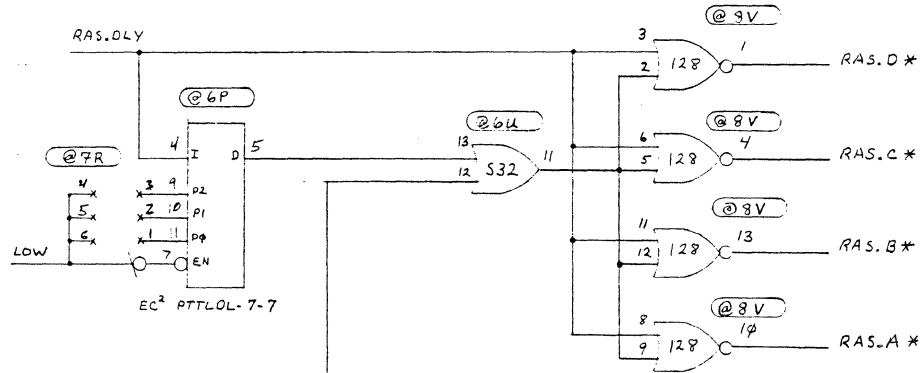
DRAWING NUMBER B-710-00002-2



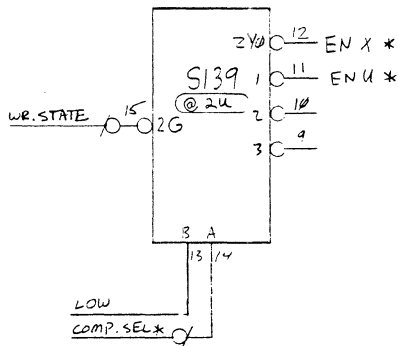
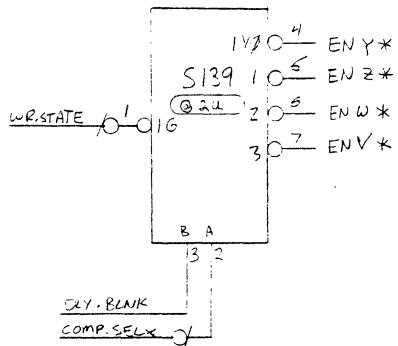
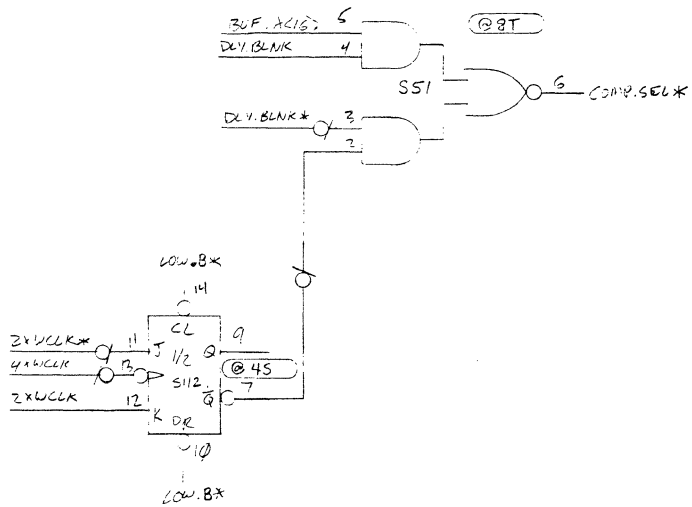
VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY TILLEMANN
DATE: OCT 12, 1982		REVISED
SCHEMATIC VGB (MEMORY CONTROL FSM)		
		DRAWING NUMBER B-710-00002-2



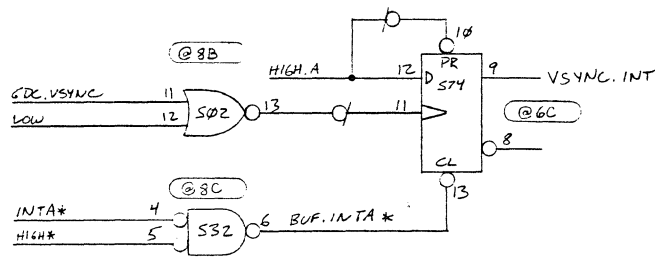
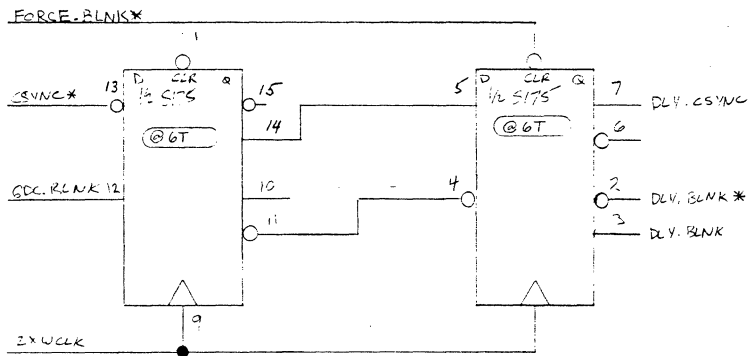
NOTE: SEL.COL NACTIVE CLEARS
FORCE.A ϕ ,HIGH AT START
OF EACH MEMORY CYCLE



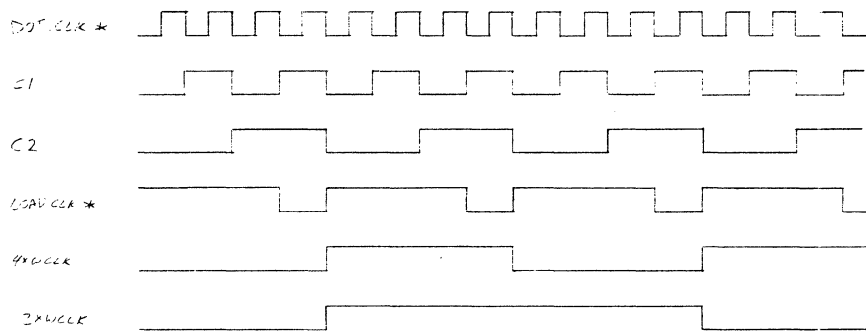
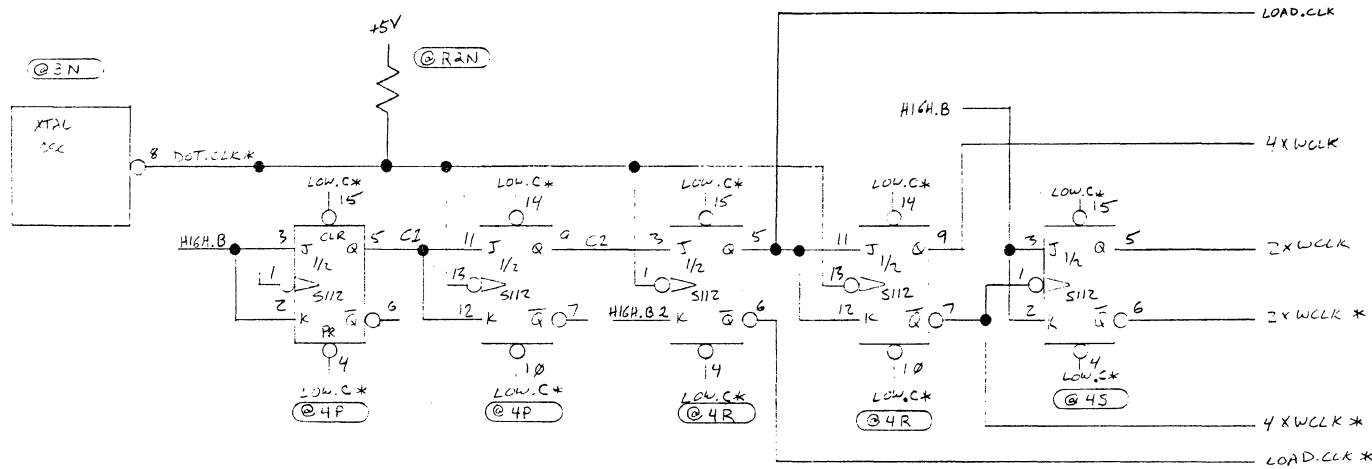
VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY TILLEMANN
DATE: DEC 12, 1982		REVISED
SCHEMATIC VGB (MEMORY CONTROL STROBES)		
DRAWING NUMBER		B-710-00002-2



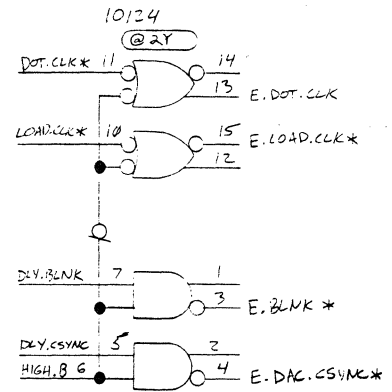
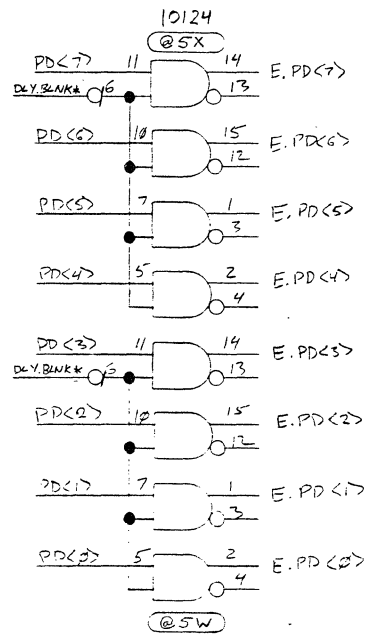
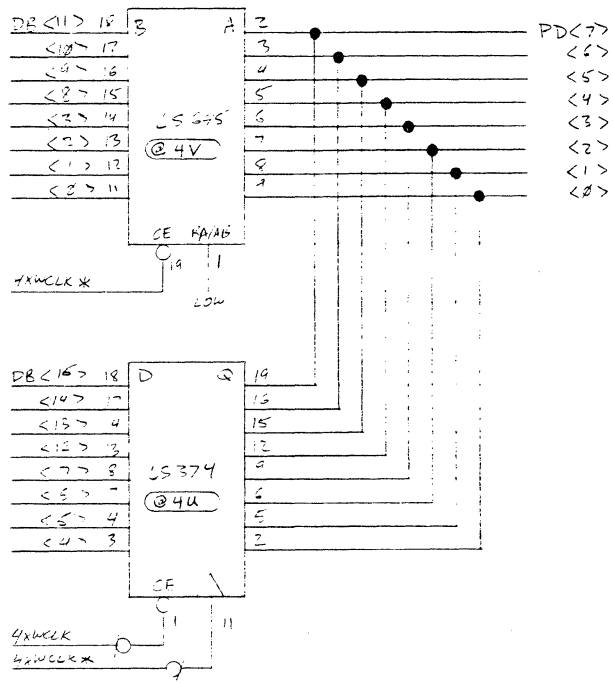
VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY TILLEMAN
DATE: OCT 12, 1982		REVISED
SCHEMATIC VGB (DATA REGISTER CONTROL)		
		DRAWING NUMBER B-710-00002-2



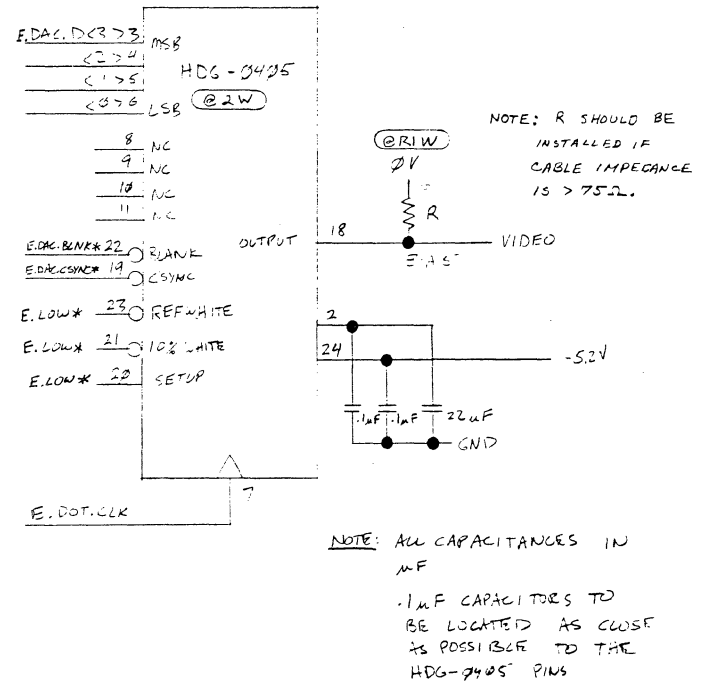
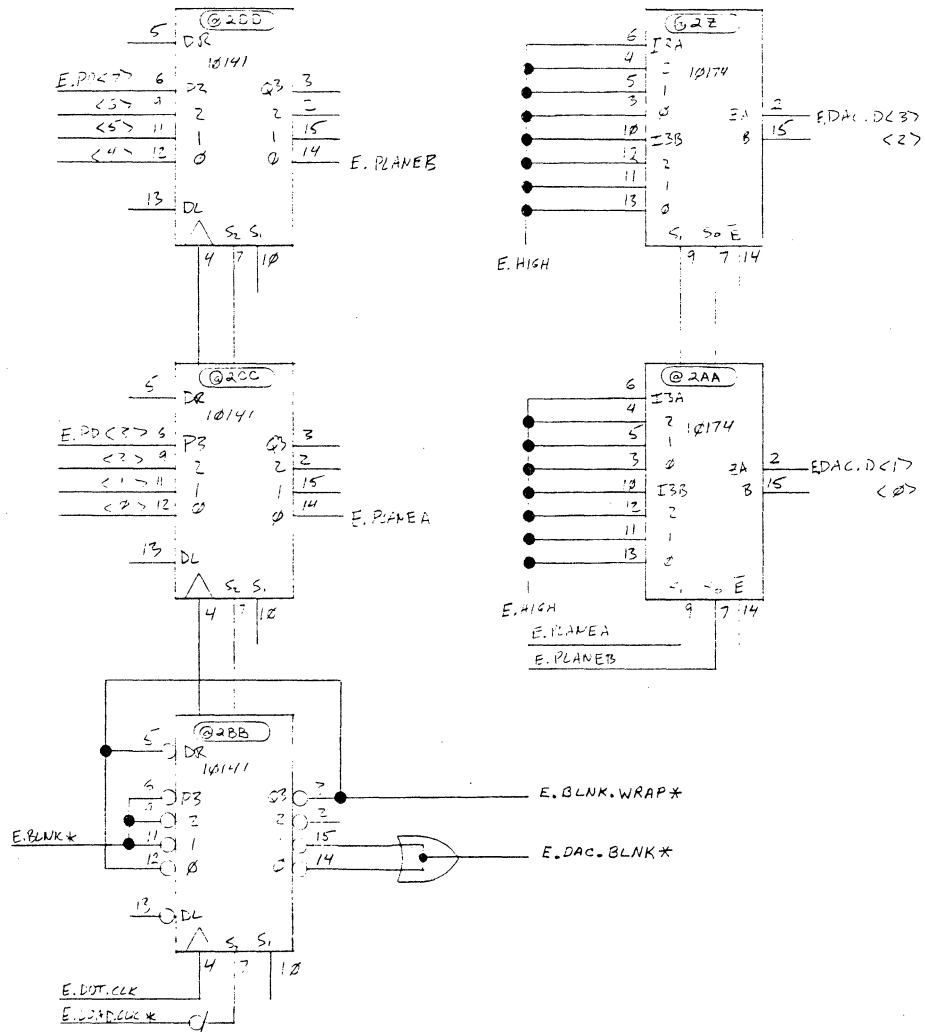
VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY TILLEMANN
DATE: OCT 12, 1982		REVISED
SCHEMATIC		
VGB (VIDEO CONTROL SIGNAL SYNC)		
		DRAWING NUMBER
		B-710-00002-2



VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY TILLEMAN
DATE: OCT 12, 1982		REVISED
SCHEMATIC VGB (CLOCK GENERATION)		
		DRAWING NUMBER B-710-00002-2

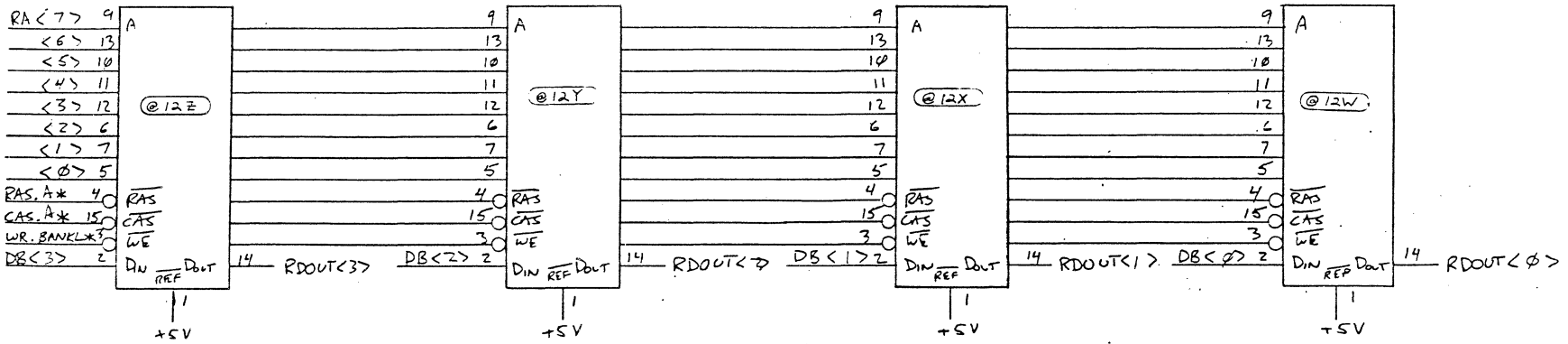
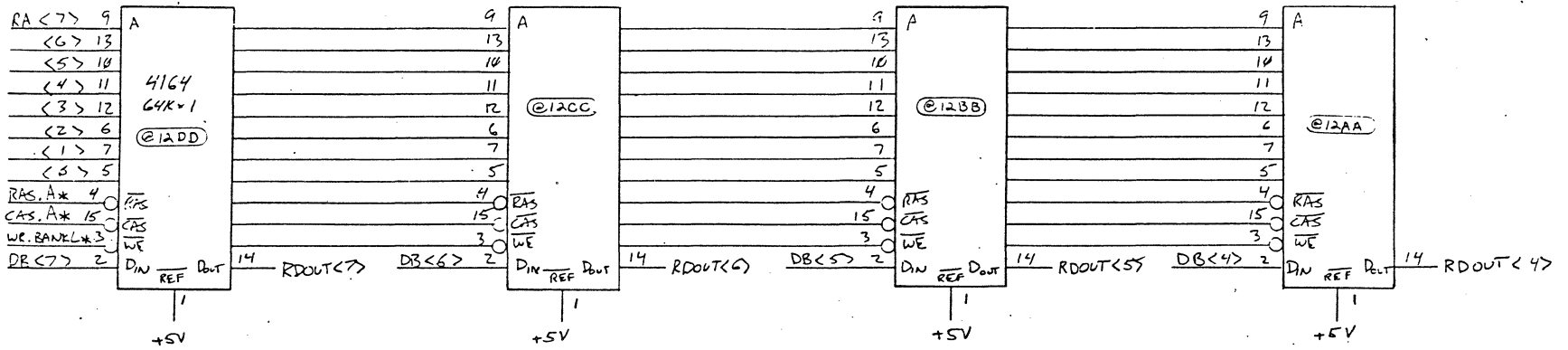


VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY TILLEMANN
DATE: OCT 12, 1982		REVISED
-SCHEMATIC VGB (DATA MUX AND ECL CONVERSION)		
		DRAWING NUMBER B-710-00002-2



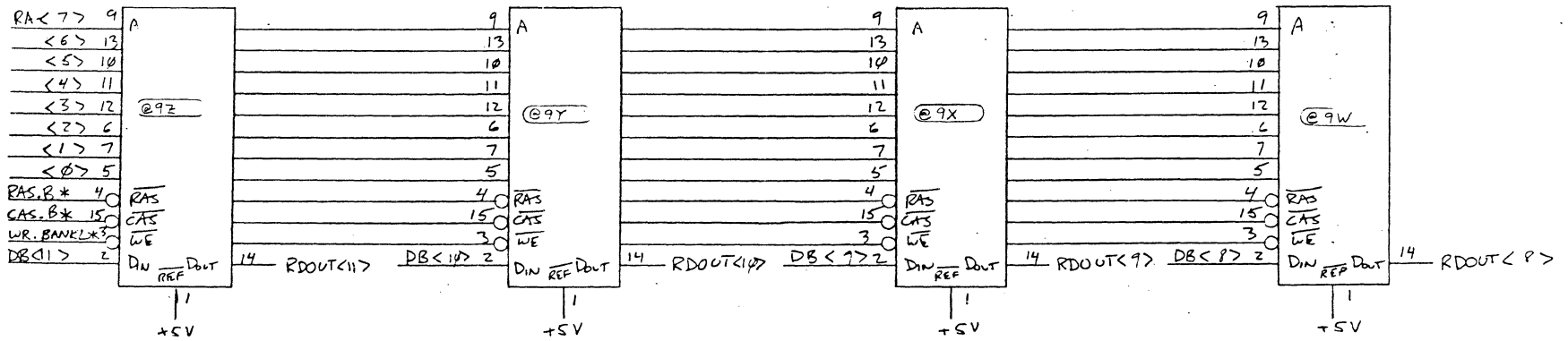
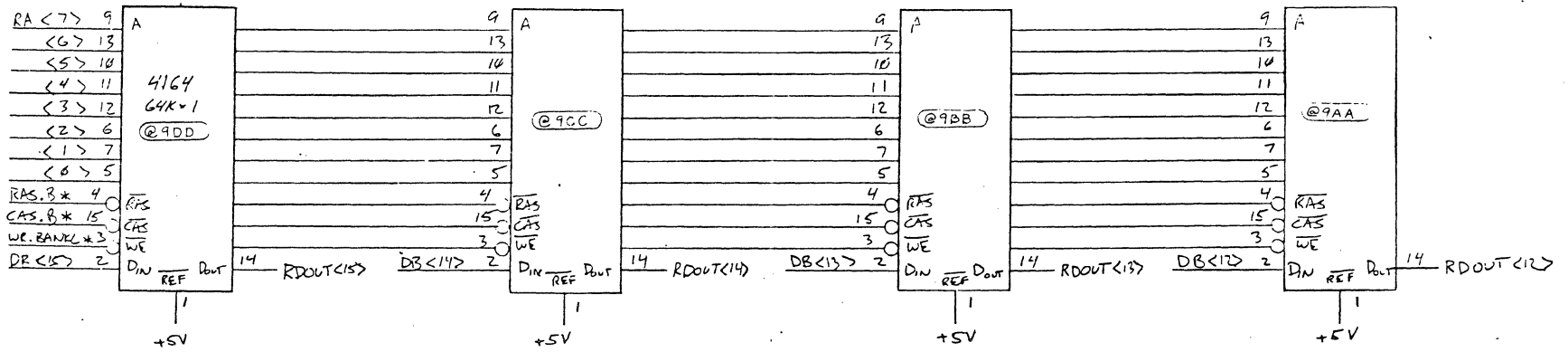
VALID LOGIC SYSTEMS INC.

SCALE:	APPROVED BY:	DRAWN BY TILLEMANN
DATE: OCT 12, 1982		REVISED
SCHEMATIC VGB (VIDEO GENERATION)		
		DRAWING NUMBER B-710-00002-2



VALID LOGIC SYSTEMS INC.

SCALE:	APPROVED BY:	DRAWN BY TILLEMANN
DATE: OCT 17, 1982		REVISED
SCHEMATIC VGB (FRAME MEMORY 1 OF 4)		
		DRAWING NUMBER B-710-00002-2



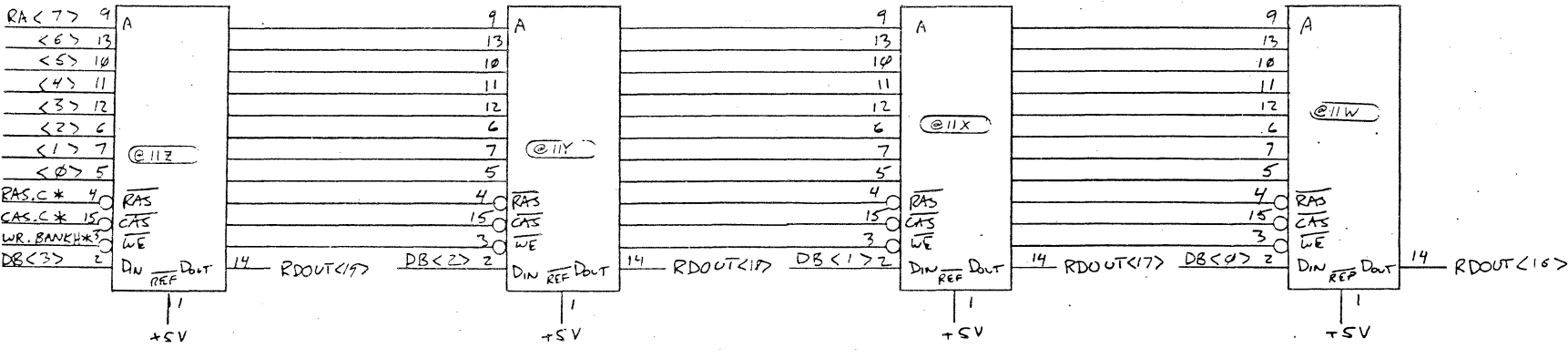
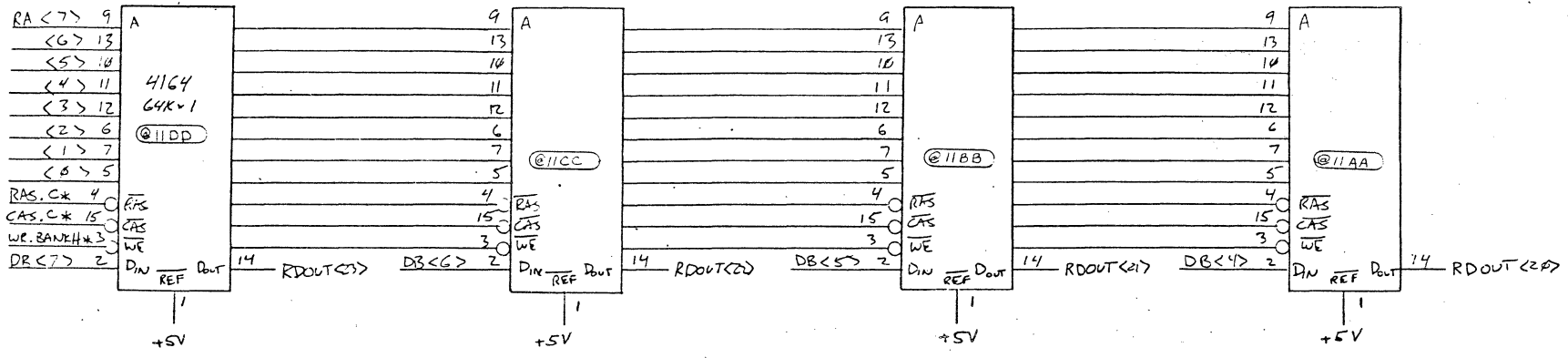
VALID LOGIC SYSTEMS INC.

SCALE:	APPROVED BY:	DRAWN BY TILLEMANN
DATE: OCT 12, 1982		REVISED

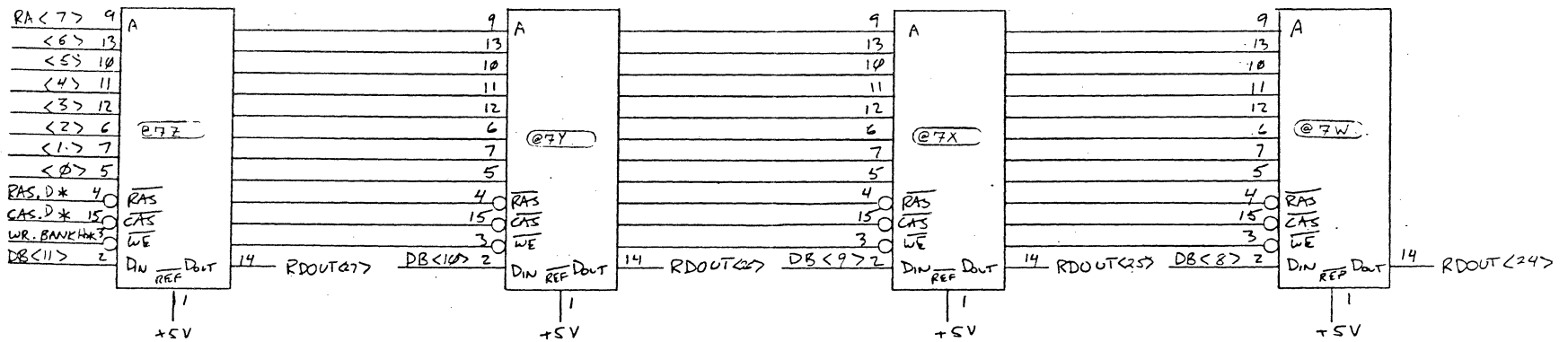
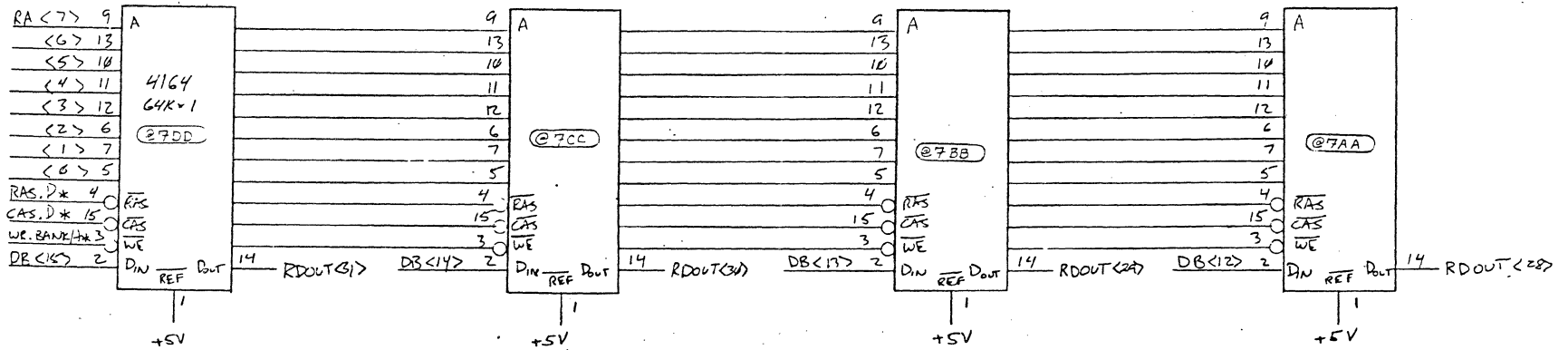
SCHMATIC
VGB (FRAME MEMORY 2 OF 4)

DRAWING NUMBER
3-710-00002-2

SHEET 25 OF 33



VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY TILLEMANN
DATE: OCT 12, 1982		REVISED
SCHEMATIC VGB (FRAME MEMORY 3 OF 4)		
		DRAWING NUMBER B-710-00002-2

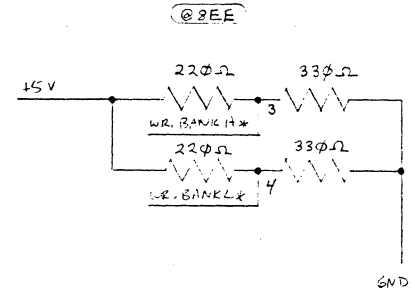
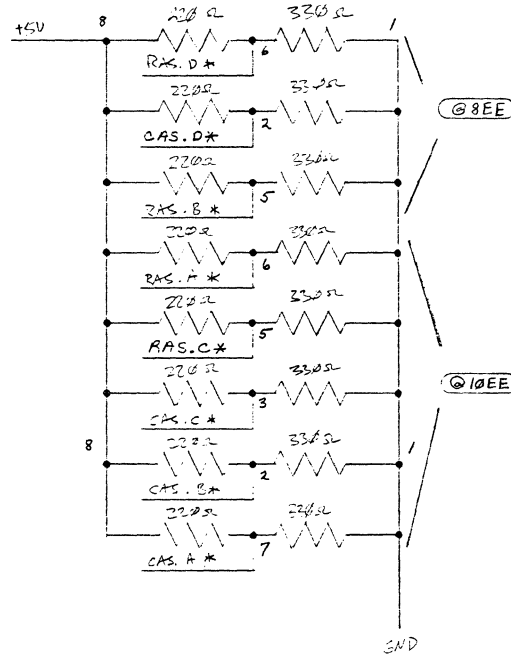
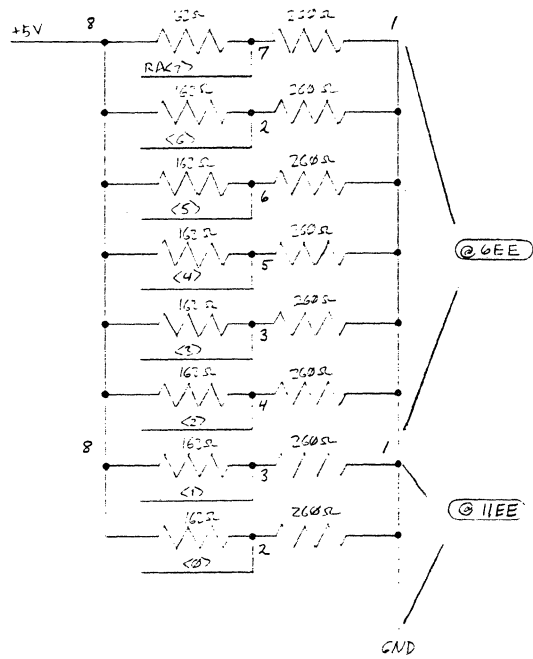


VALID LOGIC SYSTEMS INC.

SCALE:	APPROVED BY:	DRAWN BY TILLEMANN
DATE: OCT 17, 1982		REVISED

SCHMATIC
VGB (FRAME MEMORY 4 OF 4)

DRAWING NUMBER
B-710-00002-2

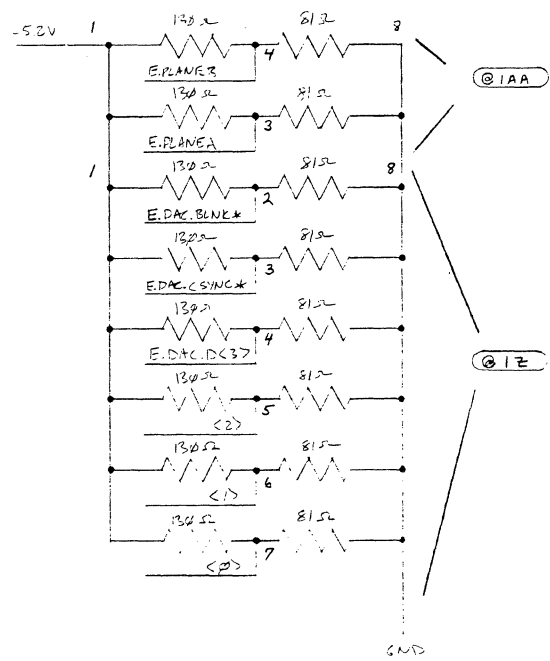
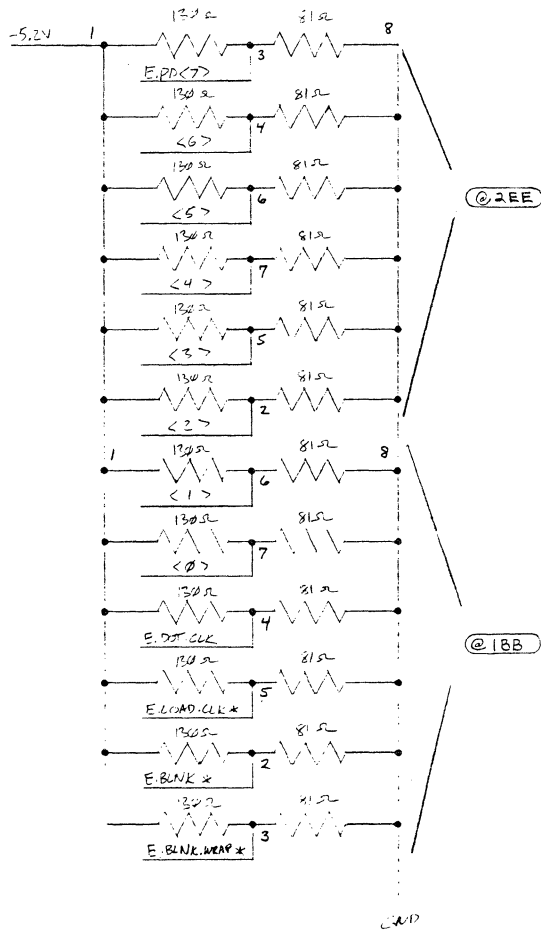


VALID LOGIC SYSTEMS INC.

SCALE:	APPROVED BY:	DRAWN BY TILLEMANN
DATE: OCT 12, 1982		REVISED

SCHMATIC
VGB (MEMORY SIGNAL TERMINATORS)

DRAWING NUMBER
B-710-00002-2



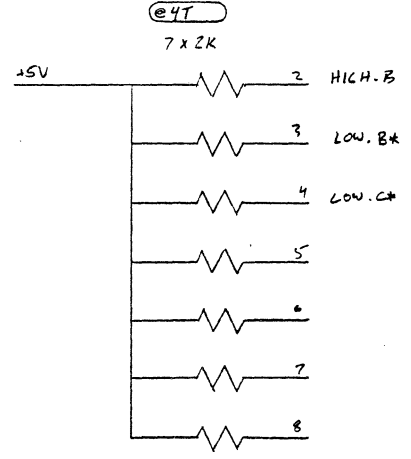
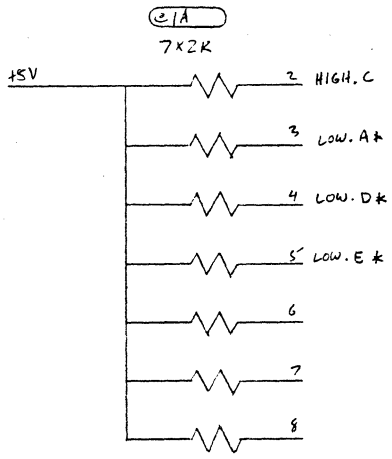
VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY TILLEMANN
DATE: OCT 12, 1982		REVISED
SCHEMATIC VGB (ECL TERMINATORS)		
		DRAWING NUMBER 8-710-00002-2

4

3

2

1



REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		

D

C

B

A

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± MXX ±	CONTRACT NO.		SCHEMATIC VGB (PULLUPS)		
	APPROVALS	DATE			
	MATERIAL	CHECKED	SIZE B	CODE IDENT NO. SEE SHEET 1	DRAWING NO. 710-00002-2
	FINISH		SCALE	SHEET 30 OF 33	
DO NOT SCALE DRAWING					

4

3

2

1

4

3

2

1

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		

D

C

B

A

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES \pm .XX \pm \pm \pm .XXX \pm	CONTRACT NO.		APPROVALS DRAWN TILLEMANN 10-17-87		DATE 10-17-87		SCHEMATIC VGB (BLANK)		
	MATERIAL	CHECKED							
	FINISH			SIZE B	CODE IDENT NO. SEE SHEET 1	DRAWING NO. 710-00002-2			
	DO NOT SCALE DRAWING			SCALE					SHEET 31 OF 33

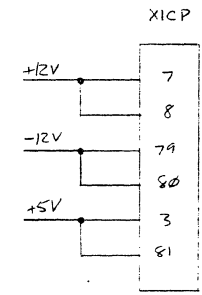
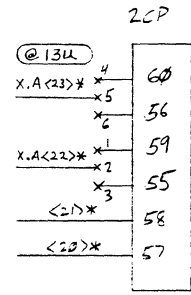
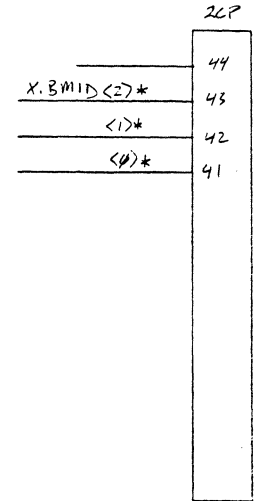
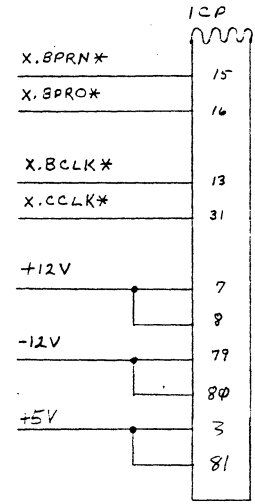
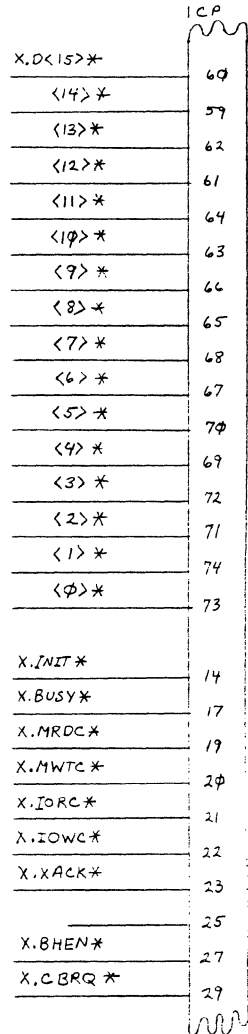
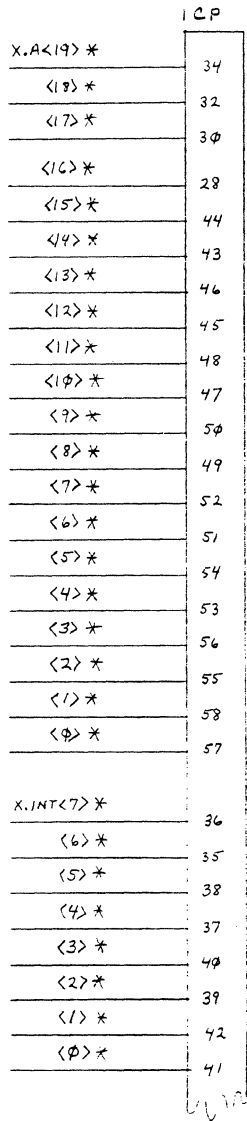


4

3

2

1

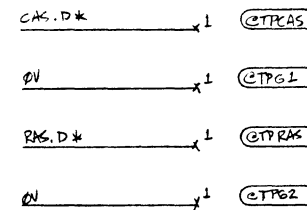


VALID LOGIC SYSTEMS INC.

SCALE:	APPROVED BY:	DRAWN BY TILLEMANN
DATE: OCT 12, 1982		REVISED
SCHEMATIC		
VGB (796BUS CONNECTOR)		
		DRAWING NUMBER
		B-710-00002-2

②J1 16W
FLAT
CABLE

.CLK *	46	1
BCLK *	42	3
CS+1	38	5
WR *	36	7
OWN. 796BUS	34	9
INTR	32	11
RESET *	30	13
TST. STB	28	15
TIMER. OVF	26	16
RDY	24	14
EXT. ACC	22	12
A <17>	20	10
A <16>	18	8
A <14>	16	6
ΦV	14	4
ΦV	12	2



VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY TILLEMAN
DATE: OCT 12, 1982		REVISED
SCHEMATIC VGB (TEST CONNECTOR)		
		DRAWING NUMBER B-710-00002-2

Peripheral Interface Board User Manual

VED-032782-1 Revision 5-28-82 (LCW)

02 Aug 84

Copyright 1984

Valid Logic Systems Incorporated

This document contains confidential proprietary information which is not to be disclosed to unauthorized persons without the written consent of an officer of Valid Logic Systems Incorporated.

The copyright notice appearing above is included to provide statutory protection in the event of unauthorized or unintentional public disclosure.

1.0 INTRODUCTION

The Valid Logic Systems Peripheral Interface Board (PIB) is a Multibus-compatible board for use as the peripheral controller for the SCALDsystem. The PIB supports:

- o A high resolution electrostatic printer/plotter. The PIB provides a high speed, parallel, standard Versatec interface capable of data rates to 200Kbytes per second. This interface is configurable for short- or long-line driving.
- o Two high-speed, long distance, RS-449 serial host computer communication links capable of supporting SDLC, HLDC, BISYNC and ASYNC protocols at transfer rates to 500Kbaud.
- o A high-speed parallel interface. The PIB provides a bidirectional 16-bit parallel interface for use as a communication link to a local host computer. This interface supports transfer rates to 200Kbytes per second.

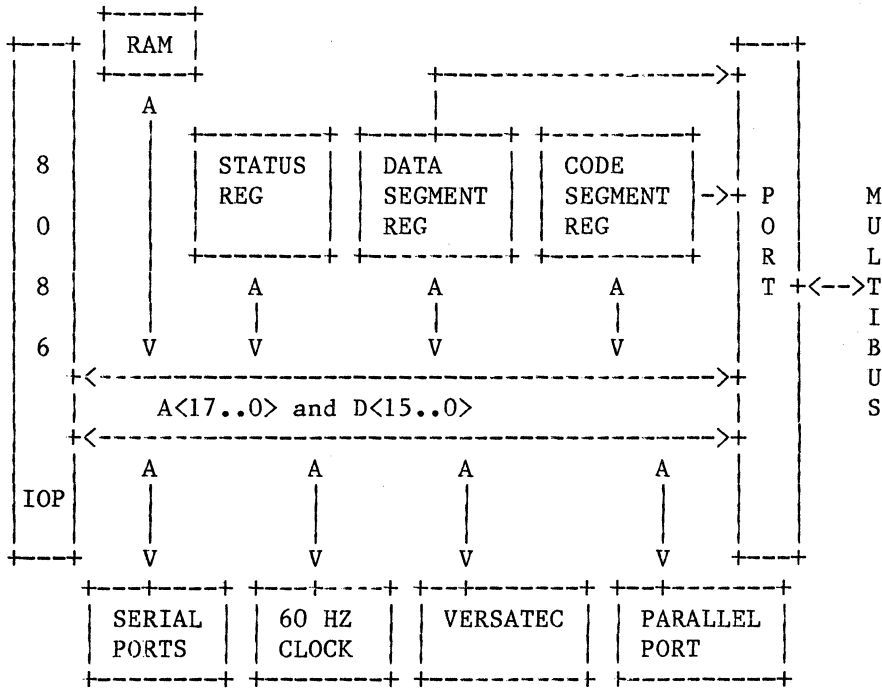
The PIB offers the following features:

- o Full interface to Versatec printers and plotters
- o High speed, configurable, RS-449 interface ports
- o 16-bit parallel interface port
- o On-board 8MHz 8086 control (input/output) processor
- o 4KBy of local RAM for the IOP, upgradeable to 16KBy
- o Full interrupt capabilities on all peripherals
- o On-board status, error and memory-relocation registers
- o Full interface to 16MBy IEEE-796 bus (Intel Multibus)
- o Maintenance port for testing and diagnosis

2.0 PERIPHERAL INTERFACE BOARD ARCHITECTURE

The architecture of the PIB is designed to allow the board to operate as autonomously as possible by shifting a great deal of the load associated with peripheral control away from the CPU and the Multibus. This high degree of local processing is effected by incorporating a high-performance, on-board input/output microprocessor or "IOP" with on-board local memory and an interface to the Multibus. The IOP controls a local bus attached to the nine logical devices shown in the following figure:

Peripheral Interface Board User Manual



All devices are memory mapped with the following address assignments (in IOP address space):

- 00000 ... 03FFF : on-board RAM
- 04000 : Versatec data register
- 04008 : Versatec control register
- 04010 : Versatec status register
- 04018 : board status register
- 04020 : parallel control register
- 04028 : parallel status register
- 04030 : parallel data register
- 04038 : baud rate selection register
- 04040 : serial port A control register
- 04048 : serial port A status register
- 04050 : serial port B control register
- 04058 : serial port B status register
- 06060 : USART data A
- 06062 : USART data B
- 06064 : USART control A
- 06066 : USART control B
- 04068 : code segment register
- 04070 : data segment register
- 04078 : 60Hz clock interrupt clear address
- 10000 ... 1FFFF : Multibus IO access (data segment)
- 30000 ... 3FFFF : Multibus memory access (data segment)
- 50000 ... 5FFFF : Multibus IO access (code segment)
- 70000 ... 7FFFF : Multibus memory access (code segment)

During Multibus access, a full 24-bit Multibus address is formed by concatenating the contents of either the code segment register or the data segment register with A<15..0>.

The high-order Multibus address bits (A<23..20>) are driven on connector P2. Two different standards exist for the positions of A<22> and A<23> on P2. These bits are therefore strappable to conform to either, or both standards. The positions of the non-standard Multibus signals are as follows:

Pin(s)	Signal
56, 60	A<23>
55, 59	A<22>
58	A<21>
57	A<20>

The Multibus is acquired and released using precisely the same bus arbitration logic that is used by the Valid CPU board. This arbitration logic has two strappable options. One option allows locking of the the bus, and the other selects between relinquishing the bus after each bus cycle or surrendering the bus when it is requested by another bus master. When the IOP is reset, the PIB functions as a Multibus slave.

Devices on the PIB can be accessed by the CPU using Multibus memory reads and writes. For this purpose, the PIB occupies a 64K-byte window of Multibus memory address space. This window can be located on any 64K-byte boundary using on-board switches. Within this window, the CPU address space is the same as the IOP, except that A<17..15> of the on-board address bus are forced to zero during CPU accesses. Should the IOP generate a Multibus reference to its own 64K-byte window, a timeout will occur.

Devices on the PIB may not be accessed from the Multibus while the IOP is running. However, a memory write operation to board address 008000 with D<5> set, clears the DEV.EN bit of the on-board status register whether or not the IOP is running. Clearing this bit, in turn, forces the IOP into its reset state and the board into slave mode.

To facilitate CPU/IOP communication, "door bell" interrupts are provided. There are two door bell bits in the status register. When DOORBELLO is set, an interrupt is sent to the IOP. When DOORBELL1 is set, an interrupt is sent to the CPU at a strappable priority level. In addition, straps are provided so that the IOP can sense one Multibus interrupt. A memory write operation to board address 008000 with D<1> set, sets the DOORBELLO bit of the on-board status register (see section 2.2) whether or not the IOP is running.

2.1 The IOP

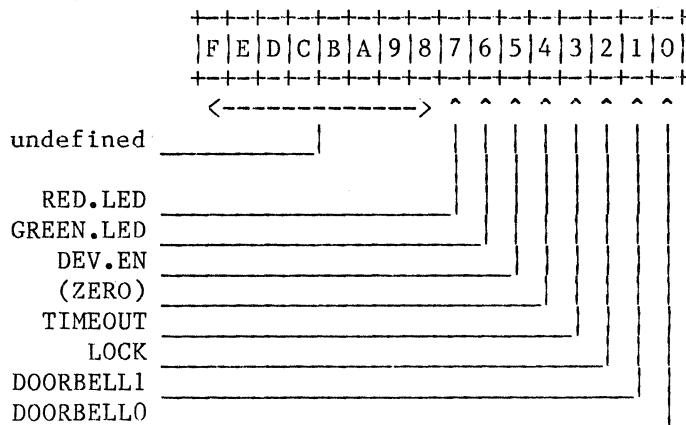
The IOP is a general-purpose microcomputer; its role on the board is determined by its programming. (Further information on the 8086 used as the IOP can be found in Intel publications The 8086 Family User's Manual and the Component Data Catalog.) The IOP typically performs the following tasks:

- o Transferring plotting data to the Versatec from an image stored in Multibus memory.
- o Handling host communication protocols and data transfers (e.g., generating and checking error checksums, requesting retransmission of garbled messages, etc.).
- o Communicating with the CPU including buffering messages to and from the host computer and providing status information about peripheral devices.
- o Running local diagnostics.

Local on-board RAM contains most of the code and constant data required by the IOP. Large, infrequently-accessed code and data may be stored in Multibus memory. Keeping frequently-accessed resources on-board eliminates most IOP wait states and removes much of the PIB memory access traffic from the Multibus.

2.2 The Status Register

An on-board, low-byte only status register contains board state information, some of which may be written. The register has eight fields (unless otherwise noted a field is read/write):



- o RED.LED lights the on-board red diagnostic LED if, and only if, it is set.
- o GREEN.LED lights the on-board green diagnostic LED if, and only if, it is set.
- o DEV.EN is normally strapped to the reset pin of the IOP. If clear, the IOP is forced to the reset state. A second on-board red LED lights if, and only if, DEV.EN is clear. DEV.EN is not only read/write, but may be "asynchronously" cleared from the Multibus as described in section 2.0.
- o TIMEOUT is set if no response to a Multibus cycle is received by the IOP within 16 microseconds after the bus is acquired or a reference is made to a non-existent local device. A timeout causes a non-maskable interrupt to be sent to the IOP. This interrupt stacks some status information and executes the appropriate interrupt routine. Note: If the timeout occurs on an intra-segment jump instruction, the stacked IP is wrong. If the timeout occurs on an inter-segment jump, both the stacked IP and CS are wrong. TIMEOUT is a clear only bit. Writing the status register clears TIMEOUT.
- o Zero bit. This bit always reads as a zero to allow a device probing program to distinguish between the PIB and other Valid boards having a status register located at the same on board address.
- o If LOCK is set, the bus is not released after the next Multibus access; if LOCK is clear, the bus is released to any requestor regardless of priority.
- o If DOORBELL1 is set, an interrupt request is made to the host at a strappable priority level.
- o If DOORBELL0 is set, an interrupt request is made to the IOP. DOORBELL0 is not only read/write, but may be asynchronously set from the Multibus.

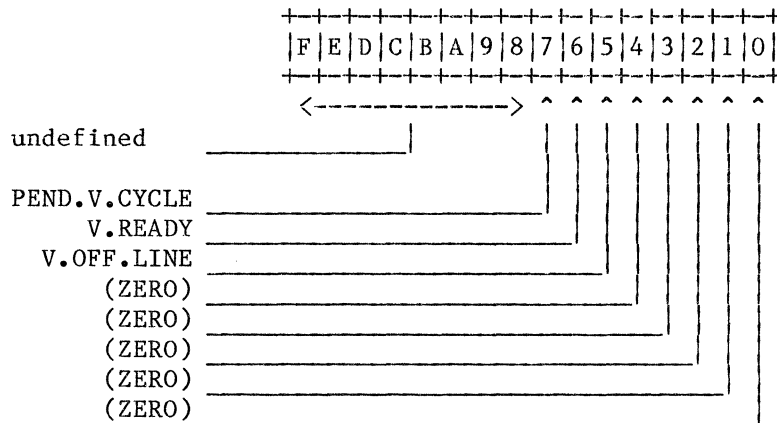
The status register is cleared on reset.

2.3 The Versatec Interface

The PIB Versatec interface provides for connection to a Versatec printer or plotter via the standard parallel interface. Printing or plotting data is transferred to the device eight bits at a time, and control operations are sent via a special control register. The interface consists of three low-byte only registers.

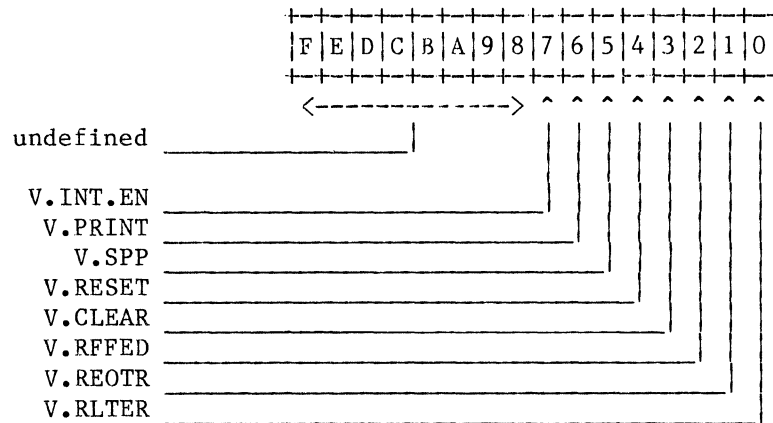
2.3.1 Versatec Data Register - Writes to this register send eight bits of data to the device to be either printed or plotted. Data should be written to this register only when the Versatec has indicated that it is ready to receive new data.

2.3.2 Versatec Status Register - This register contains information regarding the state of the Versatec. The bits are allocated as follows:



- o PEND.V.CYCLE is set ("1") whenever the Versatec has requested new data since the last byte was sent. This bit is read/write and is automatically cleared when data is written to the Versatec data register. Data should be sent to the device only when this bit is a "1."
- o V.READY contains the value of the Versatec READY signal. This signal becomes a "1" when the Versatec first requests new data, and becomes a "0" some time after the data is received. This bit is useful primarily for diagnostic purposes and should not be used to determine whether the device is ready for more data. This bit is read only.
- o V.OFF.LINE is a "1" when the device is off line for any reason. This bit is read only.
- o This bit always is "0."
- o This bit always is "0."
- o This bit always is "0."
- o This bit always is "0."
- o This bit always is "0."

2.3.3 Versatec Control Register - This register configures the Versatec device and generates remote commands. The bits are allocated as follows:



- o V.INT.EN allows PEND.V.CYCLE to generate an IOP interrupt request whenever the device is ready for data.
- o V.PRINT sets Versatec printer/plotters into print mode when "1" and into plot mode when "0."
- o V.SPP, when "1," enables simultaneous print/plot mode for printer/plotters that support this option.
- o V.RESET generates a Versatec RESET command when written to a "1."
- o V.CLEAR generates a Versatec CLEAR command when written to a "1."
- o V.RFFED generates a Versatec FORM-FEED command when written to a "1."
- o V.REOTR generates a Versatec END-OF-TRANSMISSION command when written to a "1."
- o V.RLTER generates a Versatec LINE-TERMINATE command when written to a "1."

Only one of the commands shown above should be issued at once, and PEND.V.CYCLE always should be sampled before sending a command just as with data transfers. PEND.V.CYCLE always should be written to a "0" before a command is given since it is not cleared automatically on control register writes as it is on data register writes. This register is read/write and is cleared on system reset.

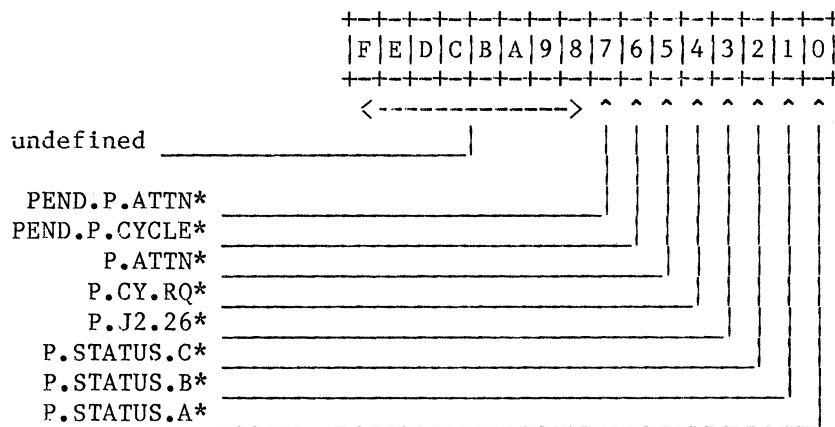
2.4 The Parallel Interface

The parallel interface provides a 16-bit parallel, bidirectional data path to another computer system or compatible peripheral device. The interface consists of one bidirectional word-wide data port, and two low-byte only control and status registers.

2.4.1 Parallel Data Register - This register performs two functions:

- o Writes to this register set the data output lines to the value written.
- o Reads from this register return the data present on the data input lines.

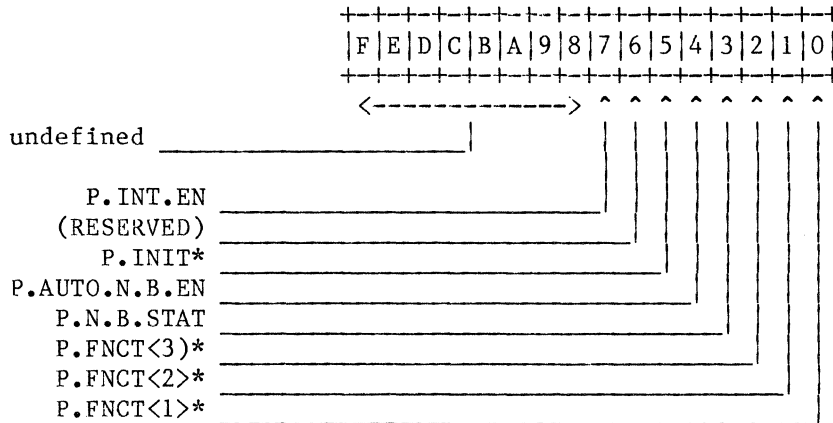
2.4.2 Parallel Status Register - This register contains information about the state of the parallel link. The bits are allocated as follows:



- o PEND.P.ATTN* is low whenever a parallel ATTN or INIT has occurred since the last time this bit was cleared. This bit is read/write.
- o PEND.P.CYCLE* is low whenever a parallel cycle request has occurred since the last read from or write to the parallel data register. This bit is automatically set to a "1" by accesses to the parallel data register. Accesses to the parallel data register should be made only when this bit is a "0." This bit is cleared by a write to the parallel status register with D<6> clear.
- o P.ATTN* is low while a parallel attention is active. This bit is read only.

- o P.CY.RQ* is low while a parallel cycle request is active and indicates that the device is either requesting new data or has not yet acknowledged the receipt of data already sent. This bit is useful primarily for diagnostic purposes and should not be used to determine whether the device is ready for more data. This bit is read only.
- o P.J2.26* is low whenever connector J2, pin 26 is active. This is an unused input bit. This bit is read only.
- o P.STATUS.C* is low whenever the STATUS.C line is active. This bit is read only.
- o P.STATUS.B* is low whenever the STATUS.B line is active. This bit is read only.
- o P.STATUS.A* is low whenever the STATUS.A line is active. This bit is read only.

2.4.3 Parallel Control Register - This register configures the parallel interface and generates hand-shaking signals. The bits are allocated as follows:



- o P.INT.EN, when set to "1," allows PEND.P.CYCLE to generate an interrupt request to the IOP.
- o Reserved
- o P.INIT*, when set to a "0," sets the hand-shaking signal INIT to a "1" and when "1," sets INIT to a "0."

- o P.AUTO.N.B.EN, when set to a "1," causes the hand-shaking signal NOT.BUSY to be generated automatically with each read from or write to the data register; NOT.BUSY is pulsed inactive (indicating BUSY) during the operation.
- o P.N.B.STAT, when set to a "1," forces NOT.BUSY active. When P.AUTO.N.B.EN is set to a "1," this bit always should be set to a "0." When P.AUTO.N.B.EN is set to a "0," NOT.BUSY always assumes the state of this bit.
- o P.FNCT<3>*, when set to a "0," sets hand-shaking signal FNCT<3> to a "1."
- o P.FNCT<2>*, when set to a "0," sets hand-shaking signal FNCT<2> to a "1."
- o P.FNCT<1>*, when set to a "0," sets hand-shaking signal FNCT<1> to a "1."

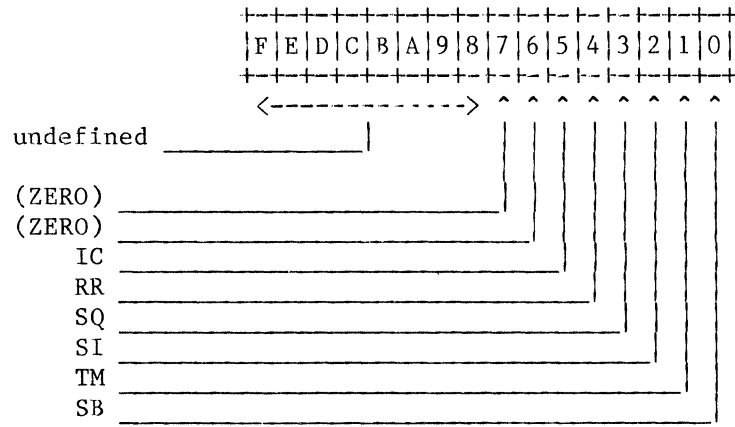
This register is read/write and is cleared on system reset.

2.5 The Serial Ports

The PIB contains two RS-449 serial ports implemented with an Intel 8274 USART, supplementary status and control registers, and associated RS-422 drivers and receivers. Both ports are configured as Data Terminal Equipment ports as described in EIA standard RS-449.

The Intel 8274 USART contains eight registers selected by addresses A<3..1>. A description of the 8274 may be found in the Intel Component Data Catalog. The send and receive data are generated/sampled by the 8274, and the RS-449 CS and DM signals are received by the 8274 as CTS and CD, respectively. The remaining RS-449 status signals for each channel are generated/sampled by supplementary control and status registers as follows.

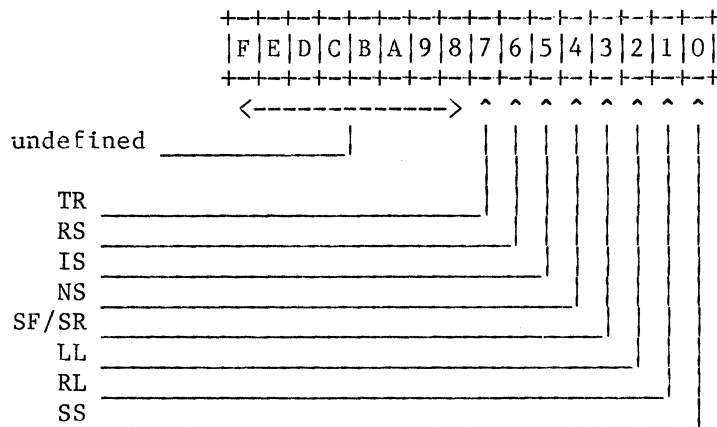
2.5.1 Serial Status Registers - These registers contains the values of the RS-449 status signals (other than CS and DM) that are received by the DTE. The bits are allocated as follows:



- o This bit always is zero.
- o This bit always is zero.
- o IC reflects the value of the RS-449 IC signal.
- o RR reflects the value of the RS-449 RR signal.
- o SQ reflects the value of the RS-449 SQ signal.
- o SI reflects the value of the RS-449 SI signal.
- o TM reflects the value of the RS-449 TM signal.
- o SB reflects the value of the RS-449 SB signal.

This register is read only.

2.5.2 Serial Control Registers - These registers generate the RS-449 status signals that are generated by the DTE. The bits are allocated as follows:

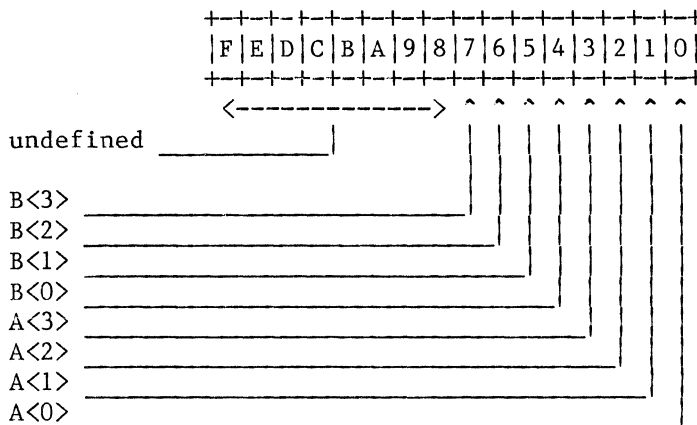


- o TR sets the RS-449 TR signal.
- o RS sets the RS-449 RS signal.
- o IS sets the RS-449 IS signal.
- o NS sets the RS-449 NS signal.
- o SF/SR sets the RS-449 SF/SR signal.
- o LL sets the RS-449 LL signal.

- o RL sets the RS-449 RL signal.
- o SS sets the RS-449 SS signal.

This register is read/write and is cleared on system reset.

2.5.3 Clock Generation - The 8274 may receive the transmit and receive clocks either from the RS-449 port, or from an on-board baud rate generator. The output of the baud rate generator also drives the RS-449 timing signal (TT). The baud rate generator used is the Western Digital BR 1941-06 (for additional information on this device, refer to the manufacturer's data). A write to the baud rate generator stores the low-order four bits of the low-order byte into the A port clock select and stores the high-order four bits of the low-order byte into the B port clock select. This register is write only.



By programming the 8274 to use either x1, x16, x32 or x64 clocks, a wide range of data rates may be obtained.

2.6 The 60 Hz Clock Interrupt.

To enable the IOP to maintain a rough approximation of real time, a nominal (+10%) 60 Hz clock interrupt is provided. An interrupt request is set every clock period and remains set until it is cleared though a write to the 60 Hz clock clear address. Any write to this address clears the interrupt regardless of the data being written.

2.7 Interrupts

Interrupt requests to the IOP can be made by several devices:

- o 8274 USART
- o 60 Hz clock
- o Versatec cycle request
- o Parallel attention
- o Parallel cycle request
- o Doorbell 0
- o Multibus interrupt

The 8086 accepts an 8-bit vector on interrupts and fetches the location of the interrupt service routine from a table in local memory. The 8274 generates a set of eight consecutive vectors, the base of which is programmable, to indicate the type of request. The other interrupts are vectored as follows:

PEND.60HZ	V.INT.REQ	PEND.P.ATTN	P.INT.REQ	VECTOR
-	-	-	-	1FH
-	-	-	*	1EH
-	-	*	-	1DH
-	-	*	*	1CH
-	*	-	-	1BH
-	*	-	*	1AH
-	*	*	-	19H
-	*	*	*	18H
*	-	-	-	17H
*	-	-	*	16H
*	-	*	-	15H
*	-	*	*	14H
*	*	-	-	13H
*	*	-	*	12H
*	*	*	-	11H
*	*	*	*	10H

A "*" indicates an interrupt request, a "-" indicates no interrupt.

The 8274 has priority over all other interrupt requests. Priorities of the remaining devices are software-configurable since each combination of requests presents a different vector. To determine whether a DOORBELLO interrupt is pending, it is necessary to read the corresponding bit in the board status register. Multibus interrupts can be recognized by the lack of other causes for the interrupt. Versatec and parallel cycle requests may be disabled by the appropriate bit in their control registers to allow a process that services one of these devices to operate in a high-speed, polled mode (with interrupts from the device disabled) and still receive interrupts from other sources.

3.0 MAINTENANCE

Maintenance is enhanced by several features, for example, the IOP's ability to run local diagnostics. The results of these diagnostics are indicated via two on-board LEDs, a red LED and a green LED. Additionally, a maintenance connector is included on the board. This connector includes critical signals to be monitored for fault diagnosis as well as the signals that must be driven to single-step the IOP. Finally, since the PIB can operate as a Multibus slave, the system CPU itself can diagnose most on-board functions (except for correct operation of the IOP).

4.0 OPTIONS

The PIB has the following strappable options:

- o The IOP may be reset either by the 796Bus reset signal (INIT) or by DEV.EN.
- o 4 or 16KBy of on-board RAM (the board must also be appropriately populated).
- o Relinquishing Multibus mastership on each clock cycle or only when there is an outstanding request.
- o Priority level of the DOORBELL1 interrupt.
- o Disabling DOORBELL1 interrupts.
- o Priority level of the Multibus interrupt (if any) monitored by the IOP.
- o Operation of the rest of the board without the serial port components.

APPENDIX A

STRAPPING OPTIONS

INTRODUCTION

This appendix describes the full strapping options and switch settings for the PIB, assembly 710-00008 and includes the default factory jumper strap positions. Jumper straps are either suit-case between terminal posts or, as with interrupt level selection, insulated wire-wrap jumpers.

The following conventions are used in the strapping configuration table:

Default jumper strap positions are noted by an asterisk (*).

An equals sign (=) indicates a single possible connection between the pin listed to the left of the equals sign and one of the pins listed to the right of the equals sign.

NC indicates that no jumper strap is installed.

JUMPER STRAPS

Board Location	Strap	Description
26HB	2 = 1	Select slow local device ACK
	3*	Select fast local device ACK
26HA	2 = 1	Select 4Kbyte local RAM
	NC*	Select 16Kbyte local RAM
30B	2 = 1	Enable software Multibus Lock
	3*	Disable software Multibus Lock
	5 = 4	Enable CBRQ
	6	Disable CBRQ
20FA	1 = 2*	Select 16Kbyte local RAM
	NC	Select 4Kbyte local RAM
6AB	2 = 1*	Reset IOP from status register
	3	Reset IOP from system bus (INIT)

Board Location	Strap	Description
30E	1 = 2	Enable CPU interrupt source
	3	Enable Multibus interrupt level 7
	4	Enable Multibus interrupt level 6
	5	Enable Multibus interrupt level 5
	6	Enable Multibus interrupt level 4
	7	Enable Multibus interrupt level 3
	8	Enable Multibus interrupt level 2
	9	Enable Multibus interrupt level 1
	10	Enable Multibus interrupt level 0
	28L	2 = 1
3*		Route A<22> to P2-59
5 = 4		Route A<23> to P2-56
6*		Route A<23> to P2-60
26B	2 = 1	Use BCLK* for system clock
	3	Use CCLK* for system clock
22BA	1 = 2	Configure PIB for head of BPRN chain
	NC*	Configure PIB to receive BPRN from higher-priority bus master.

SWITCHES

The individual switches at 28C select the upper eight address bits of the 64K-byte window occupied by the PIB. OFF is a "1," and ON is a "0."

APPENDIX B
ASSEMBLY DRAWING

APPENDIX C
SCHEMATIC DRAWINGS

4

3

2

1

NOTE: THIS PAGE PROVIDES A RUNNING HISTORY OF ALL CHANGES ASSOCIATED WITH THIS DWG

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED


REV	SHEETS	DESCRIPTION	DATE	APPROVED	REV	SHEETS	DESCRIPTION	DATE	APPROVED
A		AS ISSUED PER ECD #108	10-29-82	<i>G. Cegun</i>					
B	5, 10	REVISED PER ECD #118	12-22-82	<i>G. Cegun</i>					
B ₁	23, 24	CORRECT PER DCO # 115	2-15-84	<i>G. Cegun</i>					

D

C

B

A

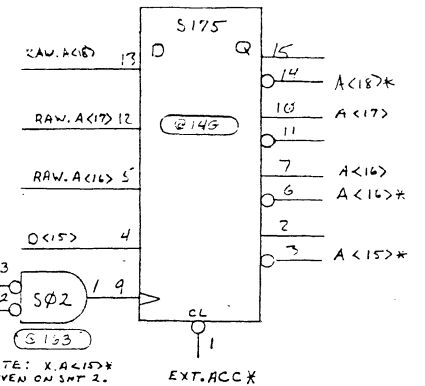
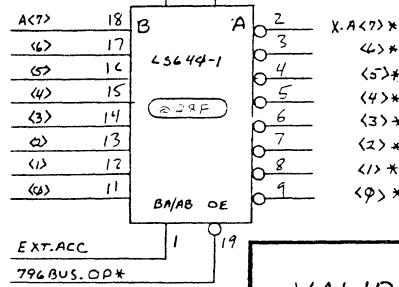
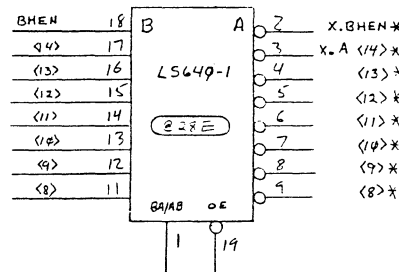
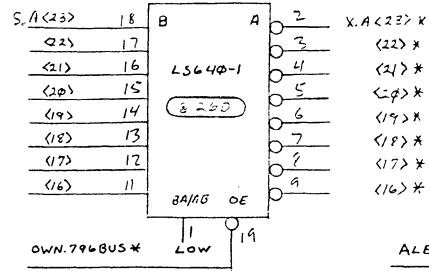
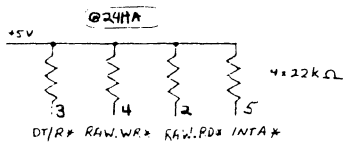
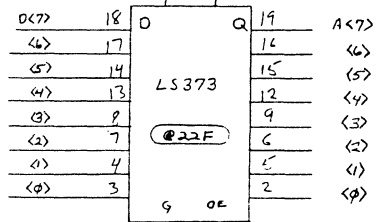
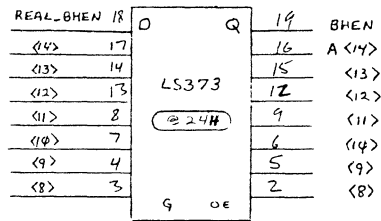
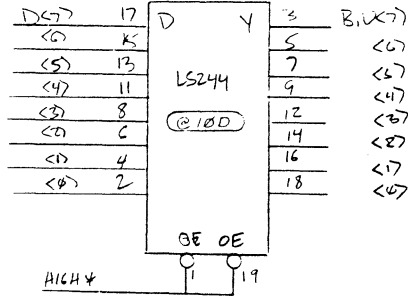
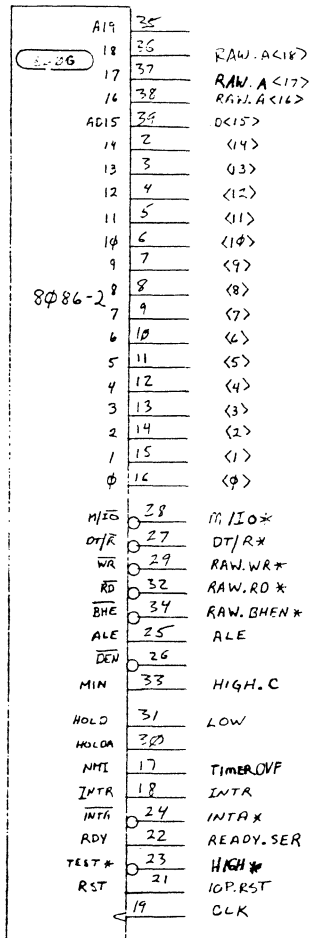
<small>UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ± ±</small>	CONTRACT NO.		SCHEMATIC PERIPHERAL INTERFACE
	APPROVALS	DATE	
	DRAWN <i>TILLEMANN</i>	10-27-82	
	CHECKED <i>[Signature]</i>	10-29-82	
MATERIAL	<i>G. Cegun</i>	10-29-82	DRAWING NO. 710-00008-2
FINISH	SIZE B CODE IDENT NO. REV B		
DO NOT SCALE DRAWING			SCALE SHEET 1 OF 31

4

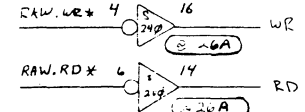
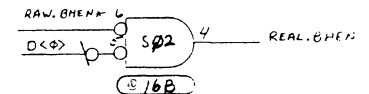
3

2

1



NOTE: X.A<15>* DRIVEN ON SMT 2.



VALID LOGIC SYSTEMS INC.

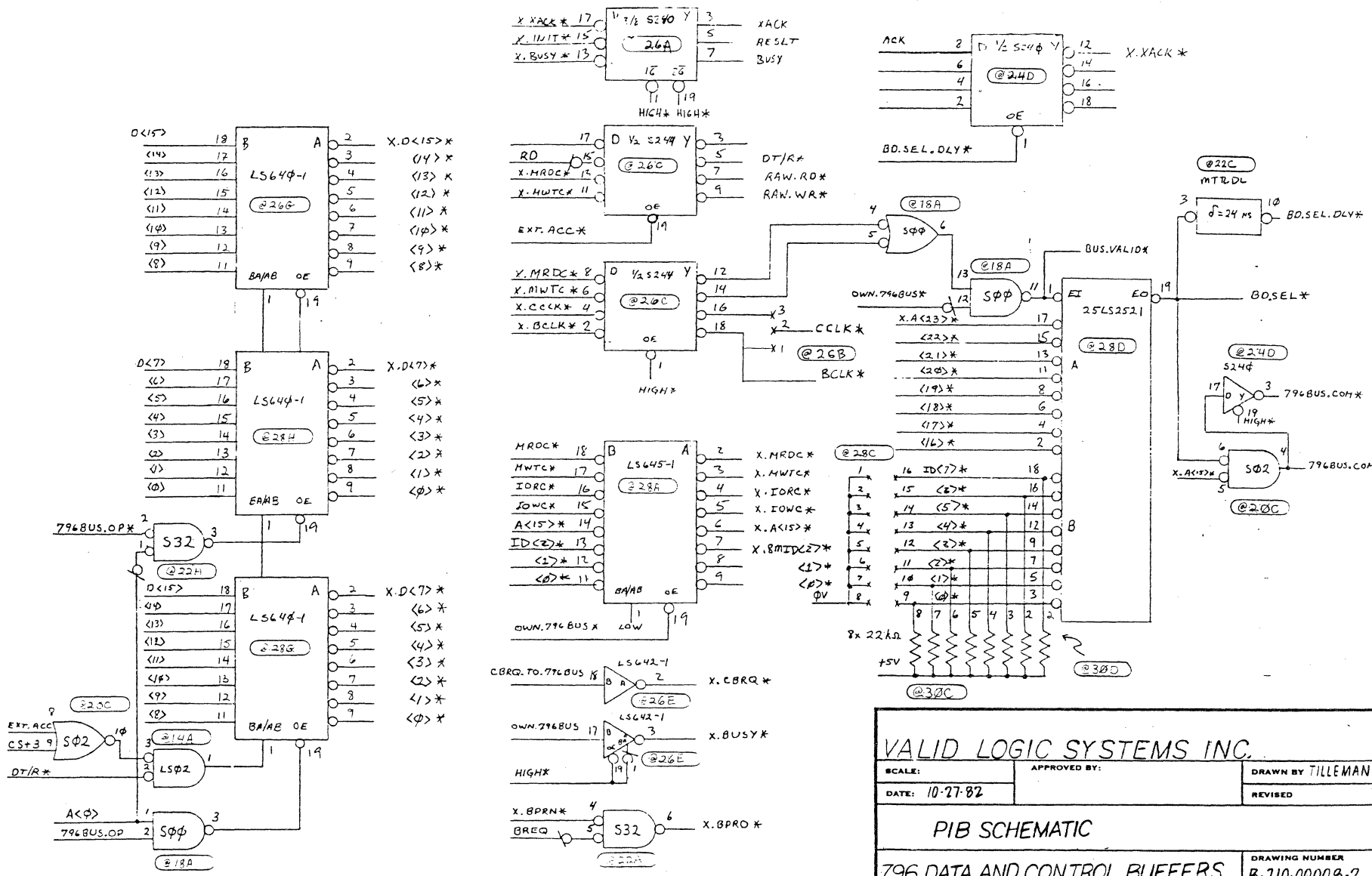
SCALE:	APPROVED BY:	DRAWN BY TILLEMAN
DATE: 10-27-82		REVISED

PIB SCHEMATIC

8086 ADDRESS BUFFERS

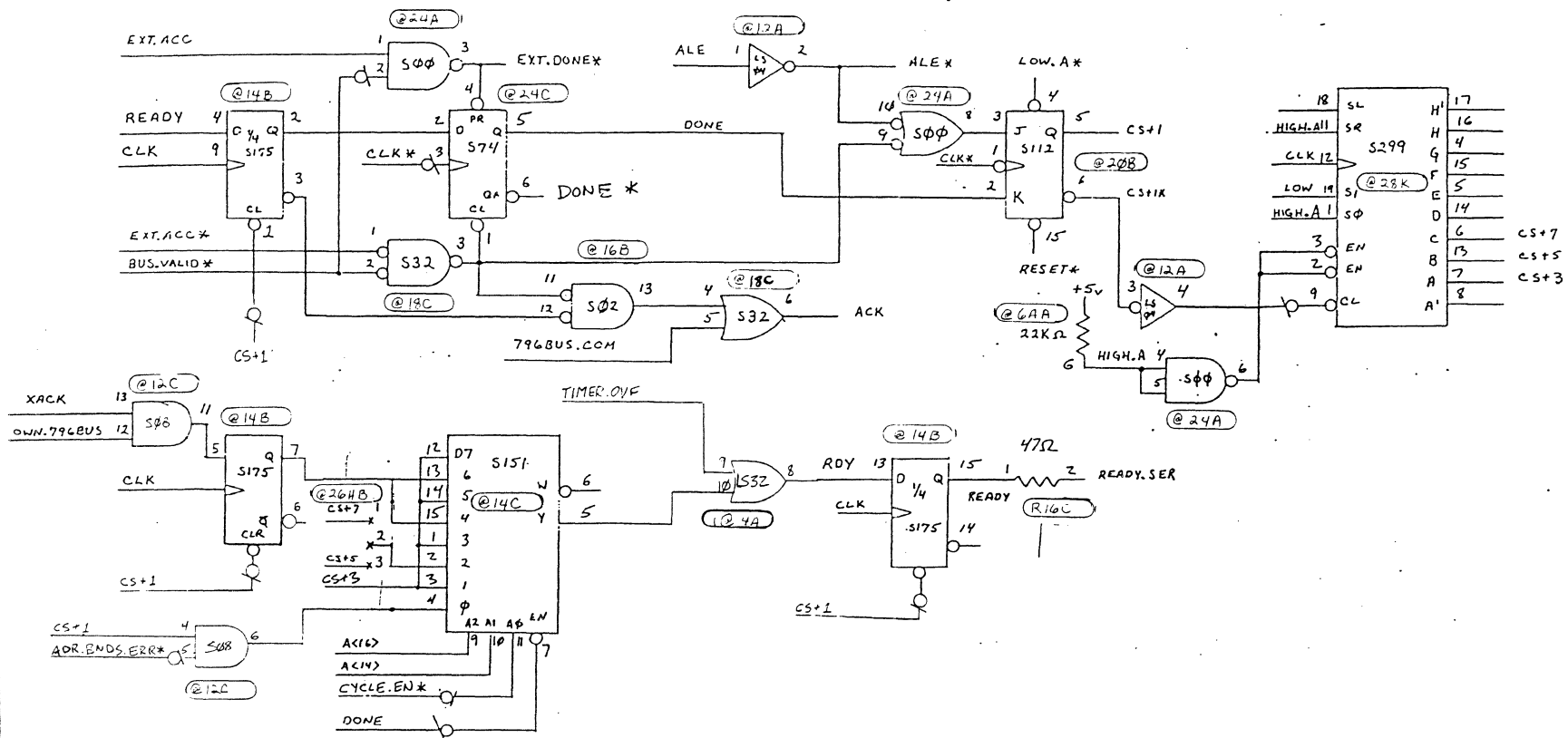
DRAWING NUMBER
B-710-00008-2

SHEET 2 OF 31



VALID LOGIC SYSTEMS INC.

SCALE:	APPROVED BY:	DRAWN BY TILLEMAN
DATE: 10-27-82		REVISED
PIB SCHEMATIC		
796 DATA AND CONTROL BUFFERS		DRAWING NUMBER B-710-00008-2



ACK	A(16)	A(15)	A(14)	A(13)	A(12)	ACCESS TYPE
X	0	0	0	0	0	LOCAL RAM
X	0	0	0	1	0	LOCAL RAM
X	0	0	1	0	0	LOCAL DEV
X	0	1	1	1	1	GDC DMA
0	1	X	0	0	0	796BUS (L4)
0	1	X	1	0	0	796BUS (IC)
1	1	X	0	0	0	796BUS (MEM)
1	1	X	1	0	0	796BUS (MEM)

VALID LOGIC SYSTEMS INC.

SCALE:
DATE: 10-27-82

APPROVED BY:

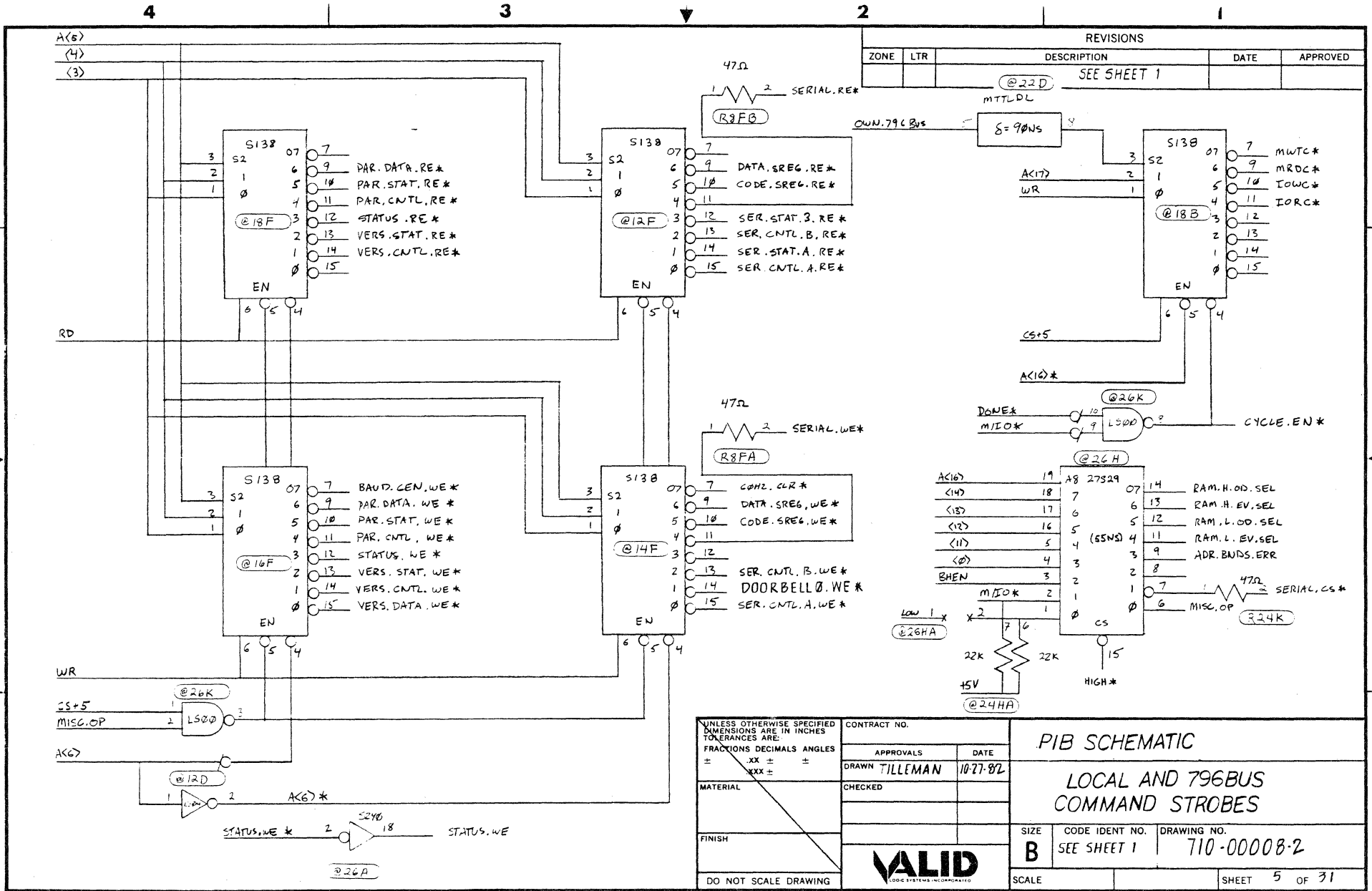
DRAWN BY TILLEMAN
REVISED

PIB SCHEMATIC

LOCAL BUS AND ACK TIMING

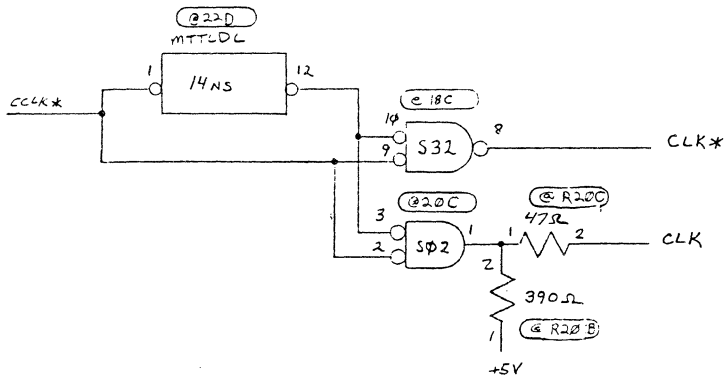
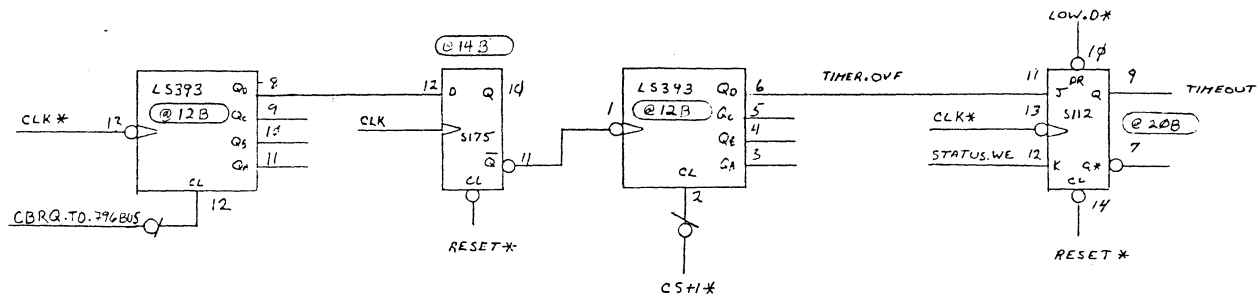
DRAWING NUMBER
B-710-00008-2

SHEET 4 OF 31



REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		@22D SEE SHEET 1		

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± ± .XXX ± ±	CONTRACT NO.		PIB SCHEMATIC		
	APPROVALS				
	DRAWN TILLEMANN		DATE 10-27-82		
	CHECKED				
MATERIAL			LOCAL AND 796BUS COMMAND STROBES		
FINISH					
DO NOT SCALE DRAWING	VALID LOGIC SYSTEMS, INCORPORATED		SIZE B	CODE IDENT NO. SEE SHEET 1	DRAWING NO. 710-00008-2
SCALE			SHEET 5 OF 31		



VALID LOGIC SYSTEMS INC.

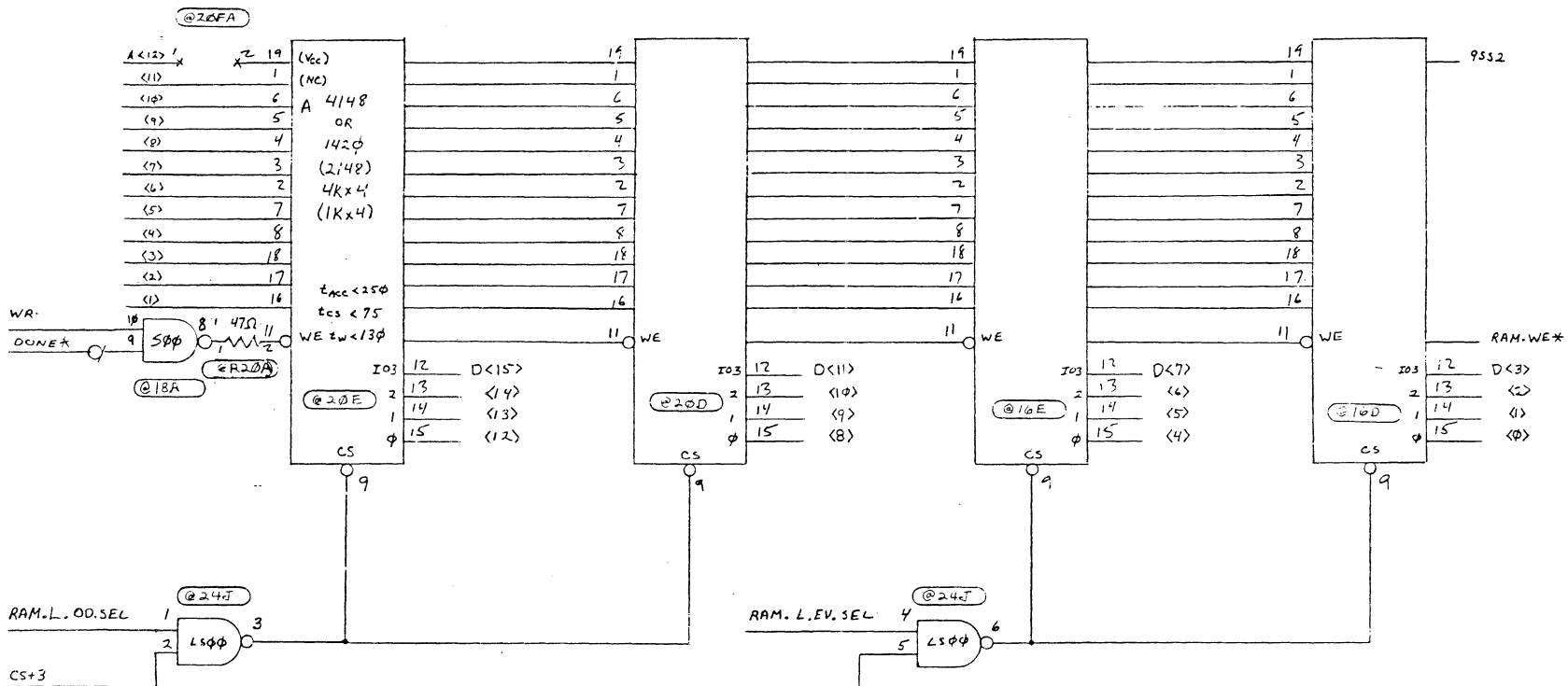
SCALE:	APPROVED BY:	DRAWN BY TILLEMANN
DATE: 10-27-87		REVISED

PIB SCHEMATIC

TIMEOUT AND CLOCK

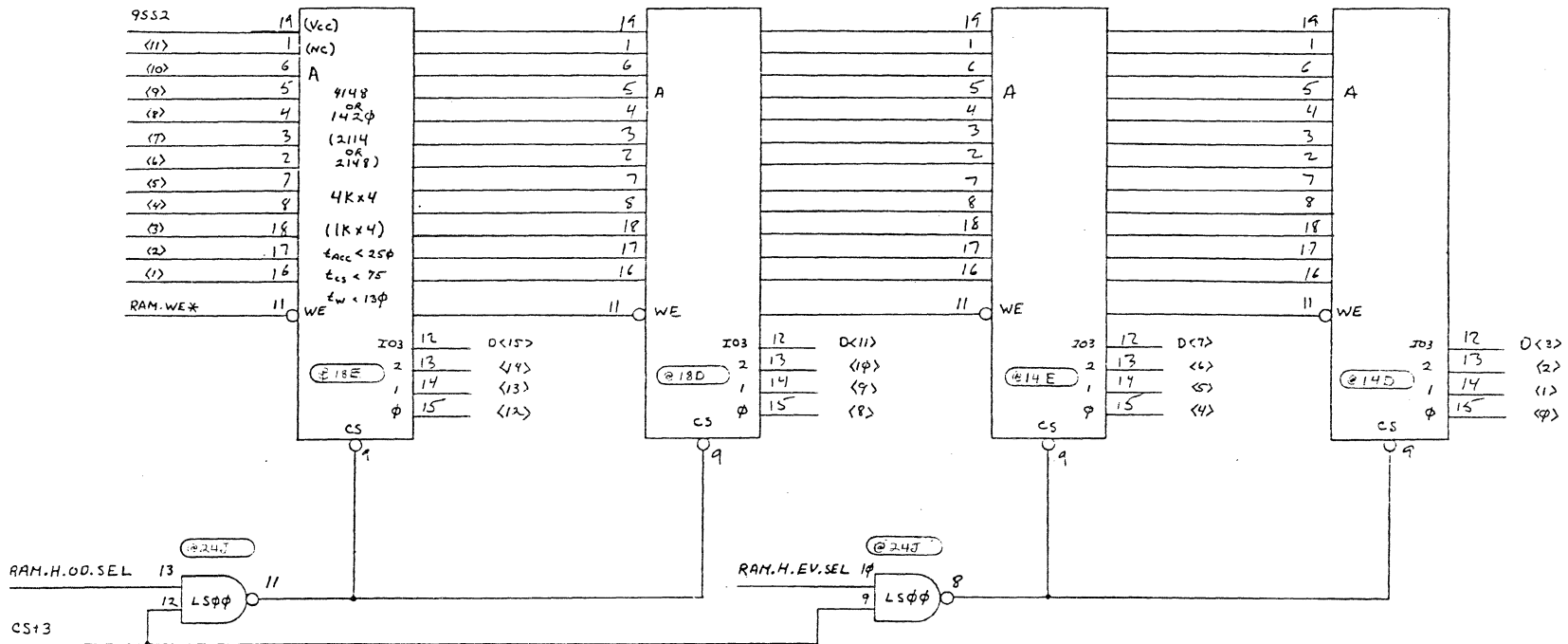
DRAWING NUMBER
B-710-00008-2

SHEET 7 OF 31



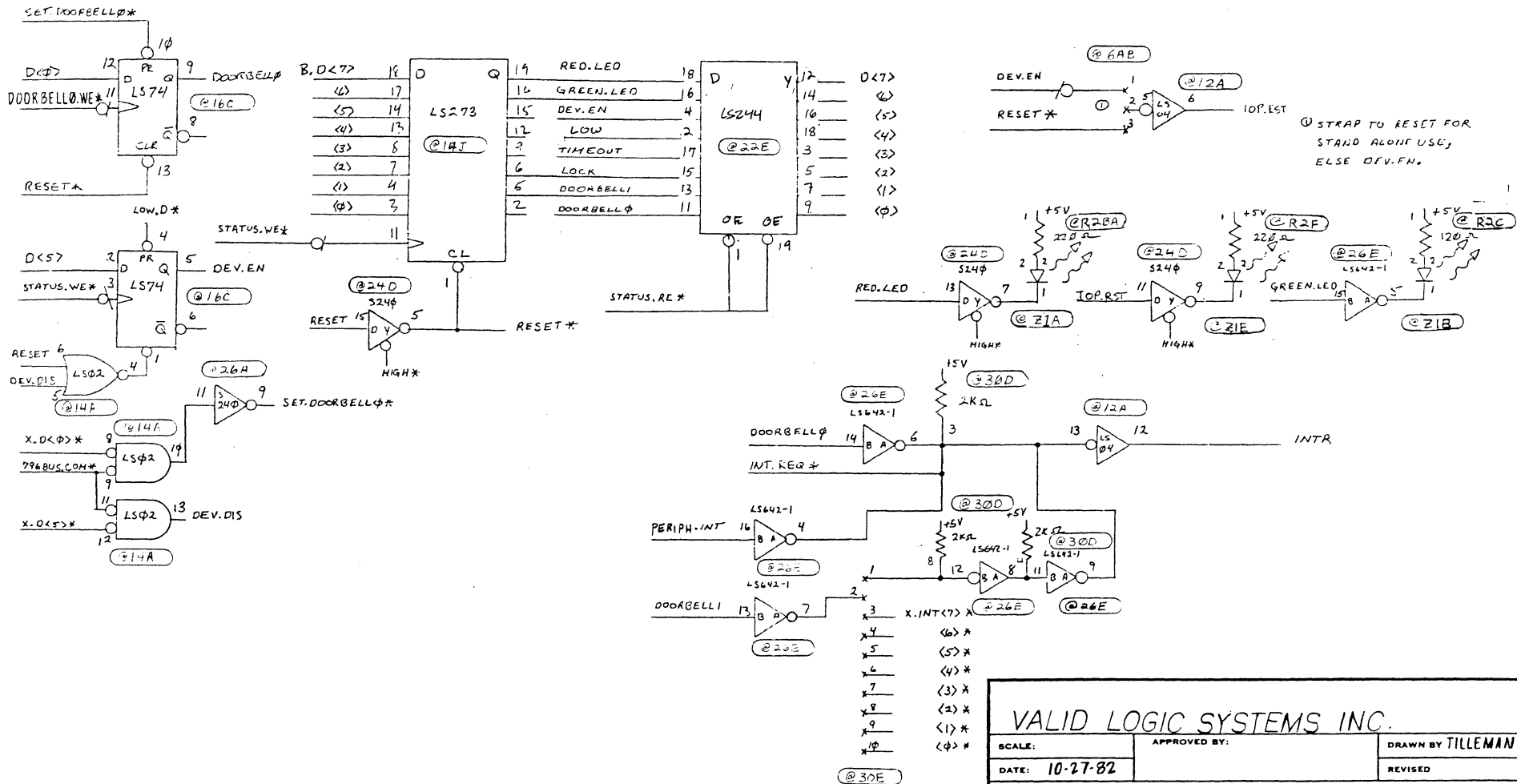
* NOTE: EACH RAM HAS A CUTTABLE TRACE FROM PIN 19 TO PIN 2φ. TRACE IS CUT AND JUMPER INSTALLED FOR 4Kx4 RAMS.

VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY TILLEMANN
DATE: 10-27-82		REVISED
PIB SCHEMATIC		
LOCAL MEMORY (1 OF 2)		DRAWING NUMBER B 710-00008-2



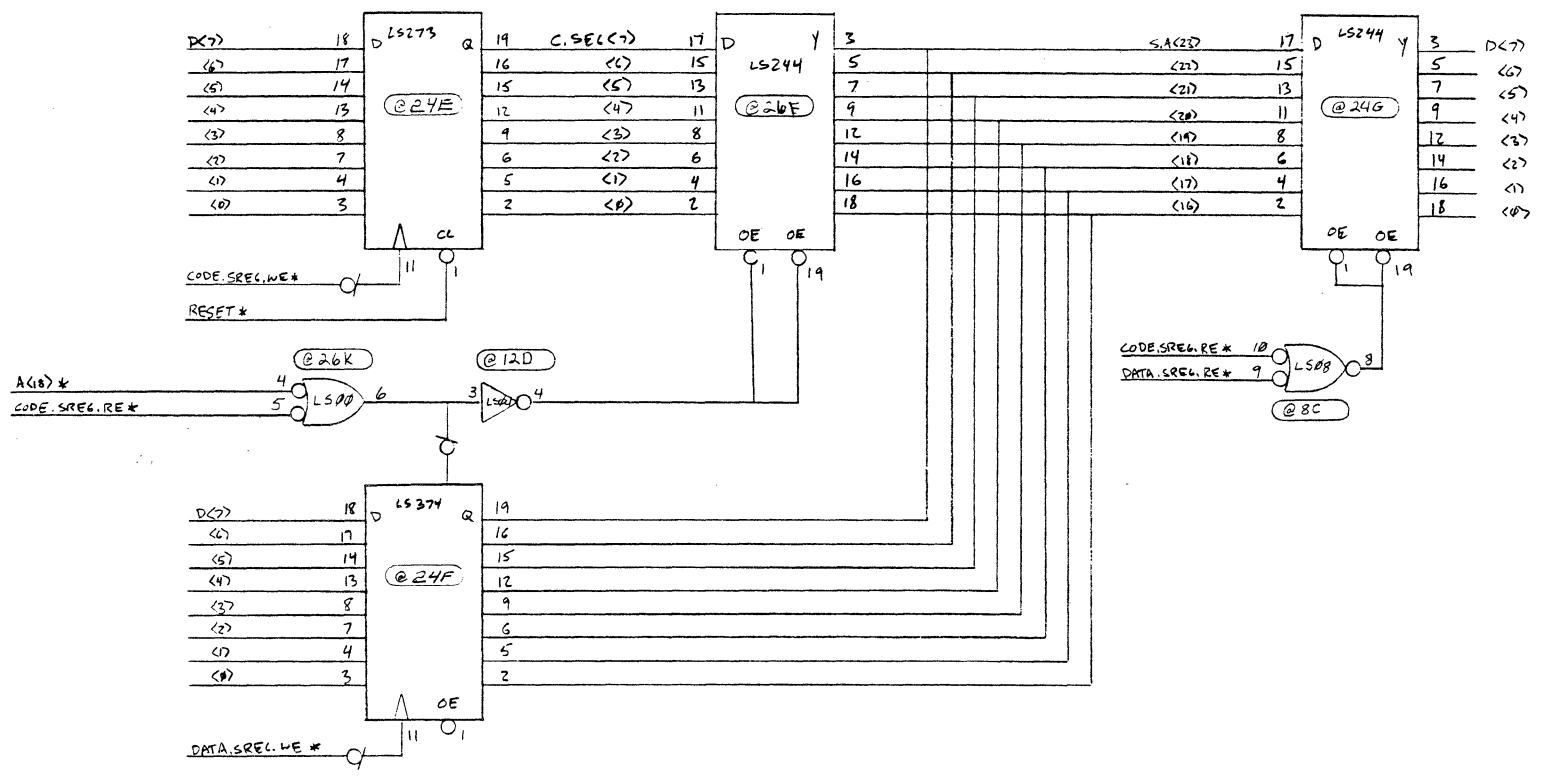
* NOTE: EACH RAM HAS A CUTTABLE TRACE FROM PIN 19 TO PIN 2φ. TRACE IS CUT AND JUMPER INSTALLED FOR 4Kx4 RAMS.

VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY TILLEMAN
DATE: 10-27-82		REVISED
PIB SCHEMATIC		
LOCAL MEMORY (2 OF 2)		DRAWING NUMBER B-710-00008-2



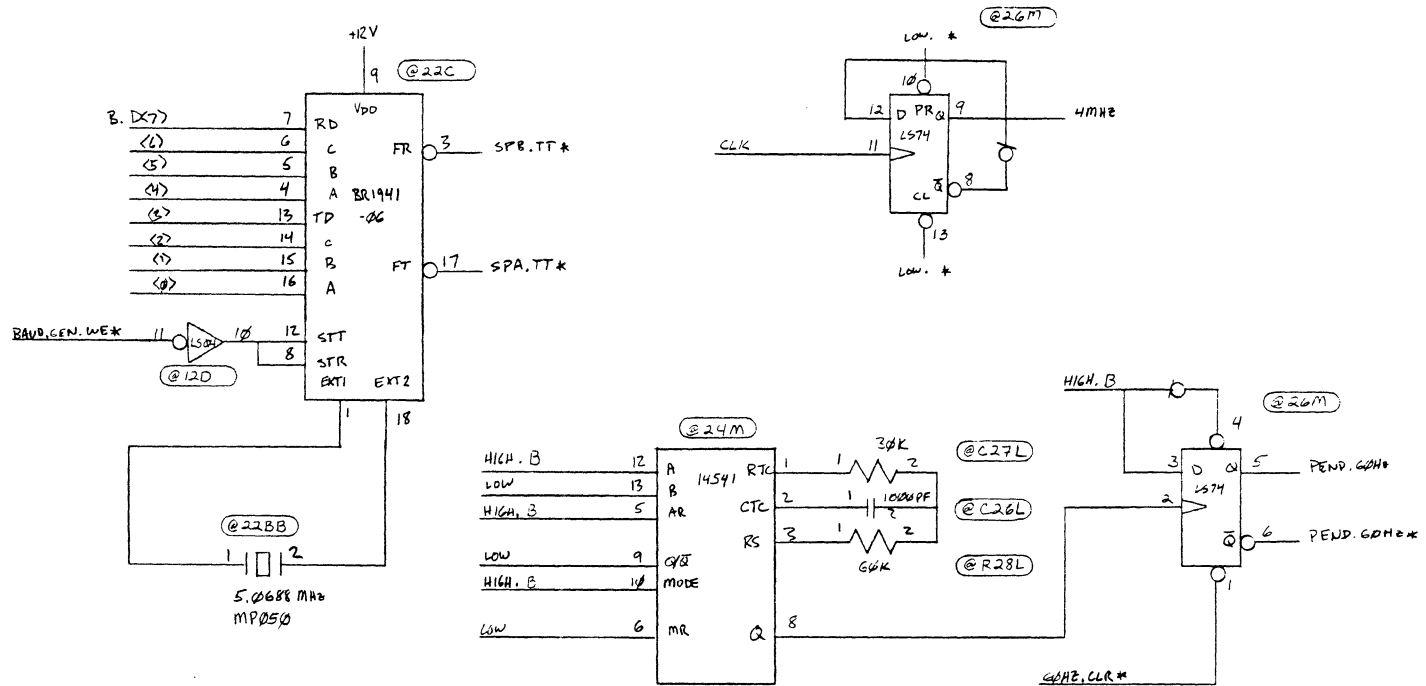
VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY TILLEMAN
DATE: 10-27-82		REVISED
PIB SCHEMATIC		
STATUS REGISTER & INTERRUPT		DRAWING NUMBER B-710-00008-2

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ±	CONTRACT NO.		PIB SCHEMATIC SEGMENT REGISTER
	APPROVALS	DATE	
	DRAWN TILLEMAN	10-27-82	
	CHECKED		
MATERIAL			SIZE B
FINISH			CODE IDENT NO. SEE SHEET 1
DO NOT SCALE DRAWING		DRAWING NO. 710-00008-2	
SCALE			SHEET 11 OF 31

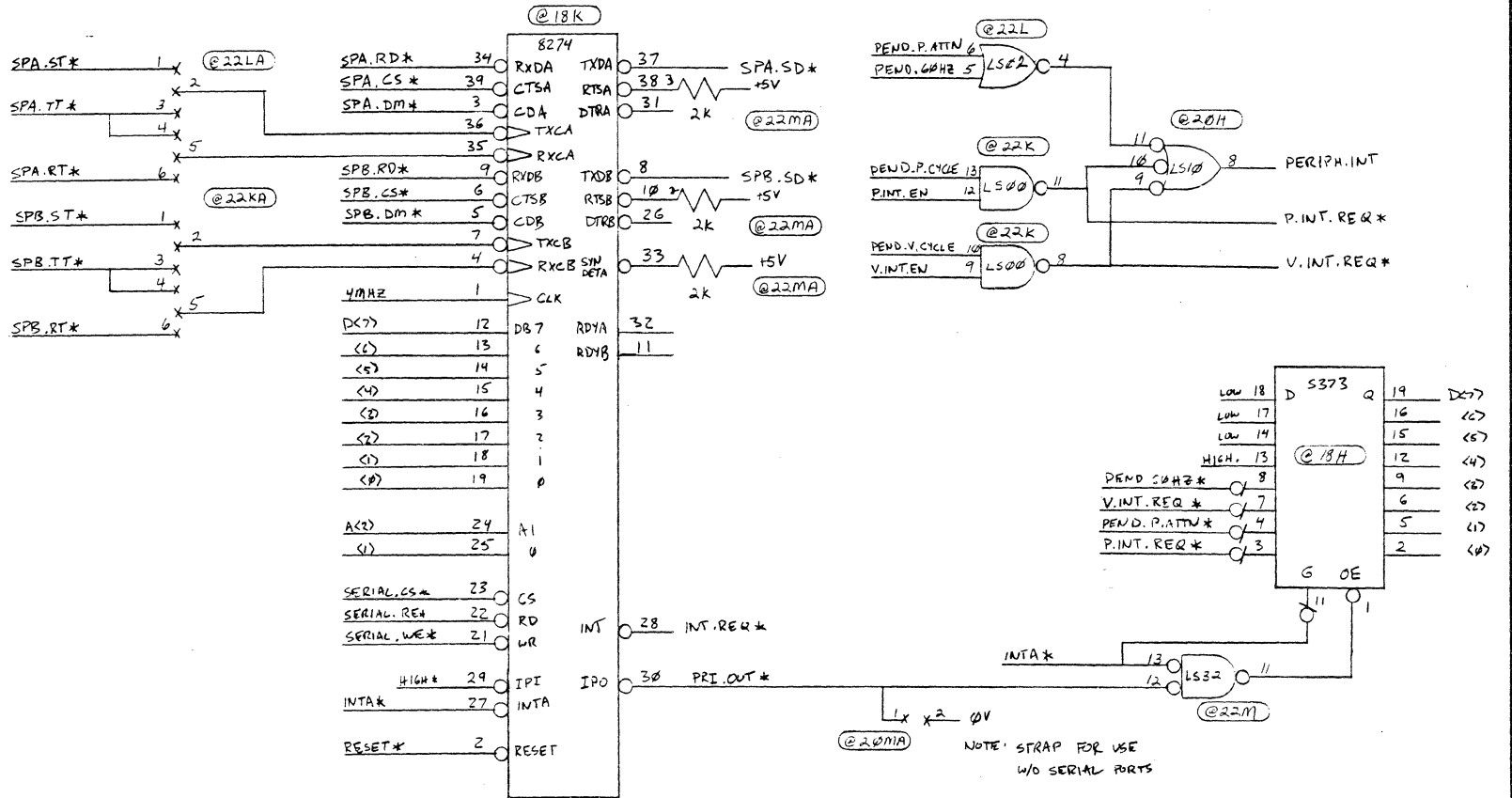
REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		



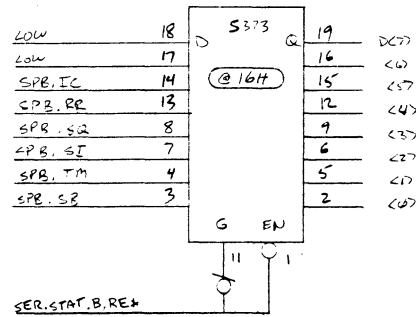
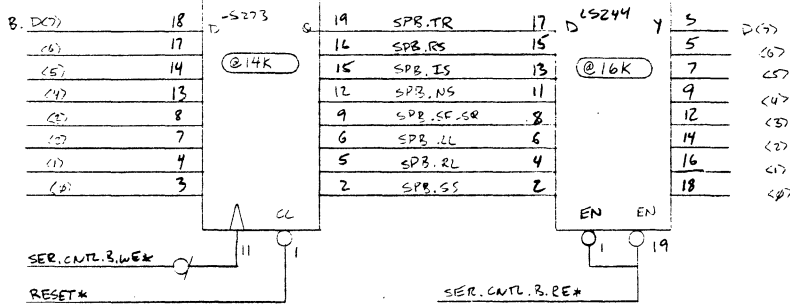
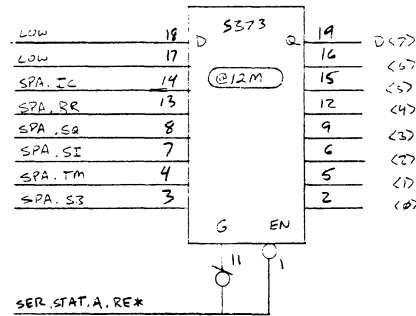
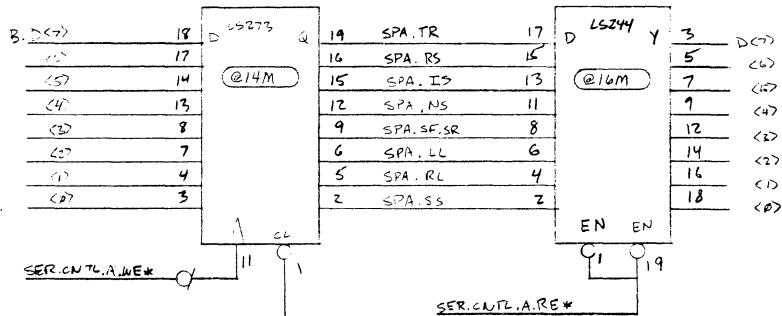
NOTE: $f \approx 15 \text{ kHz}$
 $f_Q \approx f/256 \approx 60 \text{ Hz}$

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ±	CONTRACT NO.		PIB SCHEMATIC BAUD RATE GENERATION		
	APPROVALS	DATE			
	MATERIAL	DRAWN TILLEMANN	10-27-82	SIZE B	CODE IDENT NO. SEE SHEET 1
	FINISH	CHECKED		DRAWING NO. 710-00008-2	
DO NOT SCALE DRAWING			SCALE	SHEET 12 OF 31	

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± .XXX ±	CONTRACT NO.		PIB SCHEMATIC UART AND INTERRUPT VECTOR	
	APPROVALS			
	DRAWN TILLEMANN 10-27-82		DATE	DRAWING NO. 710-00008-2
	CHECKED		SCALE	
MATERIAL	FINISH		SIZE B	CODE IDENT NO. SEE SHEET 1
DO NOT SCALE DRAWING		VALID LOGIC SYSTEMS INCORPORATED		SHEET 13 OF 31



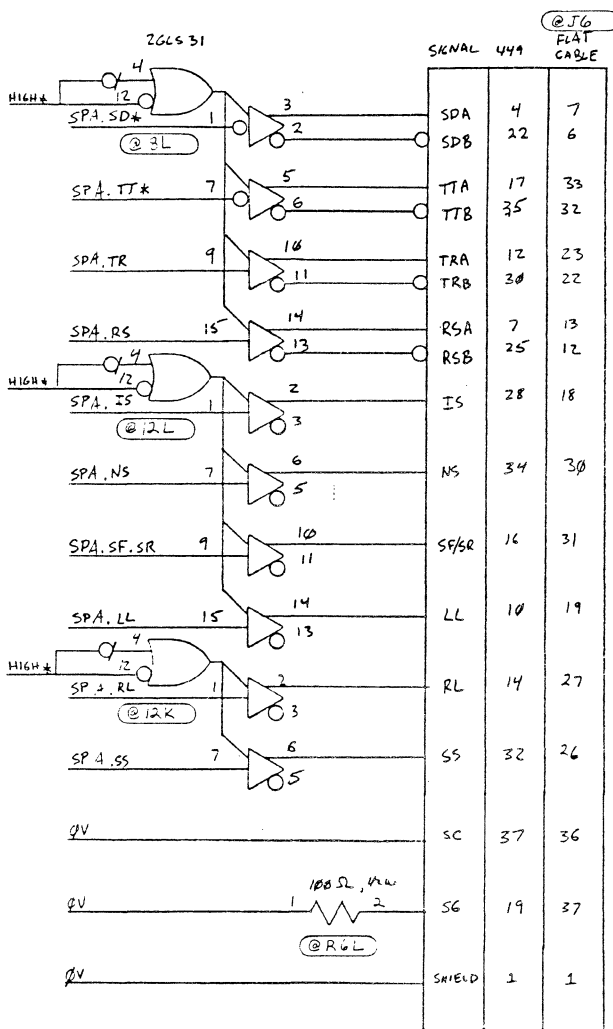
VALID LOGIC SYSTEMS INC.

SCALE: APPROVED BY: DRAWN BY: TILLEMAN
DATE: 10-27-82 REVISED:

PIB SCHEMATIC

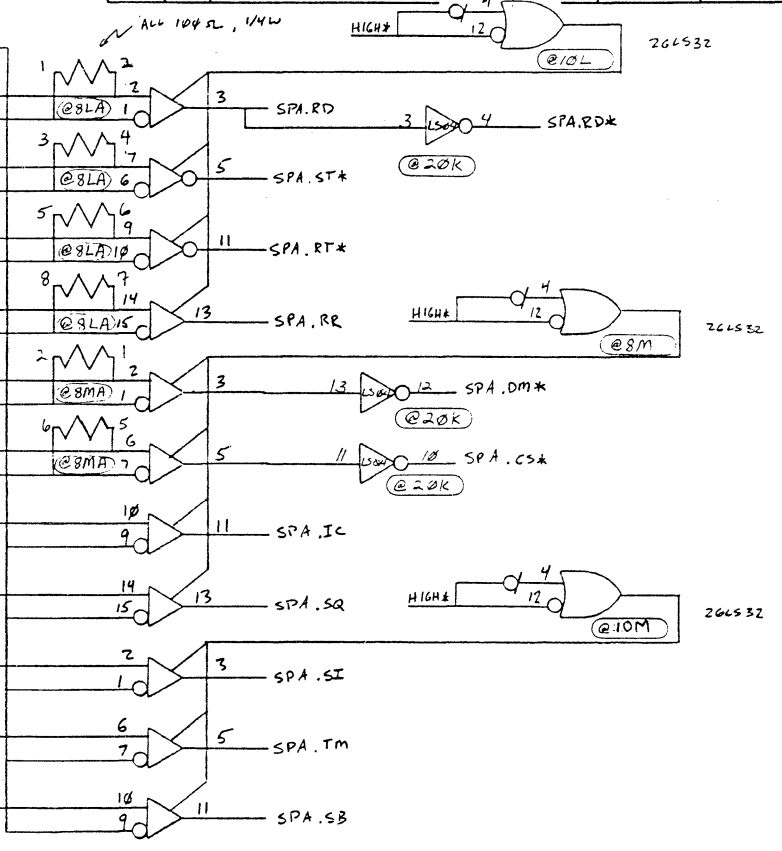
SERIAL PORT CONTROL

DRAWING NUMBER
B-710-00008-2



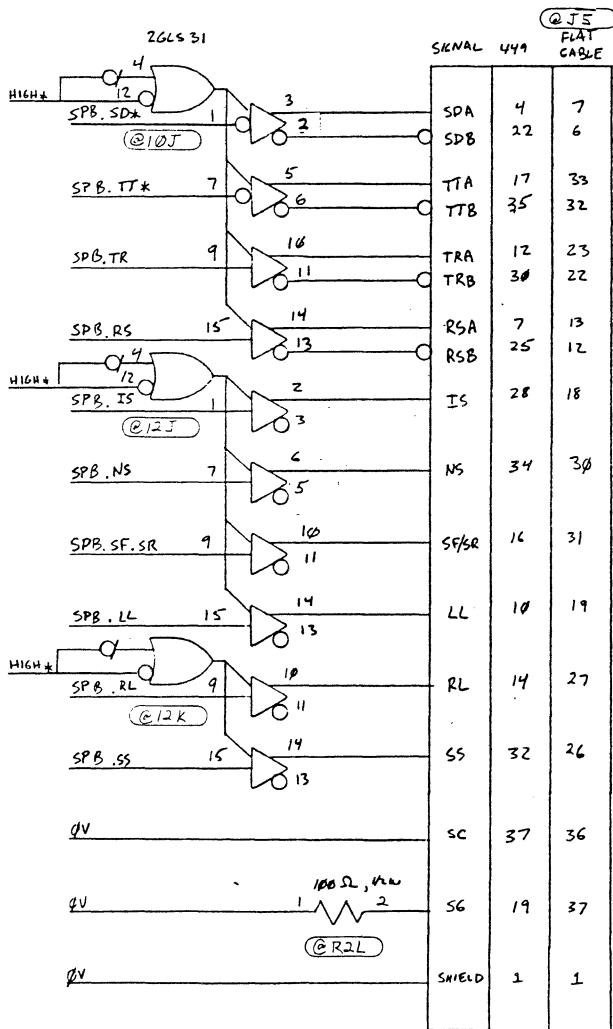
SIGNAL 449		FLAT CABLE @J6	
SDA	4	7	
SDB	22	6	
TTA	17	33	
TTB	35	32	
TRA	12	23	
TRB	30	22	
RSA	7	13	
RSB	25	12	
IS	28	18	
NS	34	30	
SF/SR	16	31	
LL	10	19	
RL	14	27	
SS	32	26	
SC	37	36	
SG	19	37	
SHIELD	1	1	

SIGNAL 449		FLAT CABLE @J6	
RC	20	2	
RDA	6	11	
RDB	24	10	
STA	5	9	
STB	23	8	
RTA	8	15	
RTB	26	14	
RRA	13	25	
RRB	31	24	
DMA	11	21	
DMB	29	20	
CSA	9	17	
CSB	27	16	
IC	15	29	
SA	33	28	
SI	2	3	
TM	18	35	
SB	36	34	



REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± ± .XXX ± ± MATERIAL FINISH DO NOT SCALE DRAWING	CONTRACT NO.		APPROVALS DRAWN TILLEMANN 10-27-82		DATE 10-27-82		PIB SCHEMATIC SERIAL DRIVERS & RECEIVERS (A)		
	CHECKED								
	SIZE	CODE IDENT NO.	DRAWING NO.		SHEET 15 OF 31				
	B	SEE SHEET 1	710-00008-2						

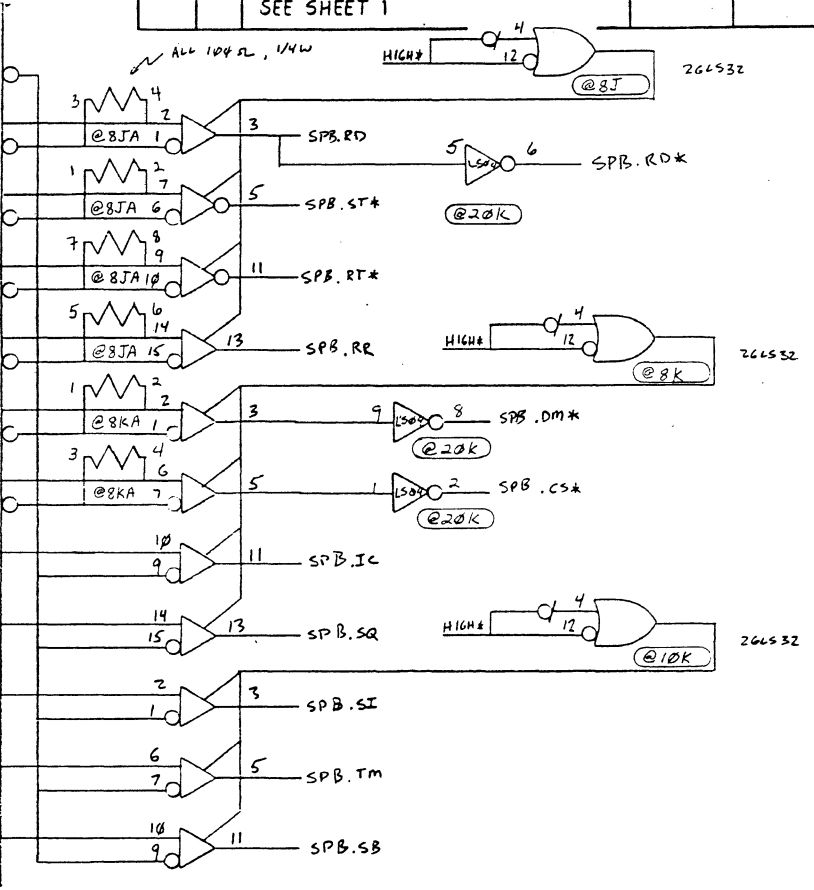


SIGNAL 449 FLAT CABLE @JS

SDA	4	7
SDB	22	6
TIA	17	33
TTB	35	32
TRA	12	23
TRB	30	22
RSA	7	13
RSB	25	12
IS	28	18
NS	34	30
SF/SR	16	31
LL	10	19
RL	14	27
SS	32	26
SC	37	36
SG	19	37
SHIELD	1	1

SIGNAL 449 FLAT CABLE @IS

RC	20	2
RDA	6	11
RDB	24	10
STA	5	9
STB	23	8
RTA	8	15
RTB	26	14
RRA	13	25
RRB	31	24
DMA	11	21
DMB	29	20
CSA	9	17
CSB	27	16
IC	15	29
SQ	33	28
SI	2	3
TM	18	35
SB	36	34

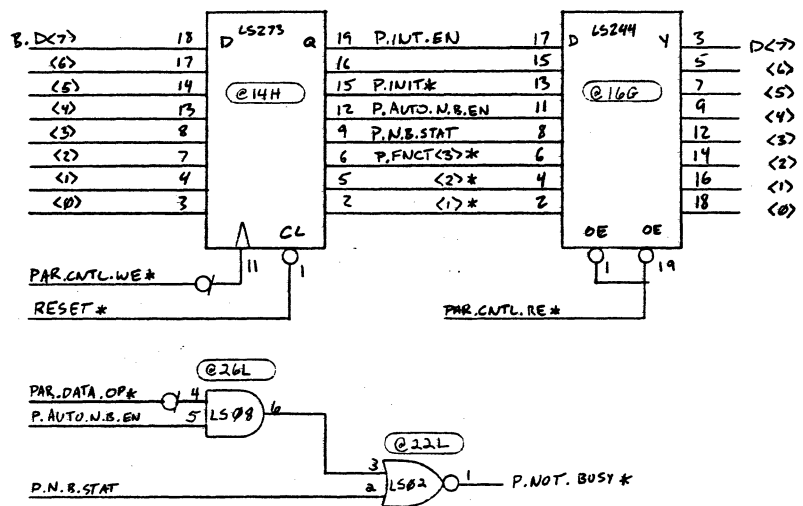


REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
SEE SHEET 1				

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± ± .XXX ± ±		CONTRACT NO.
MATERIAL		APPROVALS
FINISH		DATE
DO NOT SCALE DRAWING		DRAWN TILLEMANN 10-27-82
		CHECKED

PIB SCHEMATIC		
SERIAL DRIVERS & RECEIVERS (B)		
SIZE	CODE IDENT NO.	DRAWING NO.
B	SEE SHEET 1	710-00008-2
SCALE	SHEET 16 OF 31	

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± ± .XXX ±	CONTRACT NO.		PIB SCHEMATIC PARALLEL CONTROL REGISTER		
	APPROVALS	DATE			
	DRAWN TILLEMANN	10-27-82			
	CHECKED				
MATERIAL			SIZE	CODE IDENT NO.	DRAWING NO.
FINISH			B	SEE SHEET 1	710-00008-2
DO NOT SCALE DRAWING			SCALE	SHEET 17 OF 31	

4

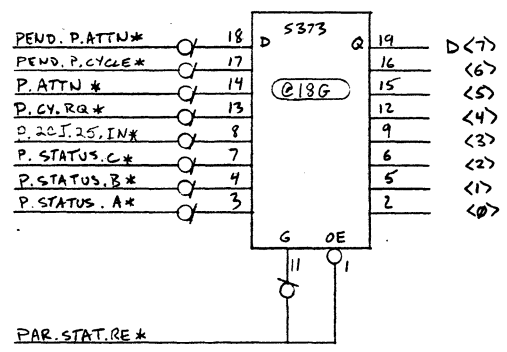
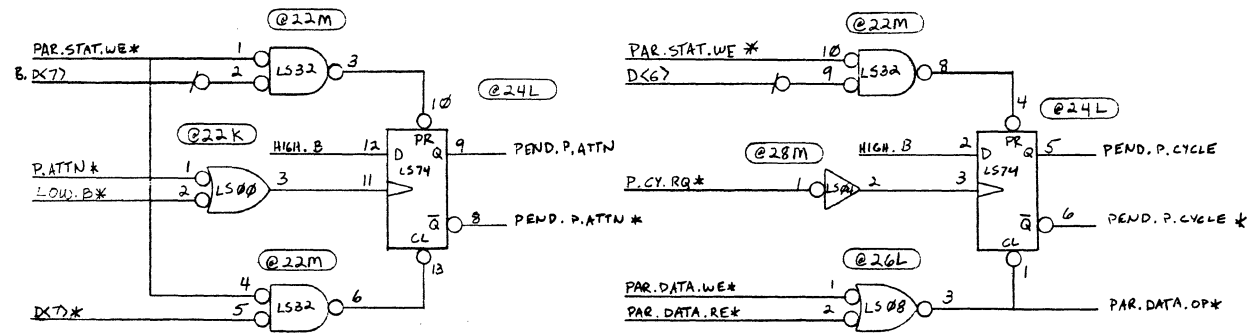
3

2

1

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		

NOTE: TWO VERSIONS OF D<7> * EXIST IN LW VERSION



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± ± .XXX ± ±	CONTRACT NO.		PIB SCHEMATIC PARALLEL STATUS REGISTER		
	APPROVALS	DATE			
	MATERIAL	CHECKED	DRAWN TILLEMANN	10-27-82	
	FINISH	DO NOT SCALE DRAWING	VALID LOGIC SYSTEMS INCORPORATED	SIZE B	CODE IDENT NO. SEE SHEET 1
SCALE			SHEET 18 OF 31		

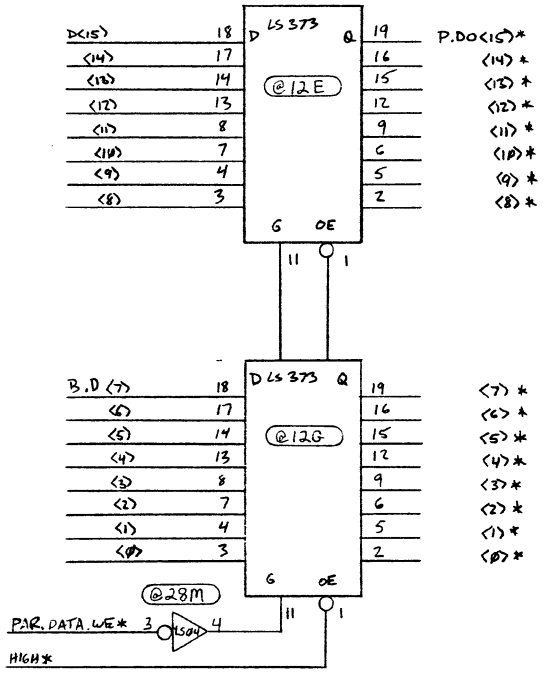
4

3

2

1

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± MATERIAL FINISH DO NOT SCALE DRAWING	CONTRACT NO.		APPROVALS		DATE	PIB SCHEMATIC PARALLEL DATA OUTPUT REGISTER
	DRAWN TILLEMAN		CHECKED		10/27/82	
SIZE B			CODE IDENT NO. SEE SHEET 1	DRAWING NO. 710-00008-2		SHEET 19 OF 31
SCALE						

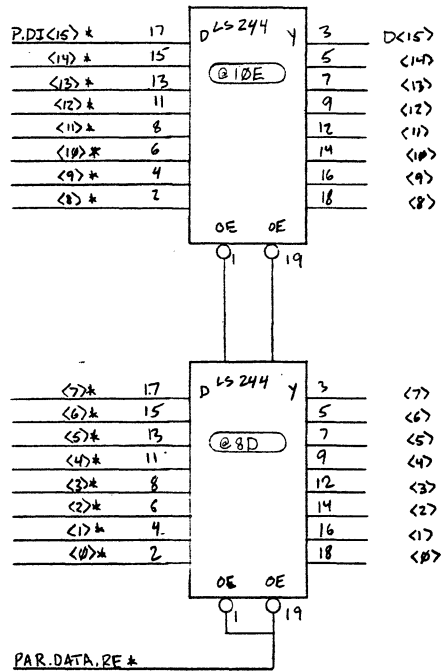
4

3

2

1

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		



D

C

B

A

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ±	CONTRACT NO.		PIB SCHEMATIC PARALLEL DATA INPUT REGISTER		
	APPROVALS	DATE			
MATERIAL	DRAWN TILLEMANN	10-27-82	SIZE B	CODE IDENT NO. SEE SHEET 1	DRAWING NO. 710-00008-2
FINISH	CHECKED		VALID LOGIC SYSTEMS INCORPORATED		SHEET 20 OF 31
DO NOT SCALE DRAWING	SCALE				

4

3

2

1

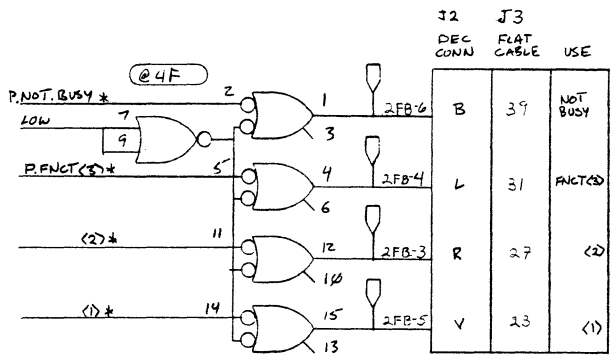
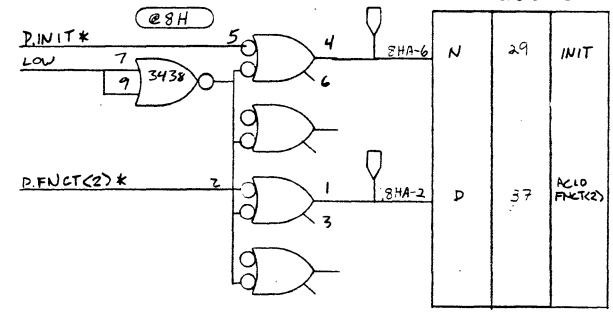
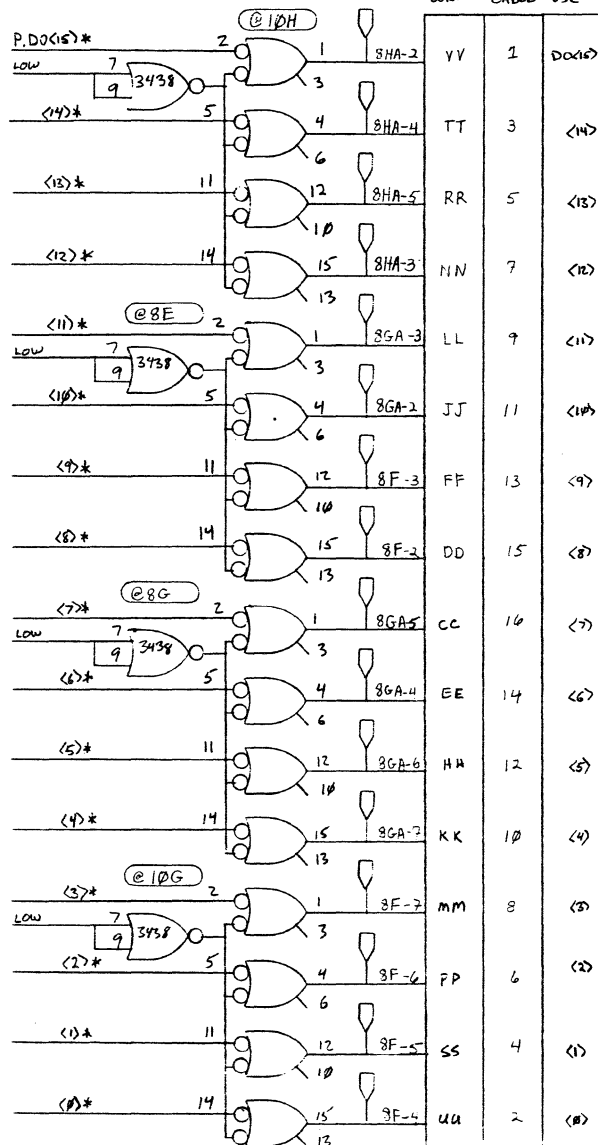
4

3

2

1

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		



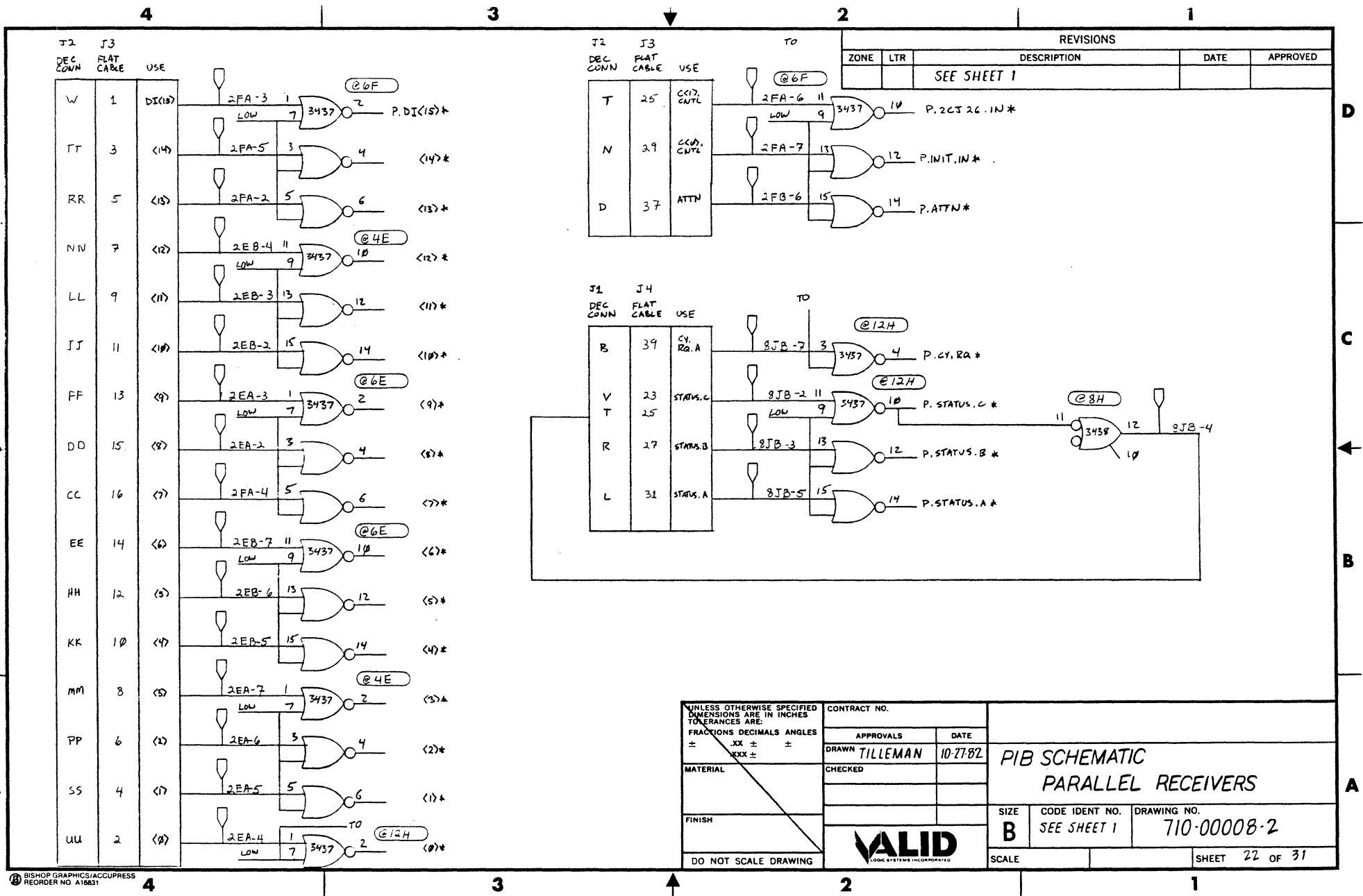
UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± XXX ±	CONTRACT NO.		PIB SCHEMATIC PARALLEL DRIVERS
	APPROVALS	DATE	
	DRAWN TILLEMANN	10-27-82	
	CHECKED		
MATERIAL			SIZE B
FINISH			
DO NOT SCALE DRAWING		VALID LOGIC SYSTEMS INCORPORATED	
		CODE IDENT NO. SEE SHEET 1	DRAWING NO. 710-00008-2
SCALE		SHEET 21 OF 31	

4

3

2

1



REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		

J2 DEC CONN	J3 FLAT CABLE	USE
T	25	CTL7. CNTL
N	29	CTL6. CNTL
D	37	ATTN

J1 DEC CONN	J4 FLAT CABLE	USE
B	39	CY. RA. A
V	23	STATUS.C
T	25	STATUS.C
R	27	STATUS.B
L	31	STATUS.A

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ±	CONTRACT NO.		PIB SCHEMATIC PARALLEL RECEIVERS
	APPROVALS	DATE	
	DRAWN TILLEMANN	10-27-82	
	CHECKED		
MATERIAL			SIZE B
FINISH			CODE IDENT NO. SEE SHEET 1
DO NOT SCALE DRAWING	VALID LOGIC SYSTEMS INCORPORATED		DRAWING NO. 710-00008-2
	SCALE		SHEET 22 OF 31

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		

J1 DEC CONN	J4 FLAT CABLE	USE	
K	32	BLK. PR	ØV
R	19	CYCLE RG. B	ØV
X	21	END CYCLE	ØV
J	33	WC. INC. ENABLE	ØV
F	35	READY	ØV

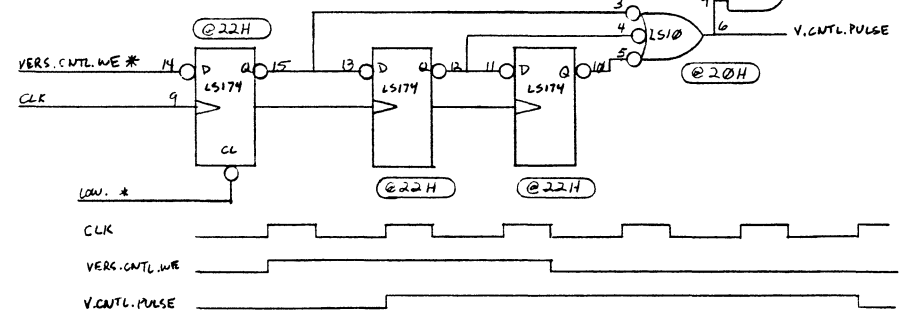
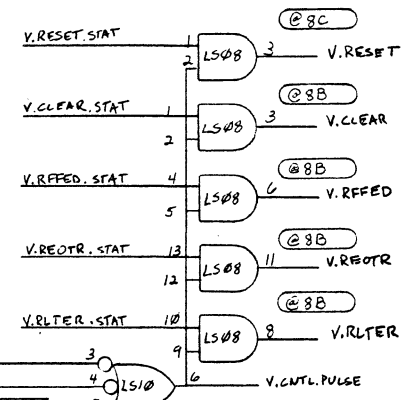
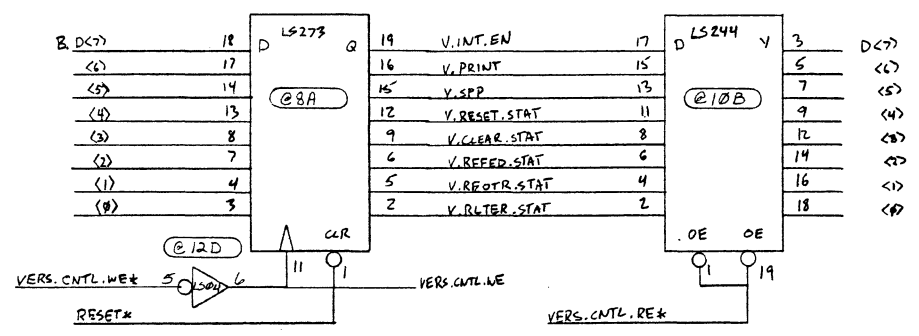
J1 DEC CONN	J4 FLAT CABLE	
AA	18	ØV
Y	20	
W	22	
U	24	
S	26	
P	28	
M	30	
H	34	
E	36	
C	38	
A	40	
BB	17	

J2 DEC CONN	J3 FLAT CABLE	
AA	18	ØV
Y	20	
W	22	
U	24	
S	26	
P	28	
M	30	
H	34	
E	36	
C	38	
A	40	
BB	17	
Z	19	

J2 DEC CONN	J3 FLAT CABLE	USE	
K	32	FNCT	ØV
X	21	GO	ØV
J	33	BANK ENB	
F	35	A(ØV)	ØV

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ± ±	CONTRACT NO.		PIB SCHEMATIC		
	APPROVALS	DATE			
	MATERIAL	DRAWN TILLEMANN	10-27-82	PARALLEL STATIC SIGNALS AND GROUNDS	
	FINISH	CHECKED			
DO NOT SCALE DRAWING	VALID LOGIC SYSTEMS INCORPORATED	SIZE B	CODE IDENT NO. SEE SHEET 1	DRAWING NO. 710-00008-2	
SCALE			SHEET 23 OF 31		

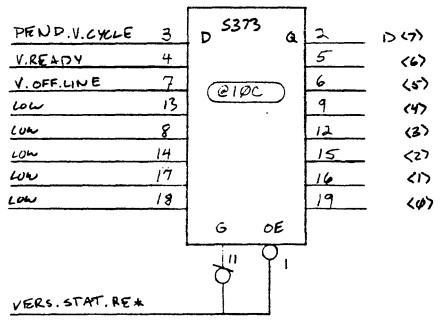
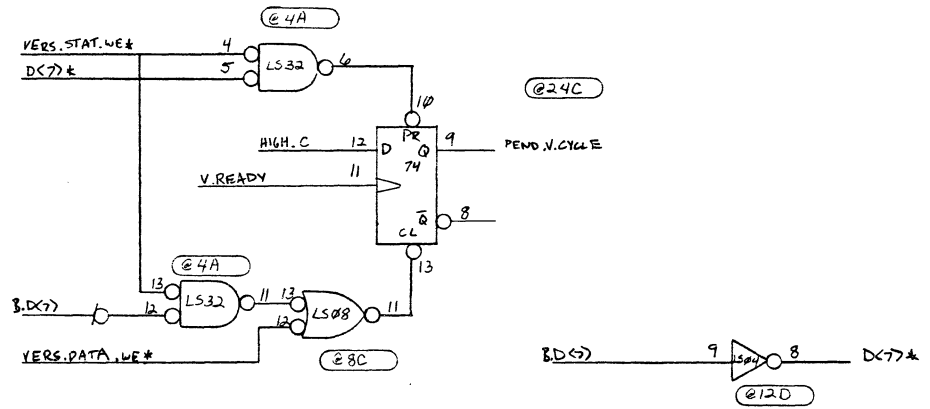
REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± .XXX ±	CONTRACT NO.		PIB SCHEMATIC VERSATEC CONTROL REGISTER
	APPROVALS	DATE	
MATERIAL	DRAWN TILLEMANN	10-27-82	SIZE B CODE IDENT NO. SEE SHEET 1 DRAWING NO. 710-00008-2
FINISH	CHECKED		
DO NOT SCALE DRAWING	VALID LOGIC SYSTEMS, INCORPORATED		SCALE
			SHEET 24 OF 31

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		

NOTE: TWO VERSIONS OF D<7>* EXIST IN WW VERSION



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ±	CONTRACT NO.		PIB SCHEMATIC VERSATEC STATUS REGISTER
	APPROVALS	DATE	
	DRAWN TILLEMANN	102782	
	CHECKED		
MATERIAL			SIZE B
FINISH			CODE IDENT NO. SEE SHEET 1
DO NOT SCALE DRAWING	VALID LOGIC SYSTEMS INCORPORATED		DRAWING NO. 710-00008-2
SCALE			SHEE. 25 OF 31

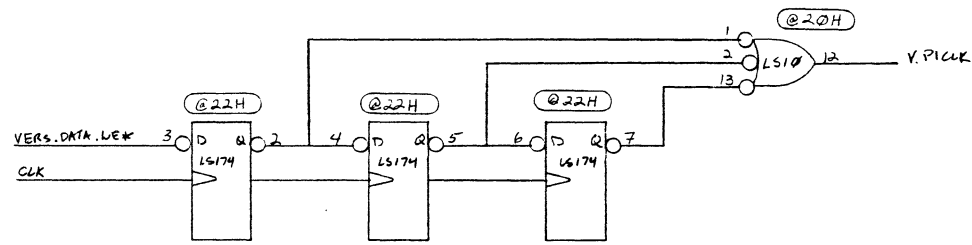
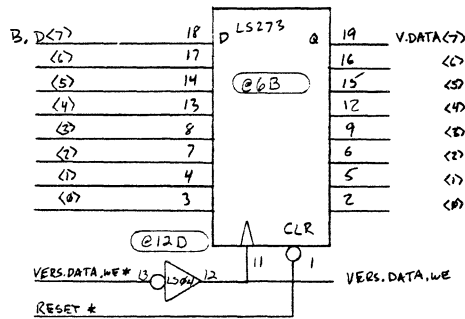
4

3

2

1

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ± ±	CONTRACT NO.		PIB SCHEMATIC VERSATEC DATA REGISTER
	APPROVALS	DATE	
	DRAWN TILLEMANN	10-27-82	
	CHECKED		
MATERIAL			SIZE B
FINISH			CODE IDENT NO. SEE SHEET 1
DO NOT SCALE DRAWING	VALID LOGIC SYSTEMS INCORPORATED		DRAWING NO. 710-00008-2
SCALE			SHEET 26 OF 31

4

3

2

1

4

3

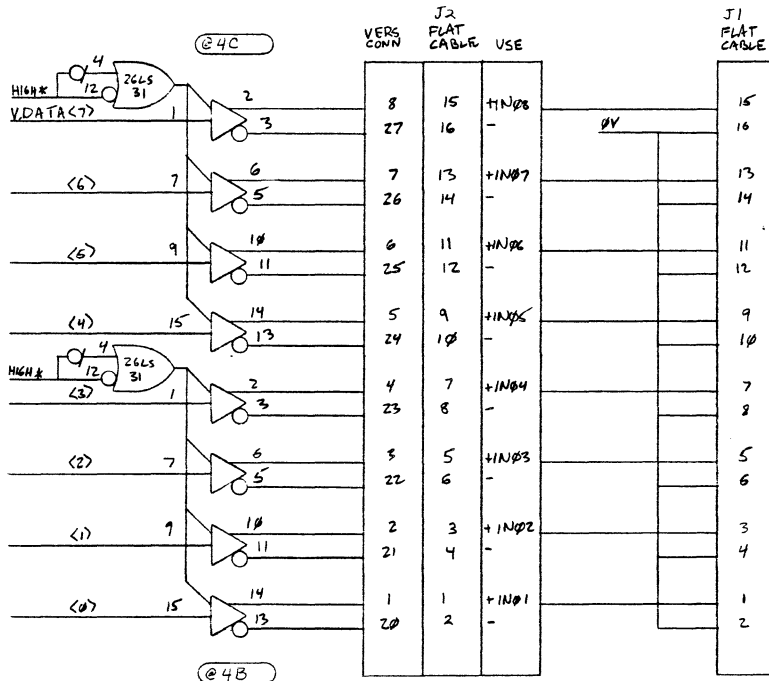
2

1

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		

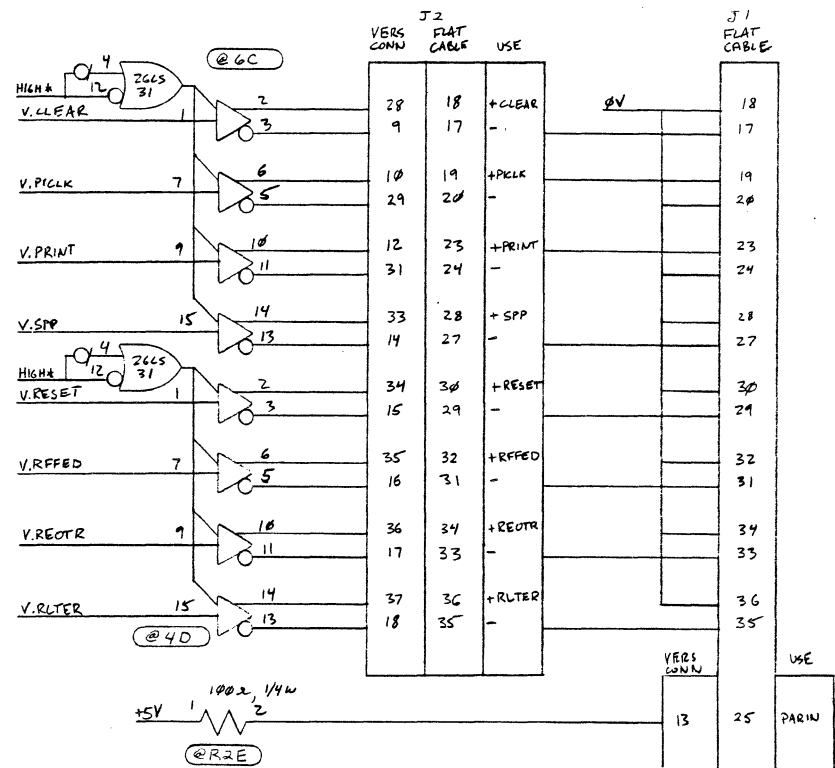
DIFFERENTIAL

SINGLE ENDED



DIFFERENTIAL

SINGLE ENDED



D

C

B

A

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ±	CONTRACT NO.		PIB SCHEMATIC VERSATEC DRIVERS
	APPROVALS	DATE	
	DRAWN TILLEMANN	10-27-82	
	CHECKED		
MATERIAL			SIZE B
FINISH			
DO NOT SCALE DRAWING	VALID LOGIC SYSTEMS CORPORATION		CODE IDENT NO. SEE SHEET 1
			DRAWING NO. 710-00008-2
SCALE			SHEET 27 OF 31

4

3

2

1

4

3

2

1

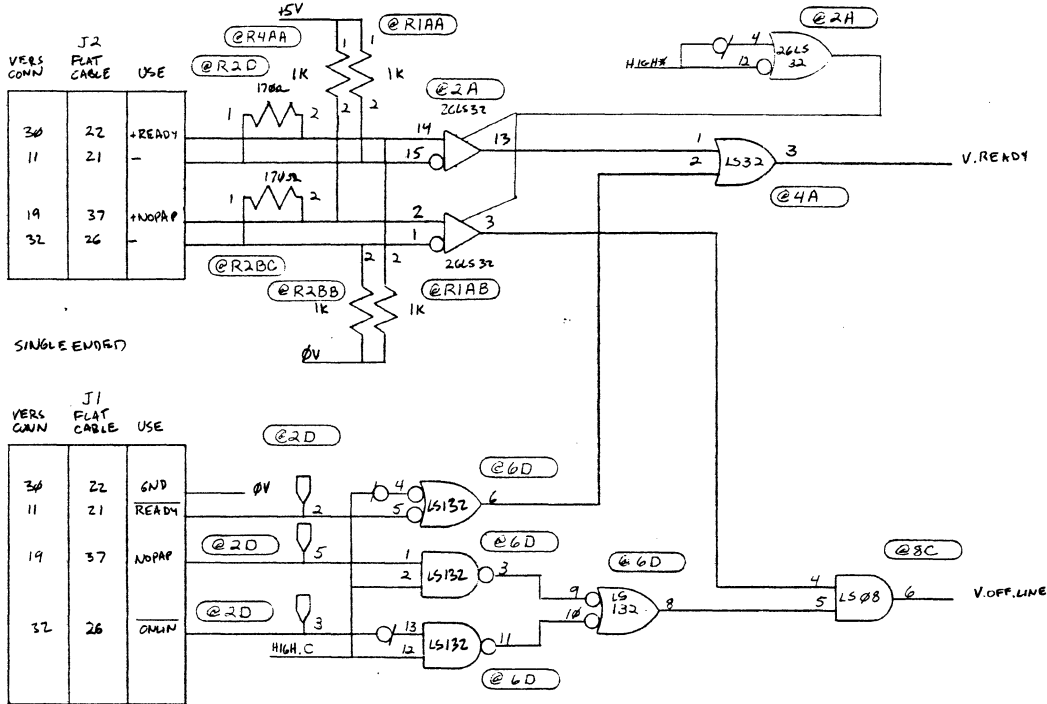
REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		

DIFFERENTIAL

VERS CONN	J2 FLAT CABLE	USE
30	22	+READY
11	21	-
19	37	+NOFAP
32	26	-

SINGLE ENDED

VERS CONN	J1 FLAT CABLE	USE
30	22	GND
11	21	READY
19	37	NOFAP
32	26	OMNIN



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ±	CONTRACT NO.		PIB SCHEMATIC VERSATEC RECEIVERS
	APPROVALS	DATE	
MATERIAL	CHECKED	DRAWN TILLEMANN	10-27-82
FINISH		SIZE B	CODE IDENT NO. SEE SHEET 1
DO NOT SCALE DRAWING		DRAWING NO. 710-00008-2	
		SCALE	SHEET 28 OF 31

4

3

2

1

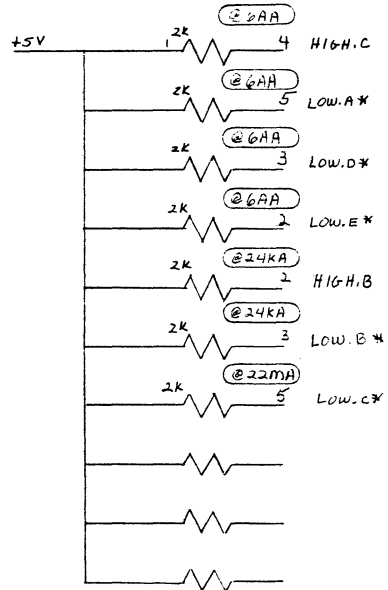
4

3

2

1

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
		SEE SHEET 1		



D

C

B

A

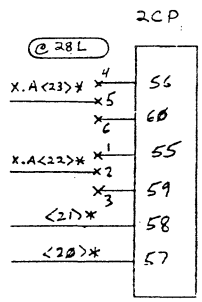
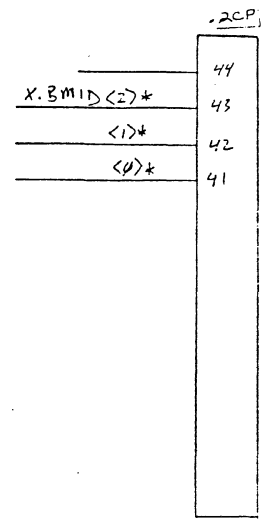
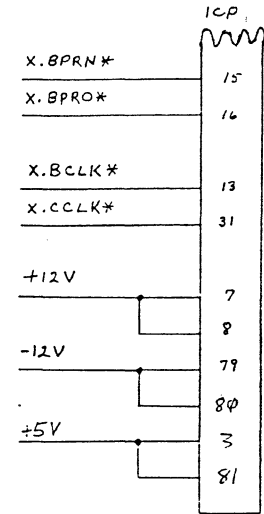
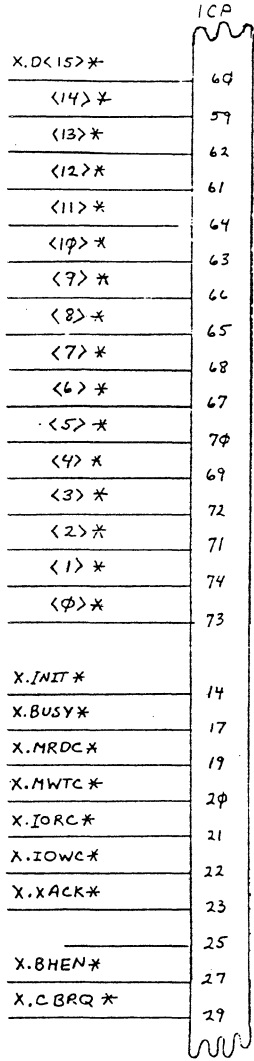
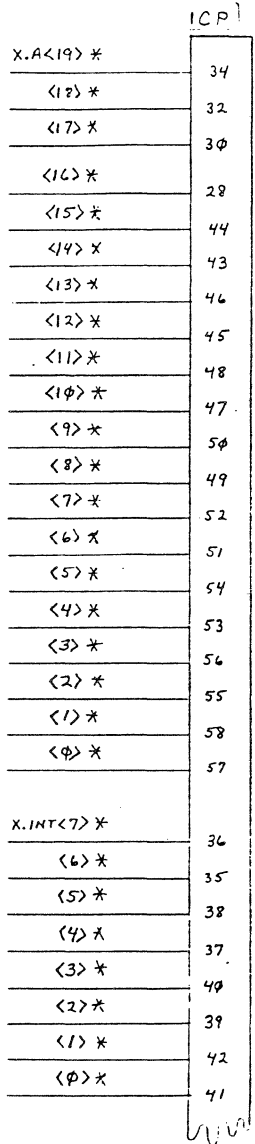
UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE FRACTIONS DECIMALS ANGLES ± .XX ± ± XXX ±	CONTRACT NO.		PIB SCHEMATIC LOW & HIGH GENERATION			
	APPROVALS					DATE
	DRAWN TILLEMANN		10-27-82	SIZE B CODE IDENT NO. SEE SHEET 1 DRAWING NO. 710-00008-2		
	CHECKED					
MATERIAL			SCALE			
FINISH						
DO NOT SCALE DRAWING				SHEET 29 OF 31		

4

3

2

1



VALID LOGIC SYSTEMS INC.

SCALE:	APPROVED BY:	DRAWN BY TILLEMAN
DATE: 10-27-87		REVISED
PIB SCHEMATIC		
796BUS CONNECTOR		DRAWING NUMBER B-710-00008-2

J7

16 W
FLAT
CABLE

CLK *	46	1
BCLK *	42	3
CS#1	38	5
WR	36	7
OWN. 796BUS	34	9
INTR	32	11
RESET *	30	13
A<18> *	28	15
TIMER.OVF	26	16
RDY	24	14
EXT.ACC	22	12
A<17>	20	10
A<16>	18	8
A<14>	16	6
ΦV	14	4
ΦV	12	2

VALID LOGIC SYSTEMS INC.		
SCALE:	APPROVED BY:	DRAWN BY TILLEMANN
DATE: 10-27-82		REVISED
PIB SCHEMATIC		
TEST CONNECTOR		DRAWING NUMBER B-710-00008-2

Valid Multibus Motherboard User Manual

VED-041582-4 Revision 7-02-82 (GSM)

02 Aug 84

Features

- Full support of the 16MBy IEEE-796 bus (Intel Multibus)
- 23 card slots
- Supports systems drawing up to 150 amperes
- Configurable for parallel or serial bus arbitration
- Strappable serial-priority bypass on each slot
- Any slot may be the head of the serial priority chain
- Shielded signal lines
- Delivers optional 4th power supply voltage
- Compatible with cards having injector/extractor handles

Copyright 1982

Valid Logic Systems Incorporated

This document contains confidential proprietary information which is not to be disclosed to unauthorized persons without the written consent of an officer of Valid Logic Systems Incorporated.

The copyright notice appearing above is included to provide statutory protection in the event of unauthorized or unintentional public disclosure.

1.0 INTRODUCTION

The Valid Logic Systems Multibus backplane (mother board) is designed to connect up to 23 Multibus compatible cards. The mother board has a number of design features that allow it to:

- Support either the serial or parallel Multibus bus arbitration protocols. If the serial protocol is used, a jumper scheme allows empty or non-bus-master card slots to be "bypassed" without additional wiring. Furthermore, any slot can be strapped to be the head of the bus priority chain.
- Several non-standard signals are configured on the Valid mother board to allow the support of high performance systems. These include:

Busing of all P2 signals.

Distribution of -5.2 volts (for high performance ECL systems) on P2 pins.

2.0 MOTHER BOARD CONSTRUCTION

The mother board is approximately 16.45" x 13.29" and 0.770" thick. The mother board has twenty three socket positions numbered one to twenty three in decreasing bus priority (when using the serial arbitration scheme). All socket positions are identical except socket 1 which has pin 15 (BPRN*) hard-wired to GND. (This ensures that BPRN* is always asserted on socket one.)

The board is a four layer PC. The layers are all two ounce copper. From top (socket side) to bottom the layers are:

- Stripe plane - this layer carries some signal wiring, board nomenclature (such as copyright notices etc.), power contact pads, and two nickel plated stripes against which copper bus bars carrying ground and Vcc (+5) are bolted.
- Vcc plane - this layer carries Vcc (+5) to pins 3,4,5,6,81,82,83,84 of all P1 sockets and connects to the Vcc stripe on the Vcc plane. This layer also carries Vee (-5) to pins 33, 34, 35 and 36 on all P2 sockets and connects to the Vee pad on the stripe plane.
- Signal plane - this layer contains most of the signal wiring. It also carries +12 from the +12 pad on the stripe plane to pins 7 and 8 of all P1 connectors, -12 from the -12 pad on the stripe plane to pins 79 and 80

of all P1 sockets, and -5 from the Vcc plane to pins 9 and 10 on all P1 sockets.

- Ground plane - this layer carries GND (0V) to pins 1,2,11,12,75,76,85,86 of all P1 sockets and connects to the GND stripe on the stripe plane.

2.1 Power Considerations

All five mother board power pads are fitted with 8-32 brass studs for attaching power cables. Internal copper conductors and the bus bars have been sized for minimal voltage drop. Each of the five power connections has different current capacities:

VOLTAGE	CURRENT	DROP (mv)
Vcc (+5)	150	25
Gnd (+0)	170	25
Vee (-5)	5	25
+12 volts	5	100
-12 volts	5	100

The rated voltage drops are measured from the appropriate power connection stud, to the appropriate contact on socket number 1 (socket furthest from the power feed studs) and make no assumptions on the distribution of the loads on the backplane.

2.2 Electrostatic Noise Protection Features

To provide some measure of immunity from ESD two structures have been built into the mother board. First, a ground trace is run between each signal wire on the P1 connectors. These ground traces are connected to ground between each socket. Second, a DC isolated trace about 0.5 inches wide runs around the periphery of the mother board on the ground plane. This square annulus connects to chassis ground along its entire length. The Gnd plane (ground return) of the mother board is connected to this "earth annulus" via a 0.005uF RF capacitor in each corner of the mother board. This provides low impedance, small loop size paths to ground for ESD transients.

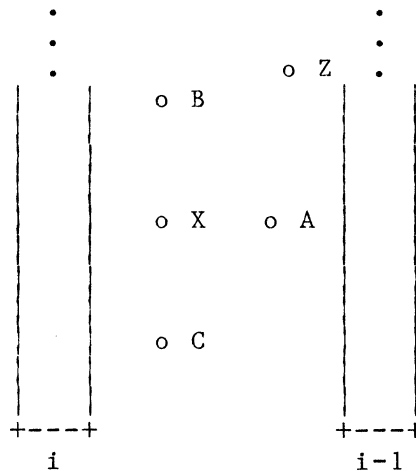
To reduce coupling noise, signal traces have been placed on both the signal layer and stripe layer. The distribution of signals was chosen so that signals which undergo transitions at different times are on opposite sides of the mother board. All signals are on the signal plane except those listed below which are on the stripe plane:

BUSY* (pin 17).
XACK* (pin 23).

LOCK* (pin 25).
 CCLK* (pin 31).
 INT<7..0>* (pins 35-42).
 DAT<15..0>* (pins 59-74).
 ADR<15..12>* (pins 43-46).

2.3 Bus Priority Jumpers

The Valid mother board has incorporated wiring and a jumper arrangement that supports both the serial and parallel bus arbitration schemes. This is done with a system of five wirewrap pins just to the left of each socket position, except J23 (the following description assumes that the mother board is being viewed from the socket side, with J1 on the right and J23 on the left):



- If pin X is jumpered to A:
 Then BPRN* of socket i is driven by BPRO* of socket i-1.
- If pin X is jumpered to B:
 Then BPRN* of socket i is connected to BPRN* of socket i-1.
- If pin X is jumpered to C:
 Then BPRN* of socket i is connected to GND.
- BPRN* of socket 1 is connected to GND.
- Pin Z is connected to BREQ* of socket i-1.

There are two intended uses of this jumper arrangement.

- For the serial arbitration scheme:
 1. The first card in the priority resolution chain is placed in socket 1 OR has a jumper installed from X to C to the right of its socket.
 2. Every card slot to the left of a bus-master card is jumpered X to A.
 3. Every card slot to the left of a non-bus-master card is jumpered X to B.

- For the parallel arbitration scheme:
 1. Sockets 1 and 23 must not have bus-master type cards installed.
 2. Pin Z to the left of a socket, and Pin X to the right of socket i comprise a BREQ*/BPRN* pair that is wired to the parallel bus arbiter.

APPENDIX A

MOTHER BOARD MECHANICAL DRAWING

Valid Bus Analyzer Board User Manual

VED-041582-3 Revision 4-21-82 (RWT)

04 May 82

Features

- Standard IEEE-796 bus (Intel Multibus) slave
- Resistively terminates bus signals
- High-performance termination values available
- Termination values configurable to Multibus standard
- Bus history buffer records 1024 bus cycles
- Buffer is fully diagnosable
- Generates interrupt on buffer almost full
- Bus activity detector restarts hung systems
- Buffers power-fail and front-panel reset signals

Copyright 1982

Valid Logic Systems Incorporated

This document contains confidential proprietary information which is not to be disclosed to unauthorized persons without the written consent of an officer of Valid Logic Systems Incorporated.

The copyright notice appearing above is included to provide statutory protection in the event of unauthorized or unintentional public disclosure.

Copy Number _____

1.0 INTRODUCTION

The Valid Logic Systems Bus Analyzer (BA) board is intended to serve four purposes.

First, it applies resistive termination to bus signals. The terminated signals are a superset of the Multibus definition. In order to allow Valid systems to run at high speed, some signals are terminated on the BA board using non-Multibus-standard values. These values may be downgraded to the Multibus standard by selectively depopulating the board.

Second, the Bus Analyzer Board contains a Bus History Buffer which records bus cycles as they occur. The buffer holds 1024 cycles of information (upgradable to 4096), retaining the most recent bus cycles in a circular queue. The buffer can be programmed to record or ignore each of the four types of Multibus cycles: Memory Read, Memory Write, I/O Read and I/O Write. The buffer may optionally record accesses to only a 64KBy window of Multibus memory space. The buffer may also be programmed to generate a Multibus interrupt after recording a selected number of bus cycles.

Third, the Bus Analyzer Board contains circuitry to receive and buffer the manual reset signal generated by an external SPDT switch, and the power-fail signal output by the power supply. These signals generate a Multibus reset and interrupt, respectively.

Fourth, this board detects long periods (greater than 3 seconds) of inactivity on the Multibus. Such periods may indicate the failure of some central portion of the system, and the Bus Analyzer Board may be strapped to reset the system automatically when they occur.

2.0 TERMINATION

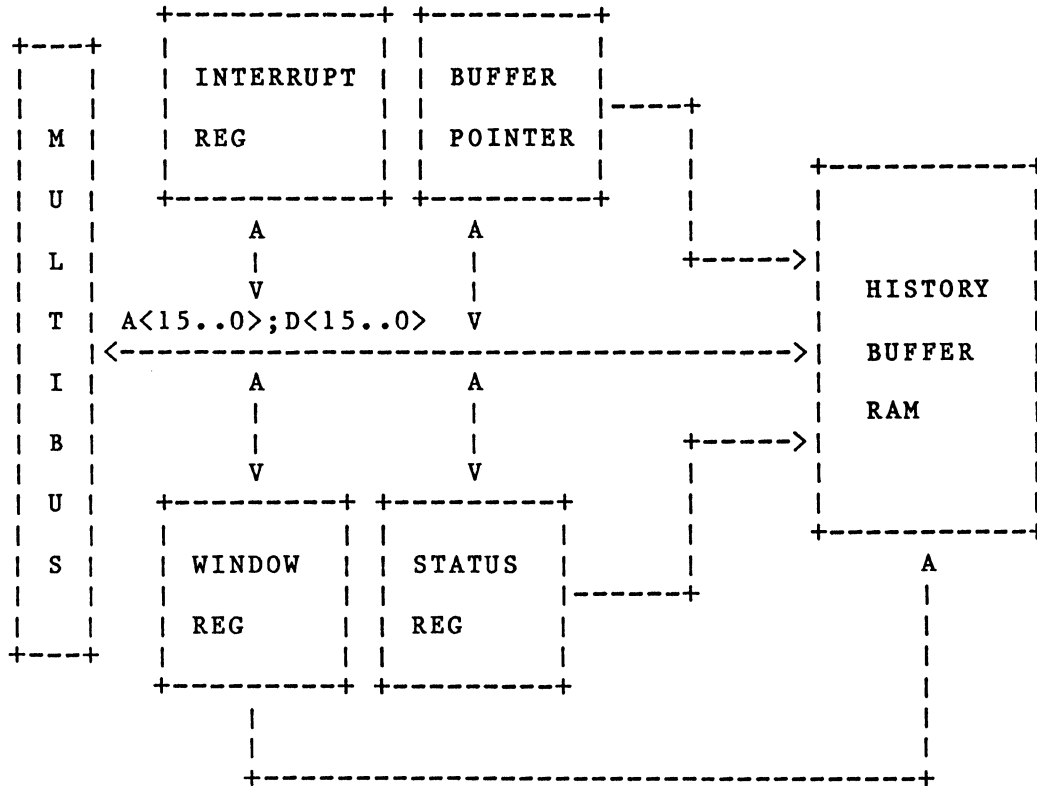
The following is a list of Multibus signals and both their Multibus standard termination values, and their Valid termination values, if the two are different. Those Valid terminators marked as "||" are parallel resistors added to the standard terminators. These parallel resistors are discrete components and may be omitted or removed to achieve standard Multibus termination. In general, address and data signals are pulled up through 2.2K Ohms, tri-state and slow open-collector control signals are pulled up through 1.0K Ohms, fast open-collector control signals are pulled up through 510 Ohms, and clocks are pulled up through 220 Ohms and down through 75 Ohms.

Signals	Termination vaules:	
	Standard	Valid
P1 signals		
D<15..0>*	2.2K	
A<19..0>*	2.2K	
BHEN*	2.2K	
MRDC*	1.0K	
MWTC*	1.0	
IORC*	1.0K	
IOWC*	1.0K	
XACK*	510	150
LOCK* (P1.AACK*)	1.0K	180
INH1*	1.0K	
INH2*	1.0K	
BCLK*	220/330	220/75
BREQ*		
BPRO*		
BPRN*		
BUSY*	1.0K	390
CBRQ*	1.0K	390
INIT*	2.2K	270
CCLK*	220/330	220/75
INTA*	1.0K	
INT<7..0>*	1.0K	
P2 signals		
PFSR*	1.0K	
PFSN*	1.0K	
ACLO*	1.0K	
PFIN*	1.0K	
MPRO*	1.0K	
ARST*		
P2.AACK*		150
A<23..20>*	2.2K	
BMID<3..0>*		510
AS*		1.0K
(all others)		1.0K

Many signals on the P2 connector are undefined in the Multibus standard, and are currently unused in Valid products. These signals are all based on the Valid Motherboard, and have all been pulled up through 1.0K Ohms on the BA board.

3.0 BUS ANALYZER BOARD ARCHITECTURE

The Bus Analyzer Board has internal data and address buses with the following devices attached:



The Bus Analyzer Board occupies 64KBy of Multibus memory space. The 64KBy area in which the board resides is selectable by setting an 8-bit DIP switch on the desired upper address bits. Closed switches represent 1's and open switches represent 0's. Within this window, A<15..0> select on-board devices as follows:

DEVICE	A<15..0>	BYTES USED
BUFFER RAM	0000..1FFF	BOTH
BUFFER POINTER	8000	BOTH
WINDOW REG	A000	LOW-ORDER
STATUS REG	C000	LOW-ORDER
INTERRUPT REG	E000	LOW-ORDER

4.0 STATUS REGISTER

The status register is an 8-bit read/write register which configures the board and controls diagnostic LED's. The bits in the Status register are allocated as follows:

SIGNAL	BIT POS IN BYTE
RED.LED	7
GREEN.LED	6
EN.INT	5
USE.WINDOW	4
EN.MRDC	3
EN.MWTC	2
EN.IORC	1
EN.IOWC	0

- RED.LED lights the red diagnostic LED if and only if it is set. This bit is set automatically after approximately 3 seconds of Multibus inactivity, and is not cleared by Auto-Resets. If the Bus Analyzer Board is strapped to generate Auto-Resets, then after reset, this bit will be a one if and only if the reset was an Auto-Reset.
- GREEN.LED lights the green diagnostic LED if and only if it is set.
- EN.INT enables a strap-selectable Multibus interrupt to occur whenever the high order bit of the buffer pointer becomes a one.
- USE.WINDOW set configures the buffer to only record cycles which have the upper 8 address bits the same as contents of the window register. When clear, the buffer will record based only on the type of bus cycle.
- EN.MRDC configures the buffer to record Memory Read cycles if and only if it is set.
- EN.MWTC configures the buffer to record Memory Write cycles if and only if it is set.
- EN.IORC configures the buffer to record I/O Read cycles if and only if it is set.
- EN.IOWC configures the buffer to record I/O Write cycles if and only if it is set.

The Status register is cleared during reset.

5.0 INTERRUPT REGISTER

The interrupt register contains the interrupt request bits. The upper 6 bits always read as 0. The low order 2 bits are read write, and are automatically set by interrupting devices.

SIGNAL	BIT POS IN BYTE
(zero)	7
(zero)	6
(zero)	5
(zero)	4
(zero)	3
(zero)	2
P.F.INT	1
BUFFER.INT.REQ	0

- P.F.INT is set whenever a power-fail signal has been received from the power supply. This signal is cleared on reset.
- BUFFER.INT.REQ is set whenever the upper-most bit of the buffer pointer register changes from a 0 to a 1.

6.0 WINDOW REGISTER

The window register is an 8-bit read/write register which is cleared on reset. If status register bit USE.WINDOW is set, then record operations will be restricted to that section of Multibus memory having the upper-most 8 address bits equal to the value of the window register.

7.0 BUFFER RAM

This is the memory into which data from recorded bus cycles is stored. The size of this memory is normally 8Kby, corresponding to 1024 bus cycles. If larger RAM's are used, 32Kby of memory, corresponding to 4096 bus cycles, may be obtained.

8.0 BUFFER POINTER

This word register stores the index of the next bus cycle to be recorded. The low-order 12 bits are read-write and are the variable part of the pointer. The high-order 4 bits always read as 0. Each time a cycle is recorded, the value of the buffer pointer is increased by 1, modulo 4096. When the high-order bit of the variable portion (2048's bit) changes from a 0 to a 1, a buffer interrupt may be generated.

9.0 BUS HISTORY BUFFER

The Bus History Buffer must perform two operations: it must record selected bus cycles (in "record cycles"), and it must allow a Multibus master to read out the recorded data (in "interrogate cycles") For testing purposes, the buffer is also writable by a bus master. The buffer monitors 60 Multibus signals, and stores them in a 64x1024/16x4096 word dual-port ram. Bus cycles are stored into the memory as sequential 64-bit words. During interrogate cycles, data is read out as 4 16-bit words, or 8 8-bit bytes, per entry; address bits A<2..1> combine as follows to select the 16-bit word written or read:

A<2..1>: BYTE

		BIT								
		7	6	5	4	3	2	1	0	
0:H		D<15>	D<14>	D<13>	D<12>	D<11>	D<10>	D<9>	D<8>	
0:L		D<7>	D<6>	D<5>	D<4>	D<3>	D<2>	D<1>	D<0>	
1:H		A<15>	A<14>	A<13>	A<12>	A<11>	A<10>	A<9>	A<8>	
1:L		A<7>	A<6>	A<5>	A<4>	A<3>	A<2>	A<1>	A<0>	
2:H		XACK	AACK	BHEN	AS	BM<3>	BM<2>	BM<1>	BM<0>	
2:L		A<23>	A<22>	A<21>	A<20>	A<19>	A<18>	A<17>	A<16>	
3:H		INT<7>	INT<6>	INT<5>	INT<4>	INT<3>	INT<2>	INT<1>	INT<0>	
3:L		--	--	--	--	MRDC	MTWC	IORC	IOWC	

Multibus accesses to the board are not recorded in the Bus History Buffer. The unused bits in the buffer may be interrogated in the same manner as the the used bits, but 0's are stored in those bits during record cycles. The Bus History Buffer supports full Multibus word and byte operations. A<0> and BHEN select which byte

or bytes are operated on as follows:

BYTE(S) USED	A<0>	BHEN
(invalid)	1	1
high-order	1	0
full word	0	1
low-order	0	0

The address of the next 64-bit entry to be written (containing the oldest data in the buffer) is stored in the Buffer Pointer, and may be read or written through word or byte read or write operations. The pointer is 12 bits wide, allowing expansion to 4096 64-bit words of storage. The unused upper 4 bits of the pointer word read as 0's. To enable the CPU to detect the buffer becoming almost full, the high order bit of the buffer pointer generates a Multibus interrupt if it is a one and the EN.INT bit is set. The CPU can set the starting location in the buffer to any initial value, so software can determine the fraction of the buffer to be filled before an interrupt occurs. When the appropriate number of cycles have been recorded, an interrupt will be generated. The contents of the Buffer Pointer are undefined at power-up time.

The Bus Master identification bits BM<3..0> are extra signals on the P2 connector which indicate which bus master owned the bus during a cycle. If each bus master in a multi-master system drives these bits to a different value when it owns the bus, then diagnostic programs can determine which master initiated each cycle. These signals are present on the following pins:

SIGNAL	P2 PIN
BM<3>	44
BM<2>	43
BM<1>	42
BM<0>	41

10.0 RESET AND POWER-FAIL

This board receives and buffers the Manual Reset and Power-Fail signals. The Manual Reset signal is derived from a break-before-make SPDT switch external to the board. The switch is debounced and buffered onto the Multibus INIT line. The Power-Fail signal may be an active-high or active-low, open-collector or standard TTL signal generated by the power supply; it indicates an impending loss of power. This signal is buffered onto a strap-selectable Multibus interrupt line.

11.0 BUS-TIMEOUT RESET

The Bus-Timeout Reset circuit is designed to detect long periods of inactivity on the Multibus, and to generate a system reset when one occurs. The circuit samples the four Multibus command strobes; if none of them change for approximately 3 seconds, then it generates a Multibus INIT pulse lasting approximately 3 seconds. The RED.LED bit is also set and remains set after reset. This allows system software to read this bit after a reset to determine whether an auto-reset has occurred.

12.0 PROGRAMMING AND USE

In any computer system, and especially in a multiprocessor system, infrequent and irreproducible software errors may occur which are very difficult to diagnose through conventional means. To help detect and troubleshoot these errors, the Bus History Buffer records the most recent bus cycles, providing a trace of selected Multibus accesses. Since some of these errors might be associated only with a certain type of bus cycle, Memory Writes for instance, the buffer is programmable to record or ignore each of the four types of bus cycles: Memory Read, Memory Write, I/O Read and I/O Write.

12.1 Sample Use 1

A typical mode of operation of the Bus History Buffer is to configure the buffer to record all cycles and, when an error occurs, to disable all recording and read out the most recent 1024 bus cycles for examination. The program which reads out the entries then formats them in a text file which can be manipulated by user programs.

To read out the contents of the buffer, a program first reads the Buffer Pointer register to determine the address of the oldest entry in the buffer. Since one 64-bit entry in the buffer is accessed as 4 16-bit words or 8 8-bit bytes, the pointer must be shifted left 3 places (multiplied by 8) to obtain the Multibus address of the low-order byte or word of the entry.

The program then reads out all entries up to the end of the buffer (at 1FFF) using normal Multibus byte or word operations. Next, the program starts at the beginning of the buffer, and reads the entries up to the pointer address. This process reads out the entries ordered from oldest to newest.

12.2 Sample Use 2

Another typical mode of operation is intended for performance monitoring or detecting errors which do not become apparent until long after they occur. In this mode, the EN.INT flag is set, enabling an interrupt to occur whenever the high-order bit of the Buffer Pointer (bit 11) becomes a 1. The CPU sets the pointer to 512 entries before the upper bit would become a 1, which would be a value of 0E00h. The CPU then enables the appropriate bus cycles, and begins execution of the suspect program.

When 512 bus cycles have been recorded, an interrupt occurs. The CPU then disables the recording of all bus cycles and reads the cycles which have been recorded, possibly storing the data in a disk file. The CPU then repeats the above process, returning to the suspect program where it left off. When the error is finally detected, the CPU converts the bus cycle data to a text file usable by user programs. The entire trace history of the program has then been recorded, and the designer may diagnose and correct the error.

12.3 Sample Use 3

A third mode utilizes the selectable 64KBy recording window by setting the window to a common communication area used by several processors for semaphores and data pointers. This is done by setting the window register to the desired value, then setting the USE.WINDOW bit in the status register. This mode uses the continuous recording technique described above, but only records references to the selected 64KBy segment. This allows the system to run at nearly full speed, the CPU being interrupted only when several hundred accesses have been made to the segment in question. Running at nearly full speed maximizes the chance of detecting timing-dependent errors.

12.4 Physical Vs Virtual Addresses

Addresses recorded by the Bus History Buffer are physical, not virtual addresses. When tracing accesses by bus masters having memory mapping, the memory mapping function must be considered. If the contents of the memory map are known at the time of each recorded cycle, then the virtual addresses may be reconstructed from the physical addresses. One way to determine any changes made to the map is to record and decode the bus cycles which caused the changes.

13.0 MAINTENANCE

The Bus Analyzer Board has several features which improve its testability. First, all RAM in the history buffer is readable and writable by the Multibus. This allows standard memory diagnostics to test the buffer.

Second, the buffer pointer is readable and writable by the Multibus, so the CPU may load the buffer pointer with a given value, and read it back to check it. Then the CPU may enable monitoring of Memory Writes and do one Multibus write. This advances the pointer by one, and the CPU may then read out the contents of the pointer to test for correct operation of the counter. This can be done throughout the entire range of the counter.

Third, the Bus Analyzer Board has a red and a green diagnostic LED, which may be set by the CPU to indicate the status of the board during testing.

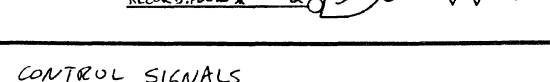
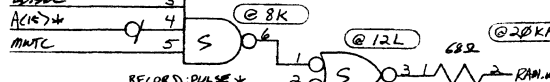
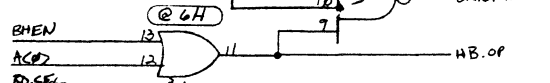
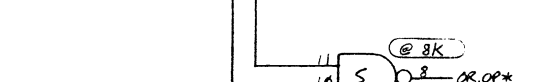
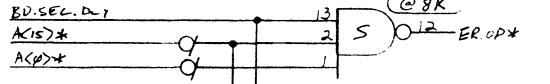
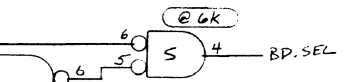
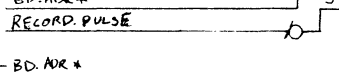
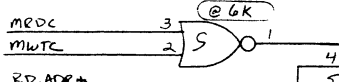
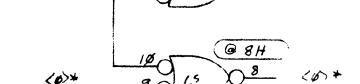
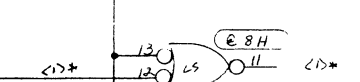
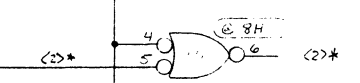
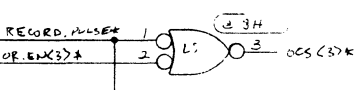
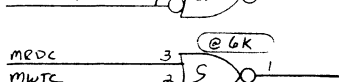
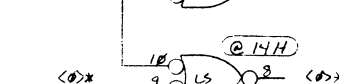
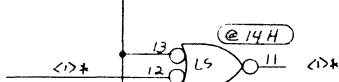
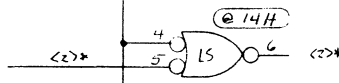
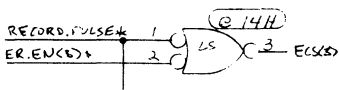
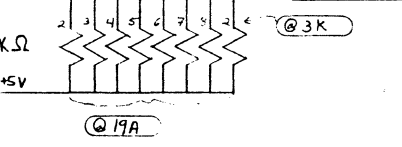
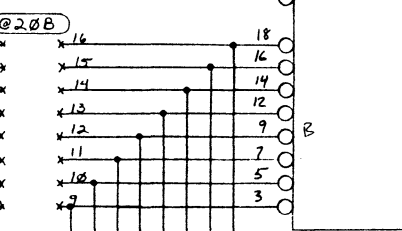
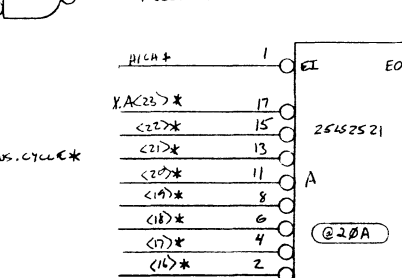
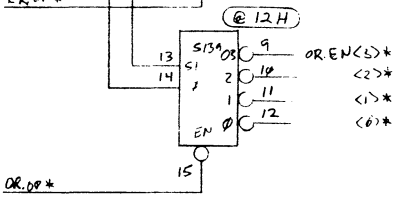
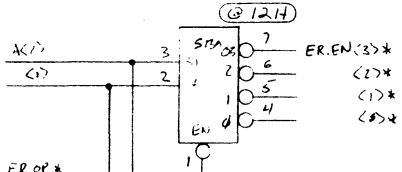
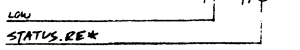
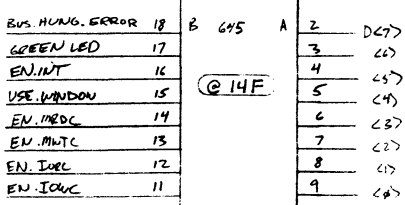
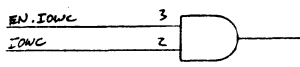
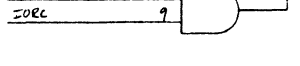
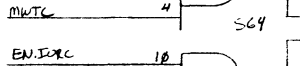
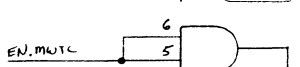
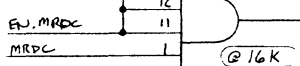
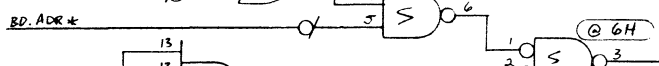
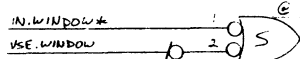
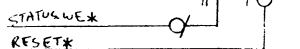
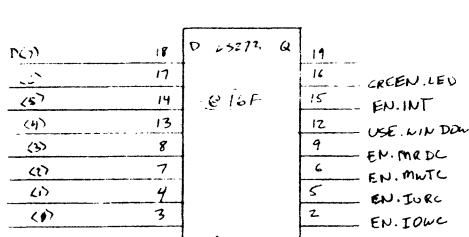
Abstract:

Describes the full strapping options for the Valid BA Board, wire-wrap version, including the default strap configuration.

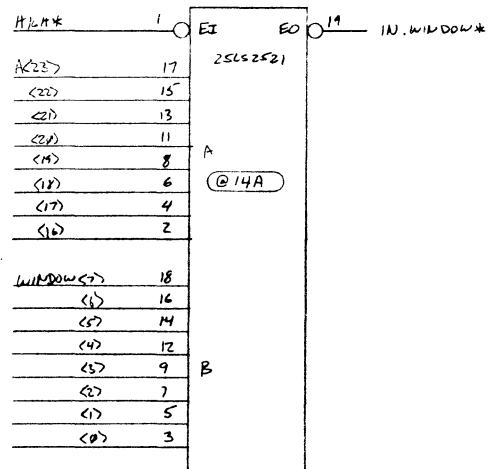
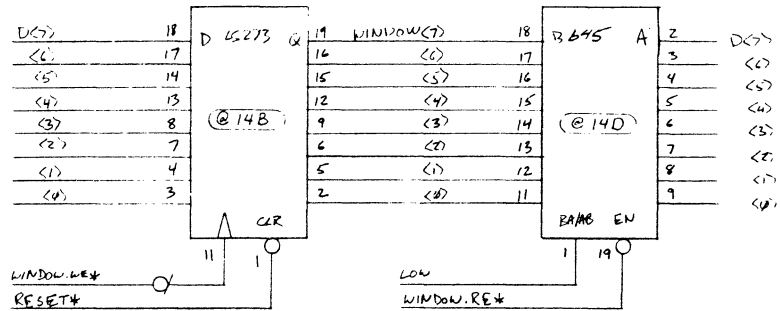
Notes:

1. Straps marked with "X" are connected by traces on the PC board.
2. Straps marked with "*" are connected using blue w/w wire during assembly.
3. Each "=" represents a single possible connection from the one pin on the left to any one of the pins on the right.

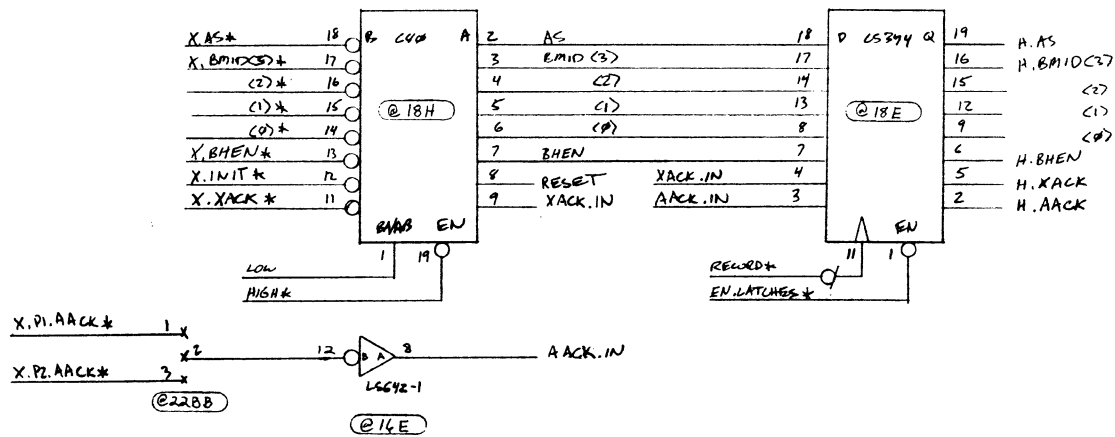
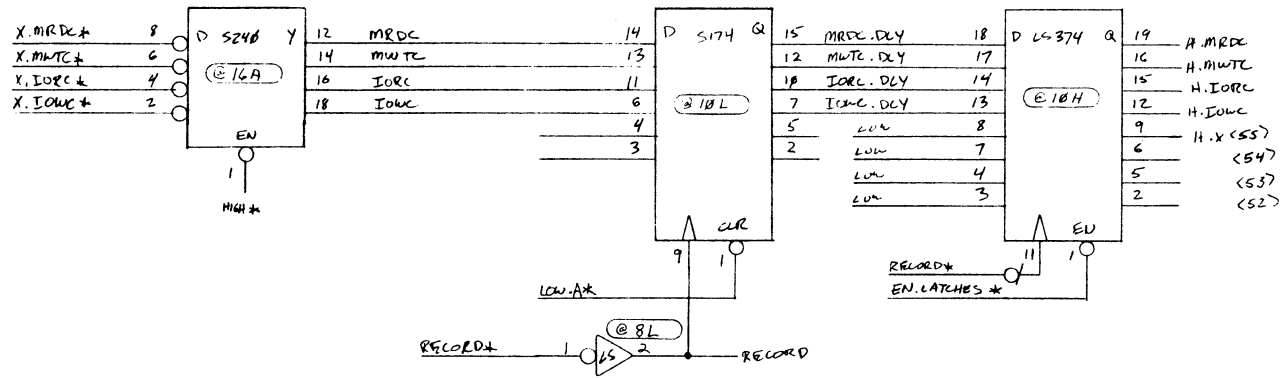
SHT	LOC	STRAP	DFLT	DESCRIPTION
3	3CJ13	2 = 1 3	*	RECEIVE AACK ON P1-25 RECEIVE AACK ON P2-40
10	49A	1 = 43AA8 43AA7 43AA6 43AA5 43AA4 43AA3 43AA2 43AA1		POWER-FAIL INTERRUPT ON MB LEVEL 0 POWER-FAIL INTERRUPT ON MB LEVEL 1 POWER-FAIL INTERRUPT ON MB LEVEL 2 POWER-FAIL INTERRUPT ON MB LEVEL 3 POWER-FAIL INTERRUPT ON MB LEVEL 4 POWER-FAIL INTERRUPT ON MB LEVEL 5 POWER-FAIL INTERRUPT ON MB LEVEL 6 POWER-FAIL INTERRUPT ON MB LEVEL 7
10	49A	2 = 43AA8 43AA7 43AA6 43AA5 43AA4 43AA3 43AA2 43AA1		BUFFER-FULL INTERRUPT ON MB LEVEL 0 BUFFER-FULL INTERRUPT ON MB LEVEL 1 BUFFER-FULL INTERRUPT ON MB LEVEL 2 BUFFER-FULL INTERRUPT ON MB LEVEL 3 BUFFER-FULL INTERRUPT ON MB LEVEL 4 BUFFER-FULL INTERRUPT ON MB LEVEL 5 BUFFER-FULL INTERRUPT ON MB LEVEL 6 BUFFER-FULL INTERRUPT ON MB LEVEL 7
11	3CJ4	2 = 1 3	*	LOW-ASSERTED POWER-FAIL INPUT HIGH-ASSERTED POWER-FAIL INPUT
12	43A	2 = 1		TIMEOUT RESETS MULTIBUS
13	3CJ1	2 = 1 3	*	SUPPLY POWER FOR 2148 RAMS SUPPLY ADDRESS BIT FOR 4148 RAMS
21	3CJ16	2 = 1 3	*	RECEIVE A<23> ON P2-56 REVEIVE A<23> ON P2-60



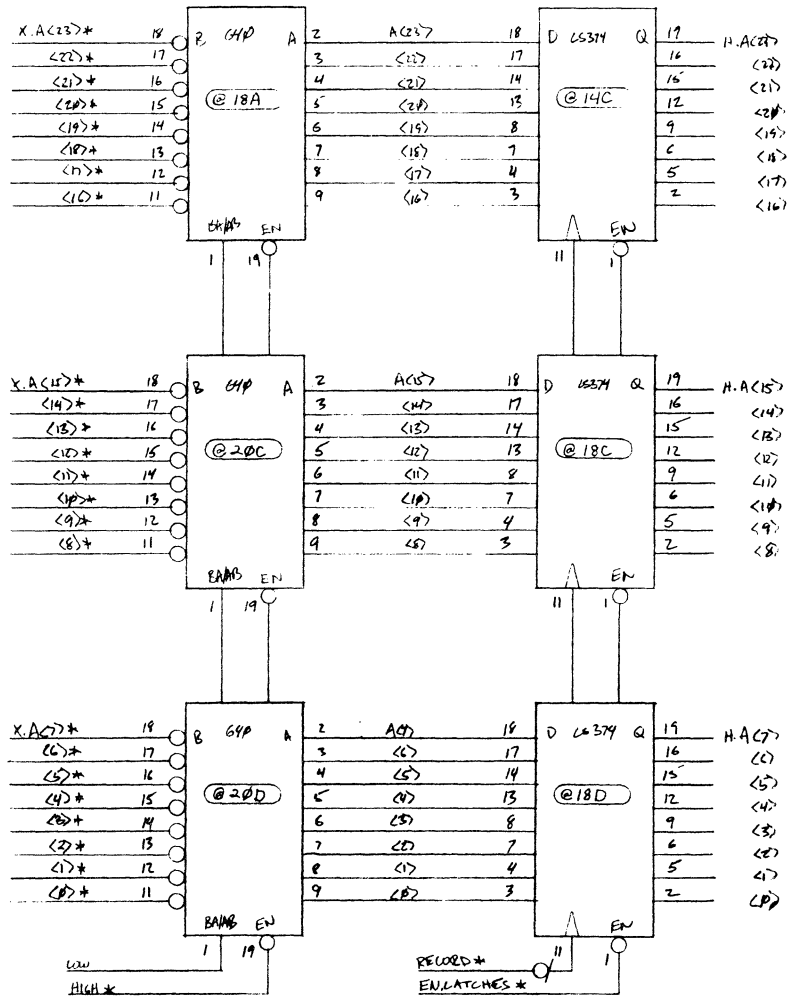
BAB CONTROL SIGNALS		
SCALE:	APPROVED BY:	DRAWN BY
DATE:		REVISED
		DRAWING NUMBER
		1 of 21



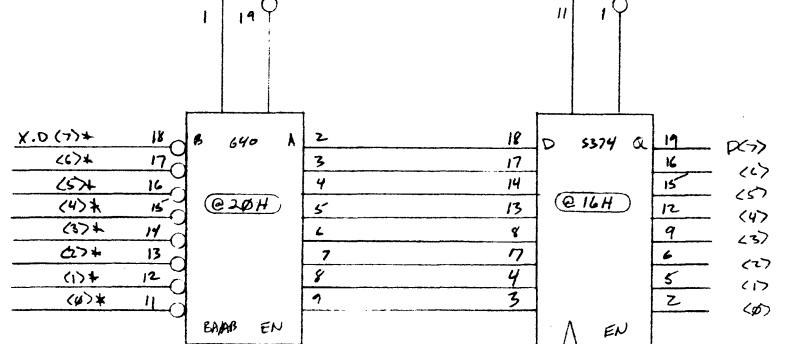
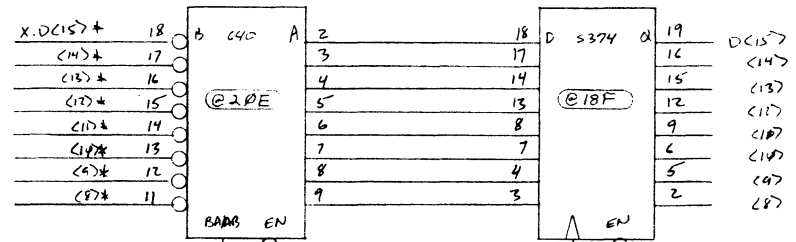
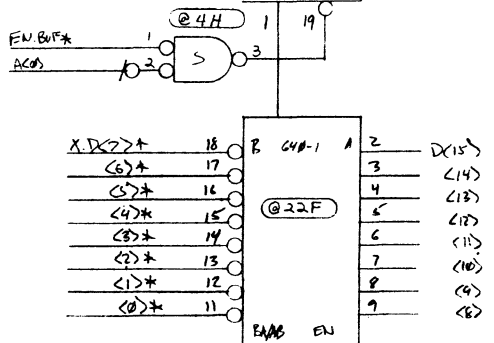
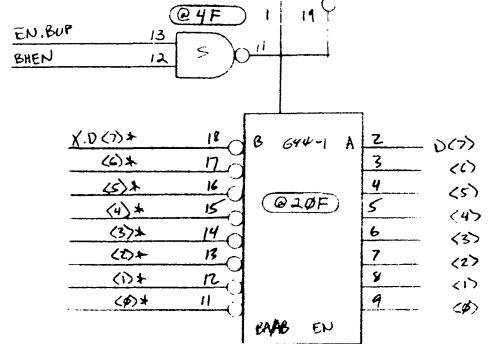
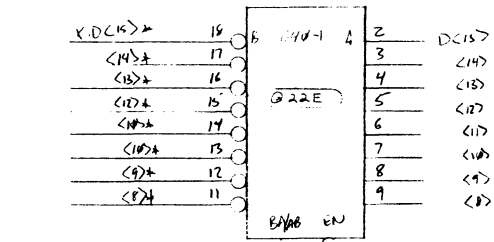
BAB WINDOW SELECT		
SCALE:	APPROVED BY:	DRAWN BY
DATE:		REVISED
		DRAWING NUMBER
		2 of 21



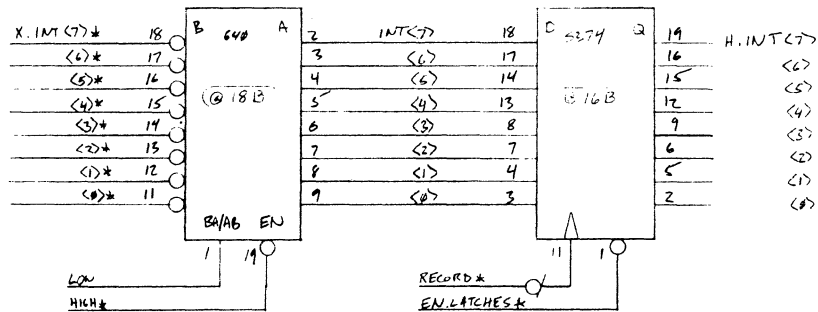
BAB CONTROL SIGNAL BUFFERS		
SCALE:	APPROVED BY:	DRAWN BY
DATE:		REVISED
		DRAWING NUMBER
		3 of 21



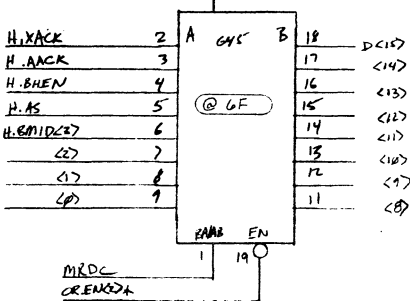
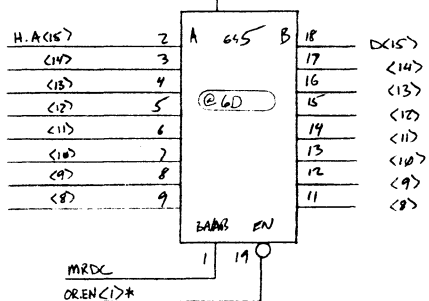
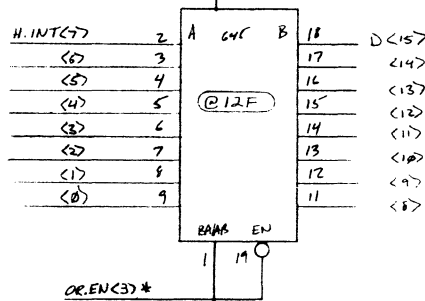
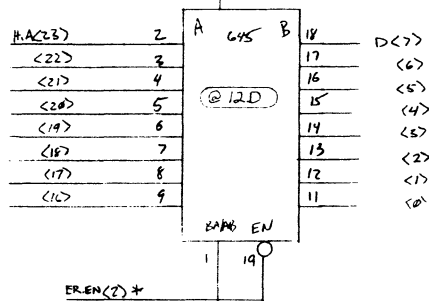
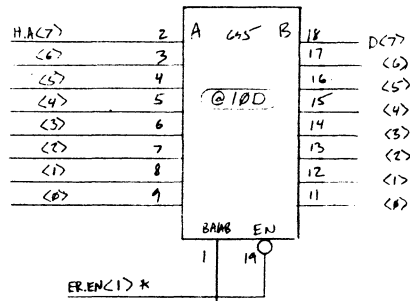
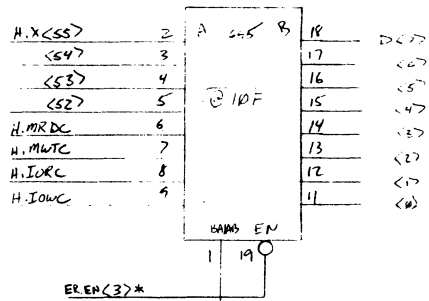
BAB ADDRESS BUFFERS		
SCALE:	APPROVED BY:	DRAWN BY
DATE:		REVISED
		DRAWING NUMBER
		4 of 21



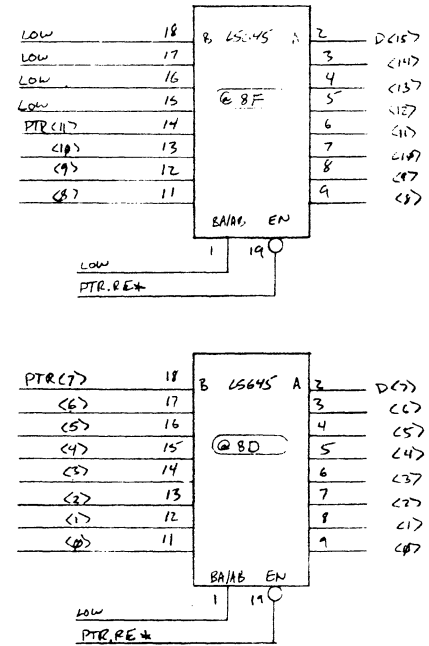
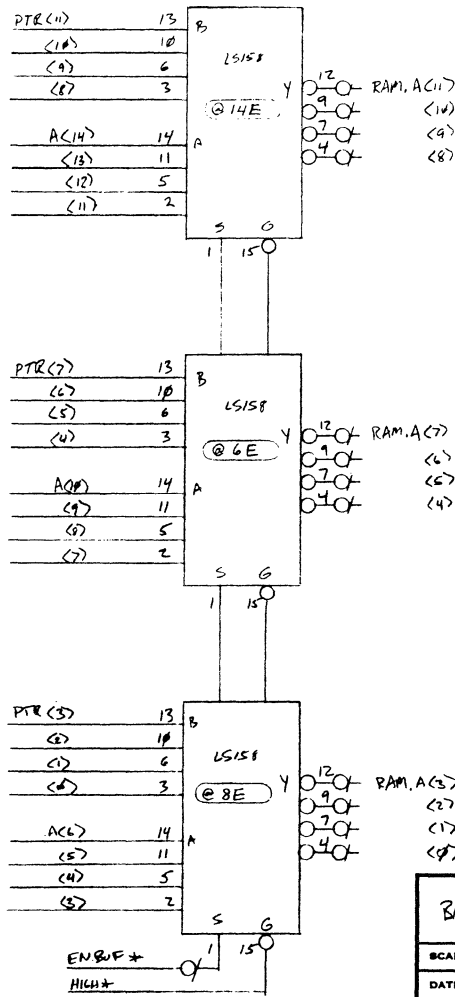
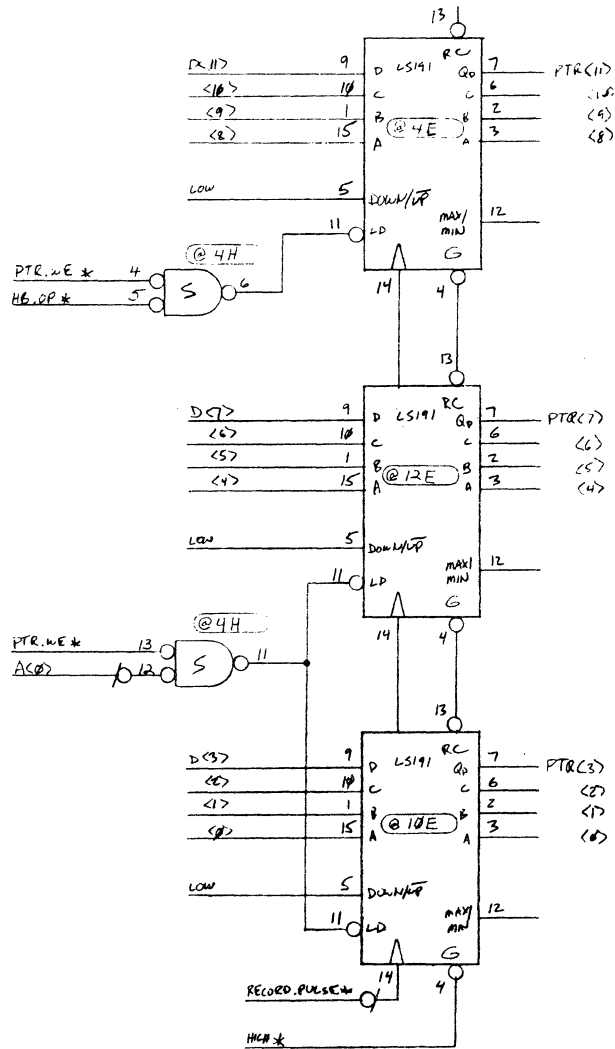
BAB DATA BUFFERS		
SCALE:	APPROVED BY:	DRAWN BY
DATE:		REVISED
		DRAWING NUMBER
		5 of 21



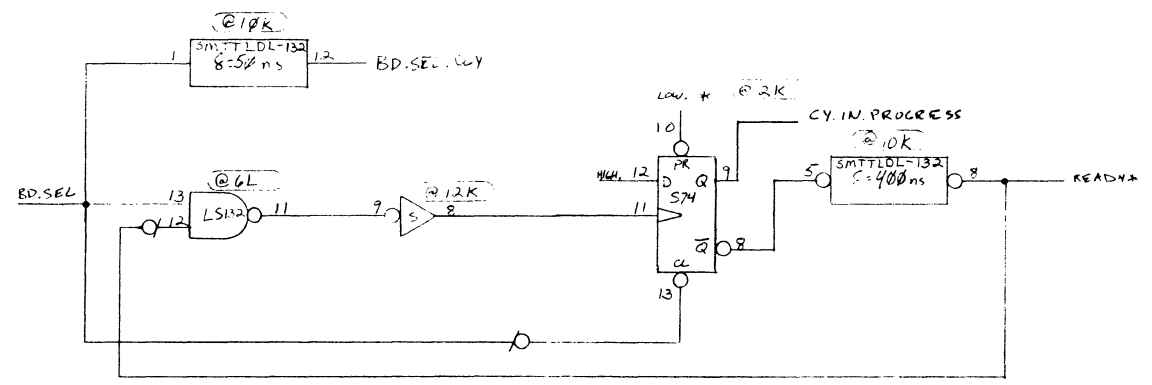
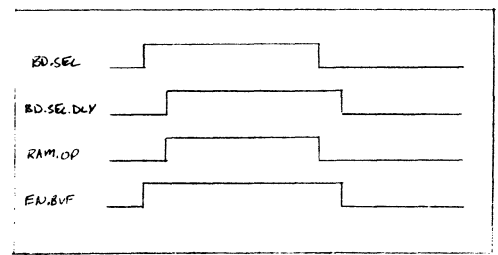
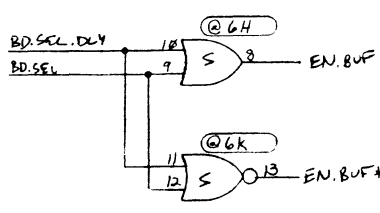
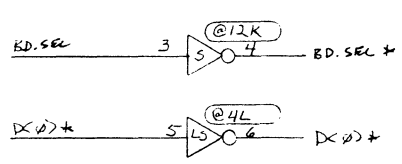
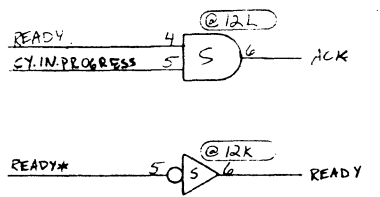
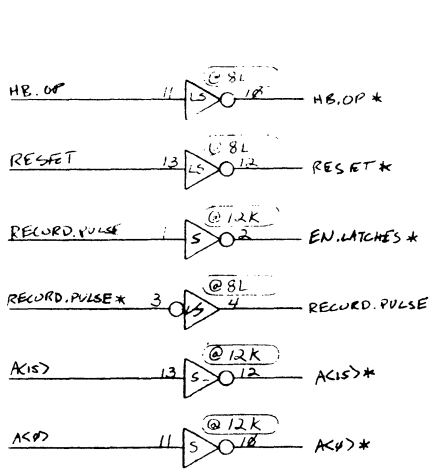
BAB INTERRUPT BUFFERS		
SCALE:	APPROVED BY:	DRAWN BY
DATE:		REVISED
		DRAWING NUMBER
		6 of 21



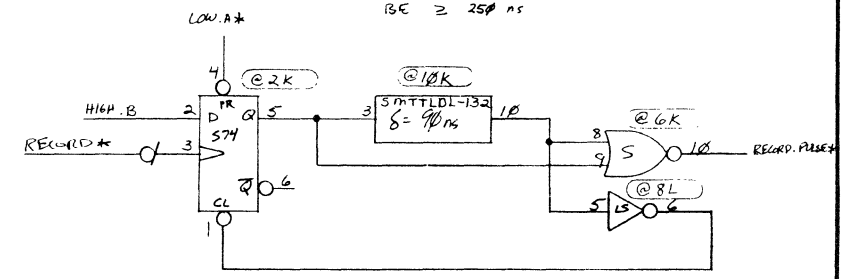
BAB HISTORY DATA BUFFERS		
SCALE:	APPROVED BY:	DRAWN BY
DATE:		REVISED
		DRAWING NUMBER
		7 of 21



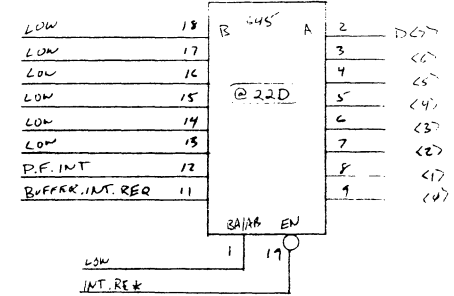
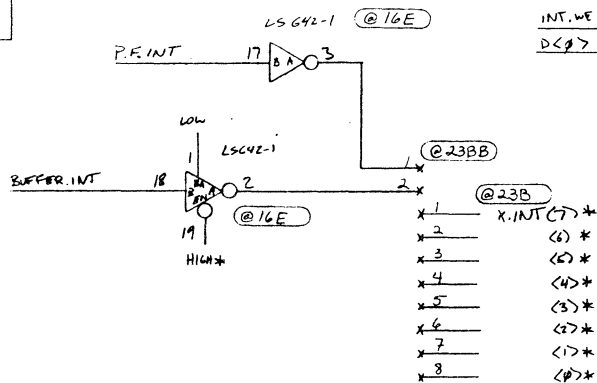
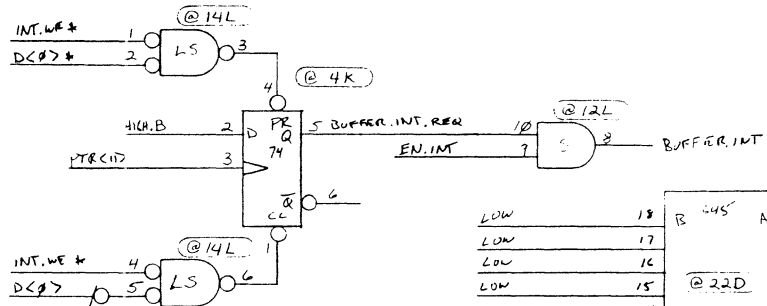
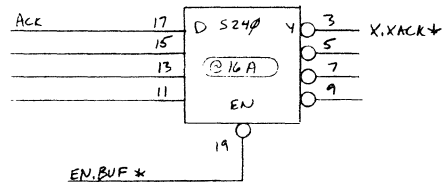
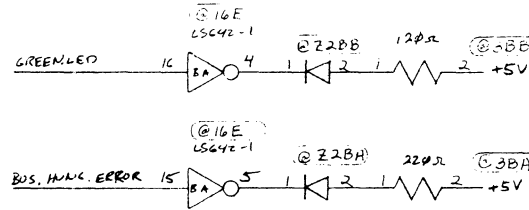
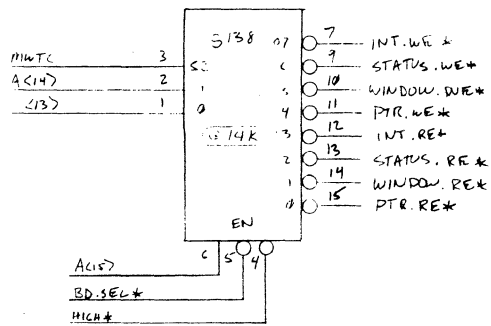
BAB BUFFER POINTER		
SCALE:	APPROVED BY:	DRAWN BY:
DATE:		REVISED:
		DRAWING NUMBER
		8 of 21



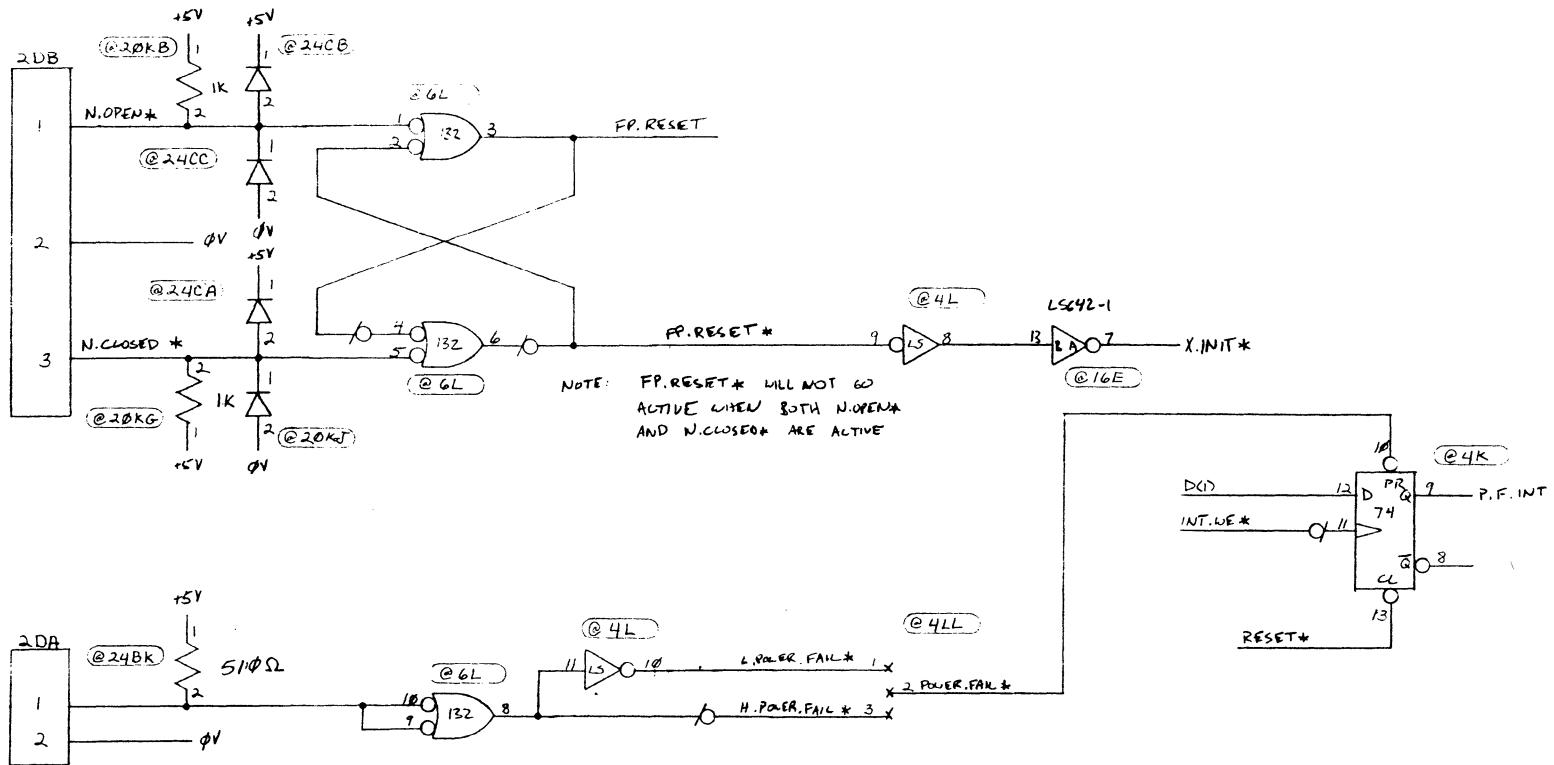
NOTE: PERIOD OF BUS CYCLES MUST BE ≥ 250 ns



BIB TIMING		
SCALE:	APPROVED BY:	DRAWN BY
DATE:		REVISED
		DRAWING NUMBER
		9 of 21

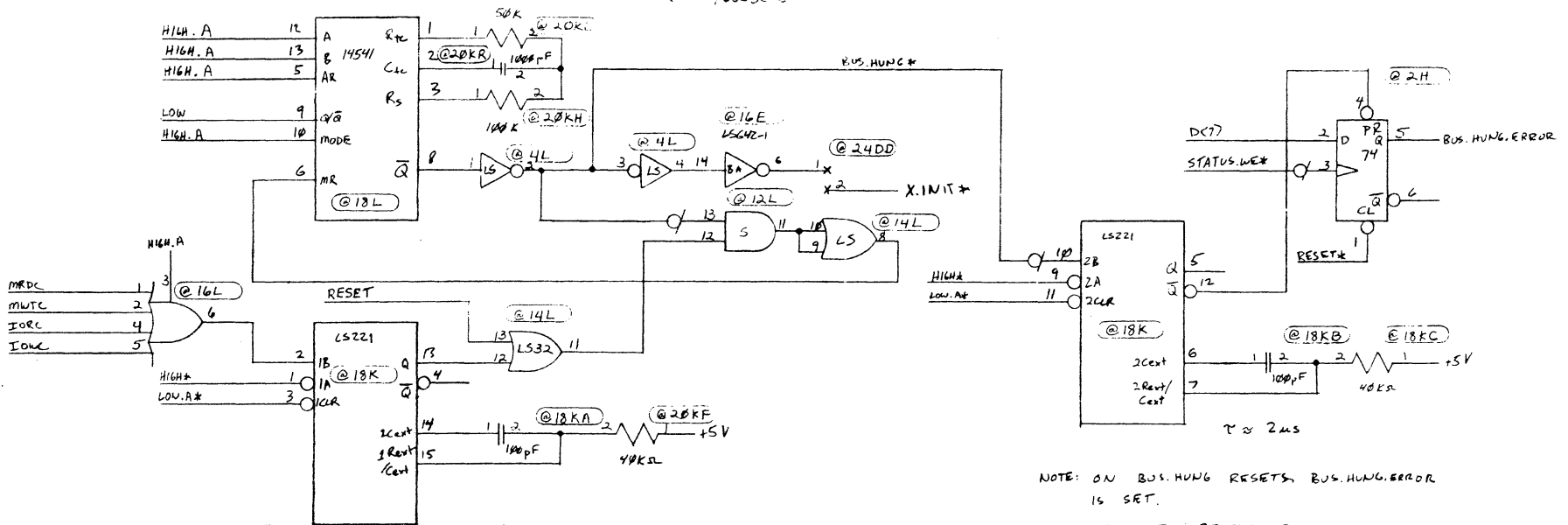


BAB CONTROL SIGNALS AND LED'S		
SCALE:	APPROVED BY:	DRAWN BY
DATE:		REVISED
		DRAWING NUMBER
		18 of 21



P.F. POWER FAIL AND MANUAL RESET		
SCALE:	APPROVED BY:	DRAWN BY:
DATE:		REVISED:
		DRAWING NUMBER
		11 of 21

Note $f \approx 10\text{KHz}$,
 $f_Q = 1/65536 \approx .16\text{ Hz}$



NOTE: GENERATES ~ 2ms PULSE STARTING AT THE BEGINNING OF EACH BUS CYCLE.
 EACH PULSE CLEARS 3 SEC TIMER.
 ERRONEOUS PULSES ARE LOCKED OUT DURING RESET.

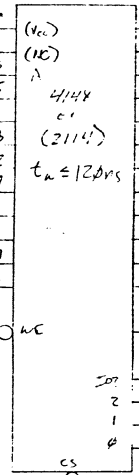
NOTE: ON BUS HUNG RESETS, BUS HUNG ERROR IS SET.
 ON SYSTEM RESETS, BUS HUNG ERROR IS CLEARED.

BAB BUS TIMEOUT RESET		
SCALE:	APPROVED BY:	DRAWN BY
DATE:		REVISED
		DRAWING NUMBER
		12 of 21

+5V

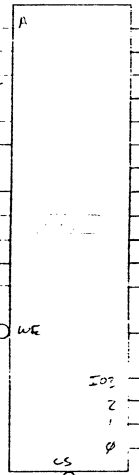
- <A> 1
- <A> 6
- <A> 5
- <A> 4
- <A> 3
- <A> 2
- <A> 7
- <A> 8
- <A> 18
- <A> 17
- <A> 16

RAM WE# 11



- IO3 12 D<1>
- 2 13 D<2>
- 1 14 D<3>
- ∅ 15 D<4>

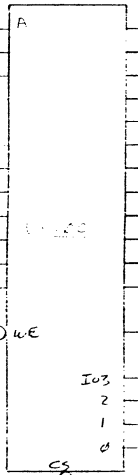
OCSCD X



- IO3 12 D<1>
- 2 13 D<2>
- 1 14 D<3>
- ∅ 15 D<4>

CS

ECSCD X



- IO3 12 D<1>
- 2 13 D<2>
- 1 14 D<3>
- ∅ 15 D<4>

CS

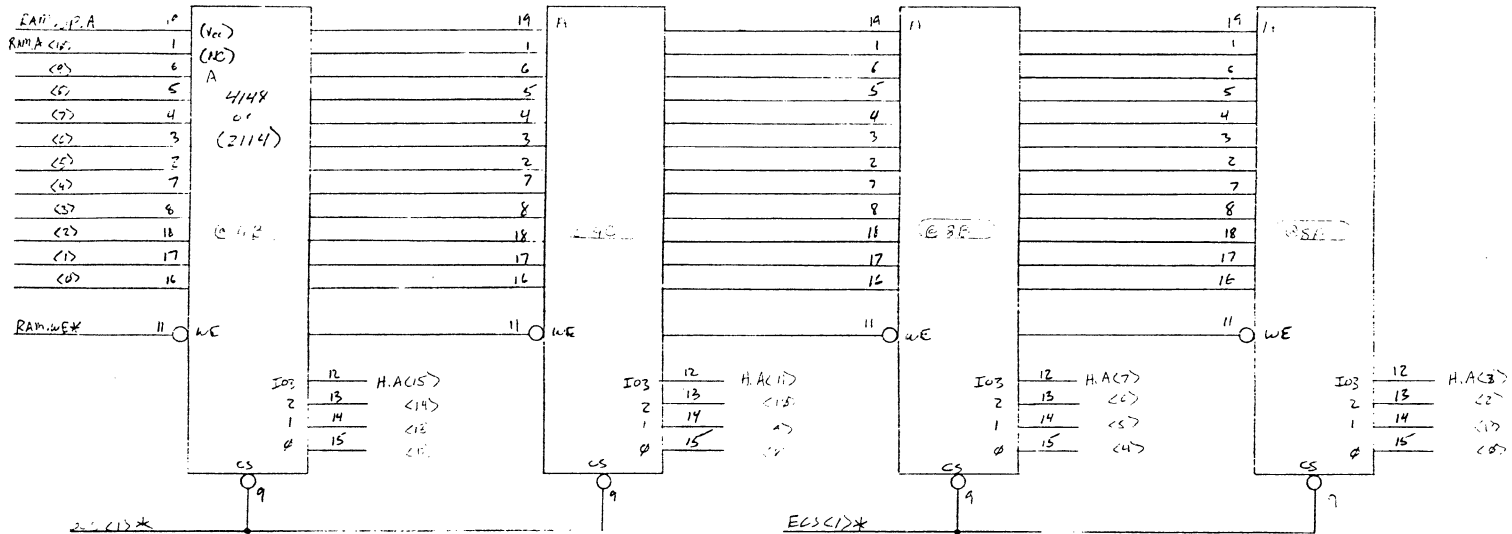


- IO3 12 D<1>
- 2 13 D<2>
- 1 14 D<3>
- ∅ 15 D<4>

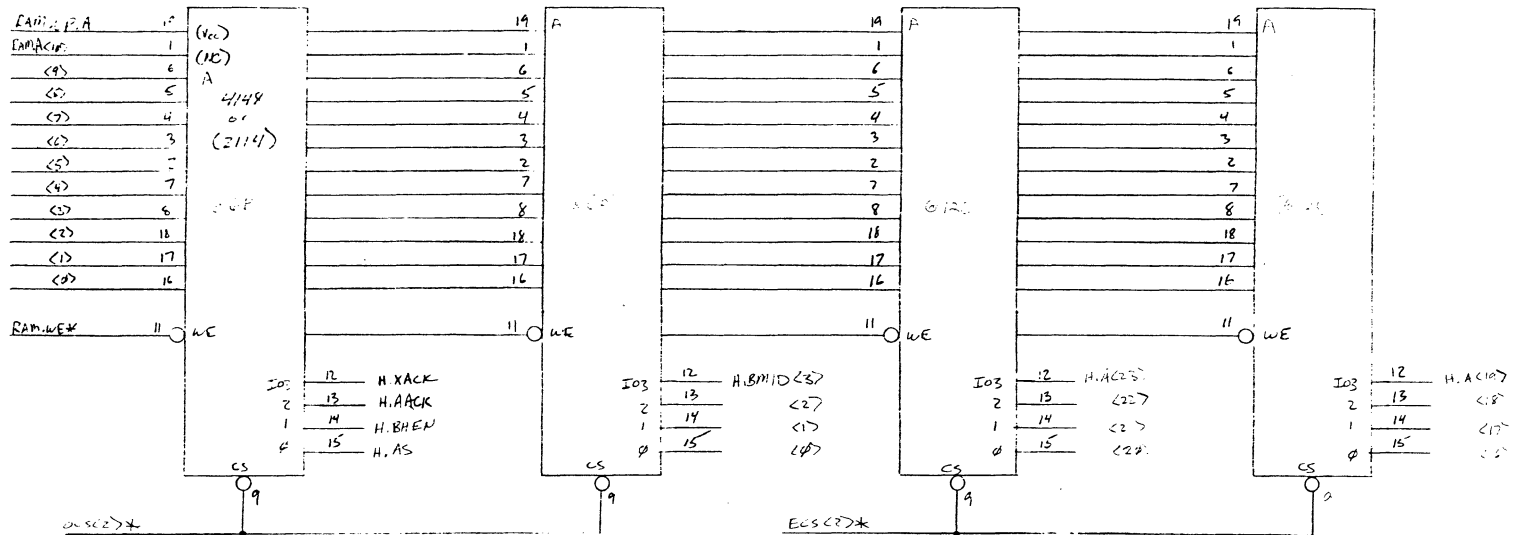
CS

RAM L.P.A

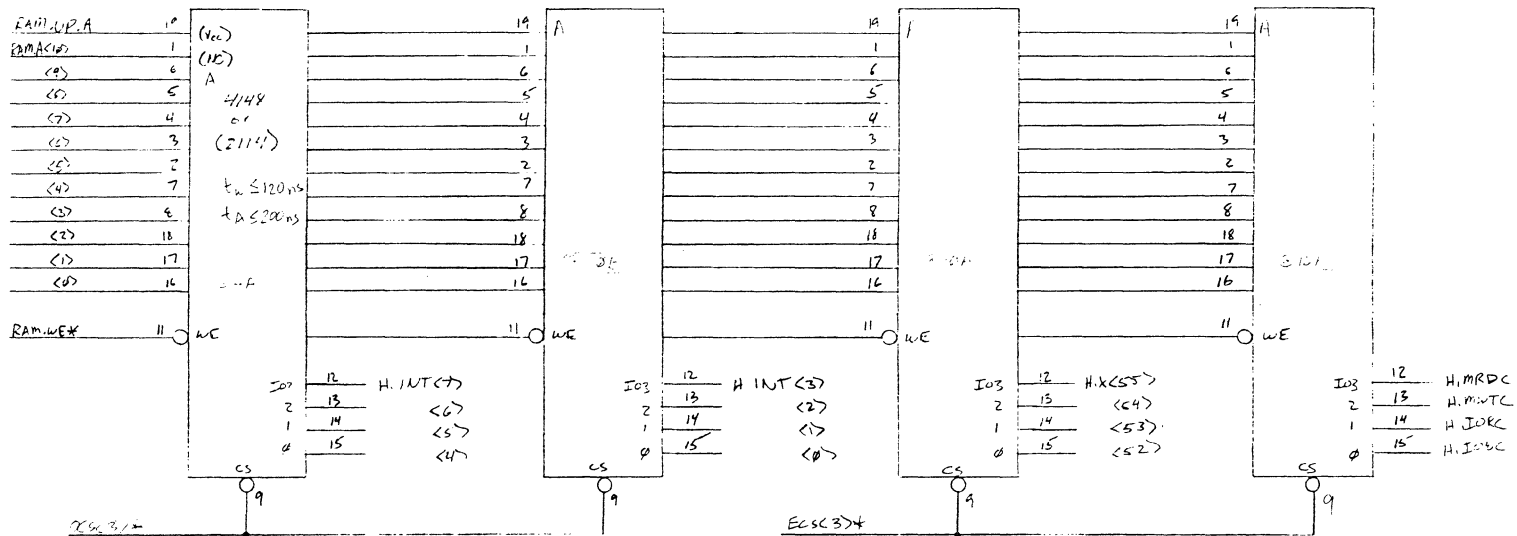
BAB HISTORY RAM 1 of 4		
SCALE:	APPROVED BY:	DRAWN BY
DATE:		REVISED
		DRAWING NUMBER
		13 of 21



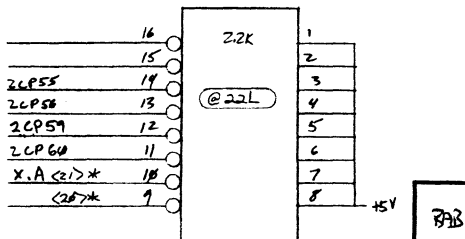
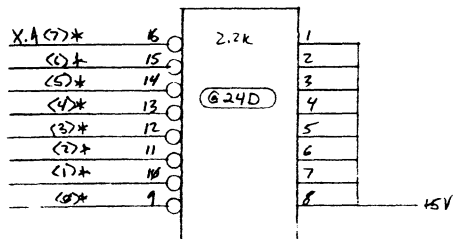
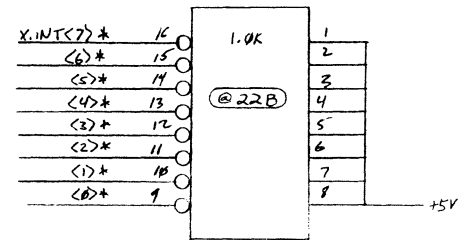
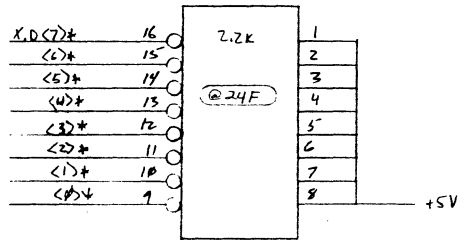
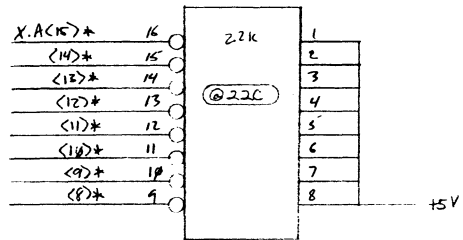
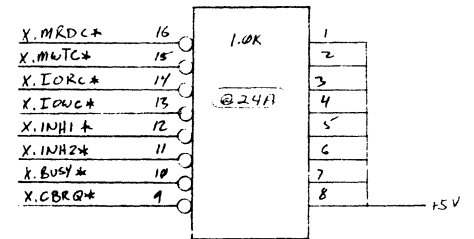
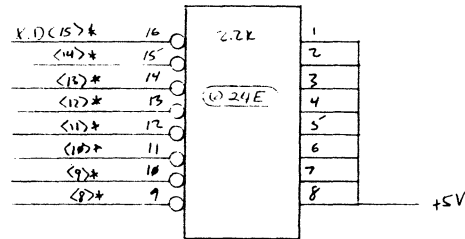
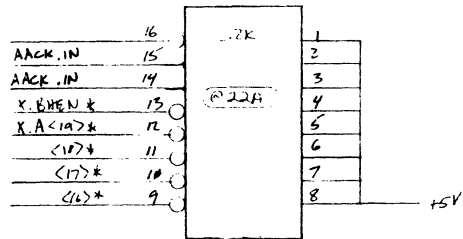
BAB HISTORY RAMP 2 of 4		
SCALE:	APPROVED BY:	DRAWN BY
DATE:		REVISED
		DRAWING NUMBER
		14 of 21



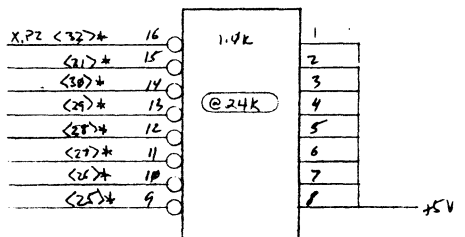
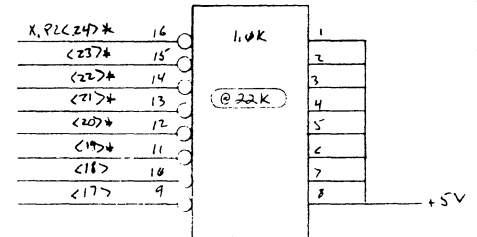
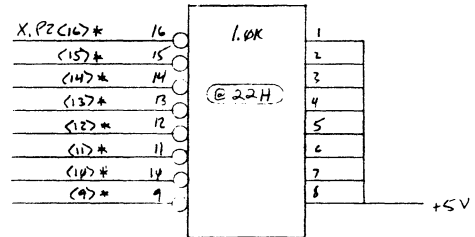
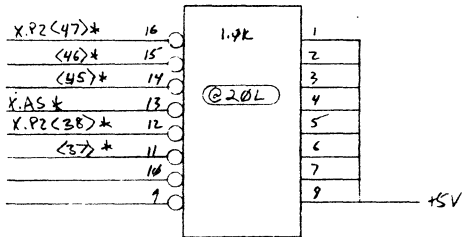
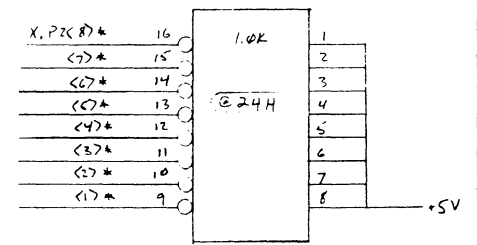
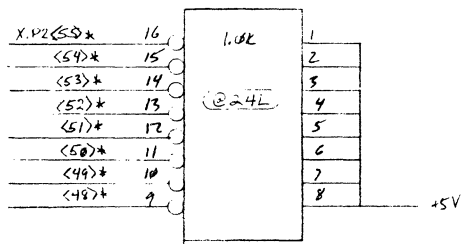
BAB HISTORY RAM 3 of 4		
SCALE:	APPROVED BY:	DRAWN BY
DATE:		REVISED
		DRAWING NUMBER
		15 of 21



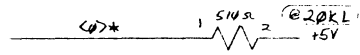
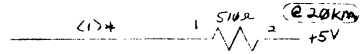
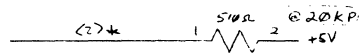
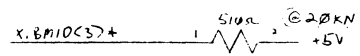
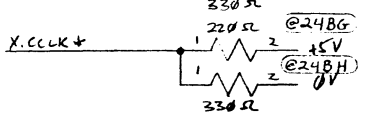
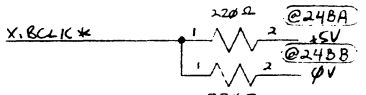
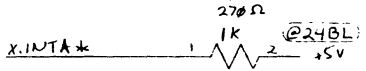
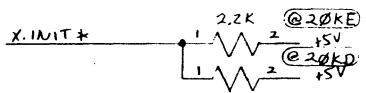
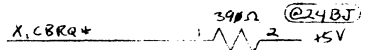
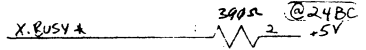
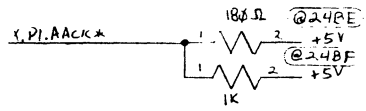
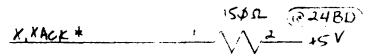
BAR HISTORY RAM 4 of 4		
SCALE:	APPROVED BY:	DRAWN BY
DATE:		REVISED
		DRAWING NUMBER
		16 of 21



BIB DIP TERMINATORS 1 of 2		
SCALE:	APPROVED BY:	DRAWN BY:
DATE:		REVISED:
		DRAWING NUMBER
		17 of 21



BAB DIP TERMINATORS 2 of 2		
SCALE:	APPROVED BY:	DRAWN BY:
DATE:		REVISED:
		DRAWING NUMBER
		18 of 21



BAB DISCRETE TERMINATORS		
SCALE:	APPROVED BY:	DRAWN BY:
DATE:		REVISED:
		DRAWING NUMBER
		19 of 21

4

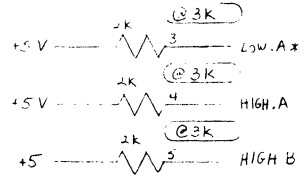
3

2

1

D

D



C

C

B

B

A

A

REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: FRACTIONS DECIMALS ANGLES ± .XX ± ± .XXX ±	CONTRACT NO.		BAB LOW AND HIGH GENERATION	
	APPROVALS	DATE		
MATERIAL	DRAWN			
FINISH	CHECKED			
DO NOT SCALE DRAWING	VALID LOGIC SYSTEMS INCORPORATED	SIZE B	CODE IDENT NO.	DRAWING NO.
		SCALE		SHEET 20 OF 21

4

3

2

1

1CP	
X.A<19>* 34	
<18>* 32	
<17>* 30	
<16>* 28	
<15>* 44	
<14>* 43	
<13>* 46	
<12>* 45	
<11>* 48	
<10>* 47	
<9>* 50	
<8>* 49	
<7>* 52	
<6>* 51	
<5>* 54	
<4>* 53	
<3>* 56	
<2>* 55	
<1>* 58	
<0>* 57	
X.INT<7>* 36	
<6>* 35	
<5>* 38	
<4>* 37	
<3>* 40	
<2>* 39	
<1>* 42	
<0>* 41	

1CP	
X.D<15>* 60	
<14>* 59	
<12>* 62	
<12>* 61	
<11>* 64	
<10>* 63	
<9>* 66	
<8>* 65	
<7>* 68	
<6>* 67	
<5>* 70	
<4>* 69	
<3>* 72	
<2>* 71	
<1>* 74	
<0>* 73	
X.INT* 14	
X.BUSV* 17	
X.MRDC* 19	
X.MWTC* 20	
X.IORC* 21	
X.IOWC* 22	
X.XACK* 23	
X.INTA* 33	
X.BHENV* 27	
X.CBRQ* 29	
X.PI.AACK* 25	

1CP	
X.BACK* 13	
X.CLK* 31	
X.INH1* 24	
X.INH2* 26	
+EV 3	
81	
X.BPC* 15	
16	

2CP	
@ 25L	
X.A<25>* x 1 56	
x 2 60	
x 3 60	
<22>* x 4 55	
x 5 59	
x 6 59	
<21>* 58	
<20>* 57	
X.BMDC<3>* 44	
<2>* 43	
<1>* 42	
<0>* 41	
X.AS* 39	
X.P2.AACK* 40	

2CP	
X.P2<54>* 54	
<53>* 53	
<52>* 52	
<51>* 51	
<50>* 50	
<49>* 49	
<48>* 48	
<47>* 47	
<46>* 46	
<45>* 45	
X.P2<38>* 38	
<37>* 37	
36	
35	
34	
33	
<32>* 32	
<31>* 31	
<30>* 30	
<29>* 29	
<28>* 28	

2CP	
X.P2<27>* 77	
<26>* 26	
<25>* 25	
<24>* 24	
<23>* 23	
<22>* 22	
<21>* 21	
<20>* 20	
<19>* 19	
<18>* 18	
<17>* 17	
<16>* 16	
<15>* 15	
<14>* 14	
<13>* 13	
<12>* 12	
<11>* 11	
<10>* 10	
<9>* 9	
<8>* 8	
<7>* 7	
<6>* 6	
<5>* 5	
<4>* 4	
<3>* 3	
<2>* 2	
<1>* 1	

BAB 796 BUS CONNECTOR		
SCALE:	APPROVED BY:	DRAWN BY
DATE:		REVISED
		DRAWING NUMBER
		21 of 21

TABLE OF CONTENTS

1.	General Information.....	1
2.	Functional Description.....	2
3.	Principles of Operation.....	3
	Addressing.....	3
	Bus Interface.....	4
	USART Section.....	5
4.	Installation/User Selectable Options.....	7
	Addressing.....	7
	CTS Selection.....	7
	Interrupt Selections.....	7
	XACK and AACK Generation.....	8
5.	Specifications.....	10
6.	Schematics.....	13
7.	2651 Data Sheets.....	17

1. General Information

The Central Data Octal Serial Interface board is designed to expand the serial I/O capacity of any Multibus¹ system. The board uses the Signetics 2651 USART (Universal Synchronous/Asynchronous Receiver/Transmitter) as the parallel to serial converter, and allows independent baud rates (transmission speeds) to be set for each of the board's eight channels.

The board supports standard EIA RS-232 interfaces, with the following pins used: TxD, RxD, DSR, CTS, DTR, and RTS. The board is capable of operating at baud rates ranging from 50 to 19,200 baud, and can be interrupt driven if so desired. Also, the board supports 16-bit I/O addressing as called for in the Multibus specification, with an option to use only 8-bit I/O addresses.

The board drives both the XACK and AACK lines of the Multibus to allow for the greatest flexibility. It can return either signal from 0-800ns after the receipt of a command, in 100ns increments. It is suggested that XACK be strapped to be equal to the access time of the board, while AACK can be strapped to allow the fastest possible system operation.

¹ Multibus is a trademark of Intel Corporation and is used throughout this manual.

2. Functional Description

The Octal Serial Interface board is divided into several major sections, which are described briefly below. For more detailed information, refer to the Principles of Operation section of this manual.

The addressing section of the board consists of the I/O address comparator and the chip-select generation circuitry for the USARTs. The board requires 32 I/O ports, which can be located on any 32-port boundary. Dip-switch I/O addressing allows the user to select the address of the board using either an 8- or 16-bit I/O address. With this ability, the board can work equally well in systems which generate a full 16-bit I/O address as well as in older 8-bit systems.

The bus interface of the board consists of the data bus buffers, interrupt circuitry, and the XACK/AACK generation logic. The data bus is buffered into and out of the board, and interrupts can be generated on the occurrence of any receiver full or transmitter empty condition. Finally, the XACK/AACK generation circuitry acknowledges all commands to the board and allows the system to run at the maximum possible speed.

Finally, the USART section is the actual interface to the external devices. This section is repeated on the board eight times, which gives eight totally independent channels. The interface between the on-board circuitry and the external connector is made through industry standard drivers and receivers, which guarantee proper RS-232 specifications.

3. Principles of Operation

This chapter details the operation of the entire interface board. Any signal names in this text followed by a slash (/) indicate that the signal is active-low.

As is all Central Data schematics, a grid system is provided to help locate sources and destinations of signals. The source of any named signal will have references to all locations on the schematics where the signal is used. At each location where a signal is used, a reference is given to where it was generated.

If the location is on the same sheet as it is being referenced, it will show only a grid location (i.e. D2). If, however, the referenced signal appears on a separate page, it will have the grid location preceded by the sheet number (i.e. 2-B5).

Addressing

Sheet 1 of the schematics includes the addressing circuitry for the board. The addressing function consists of determining when the board should be enabled as well as which USART is being accessed.

The board requires the use of 32 of the system's I/O ports. These ports can be started on any 32-port boundary, using either 8- or 16-bit addresses.

All of the address lines from the Multibus are buffered through 74LS04 gates. The buffered address lines are then routed to address decoding circuitry (A5-A15) and to chip selection circuitry (A0-A4).

The address decoding circuit consists of eleven 74LS266 open collector exclusive-NOR gates. All of the outputs of the gates are tied together, allowing any of the gates to pull the output low if its inputs do not match. If all of the pairs of inputs match, the common output is pulled high by a resistor to +5V.

One input from each of the gates goes to a buffered address line, with the other going to a dip-switch. This dip-switch, when closed, causes the corresponding gate input to become grounded. Under this circumstance, the address line leading to the same gate must also be low for the board to be addressed. If the switch position is left open, the input to the gate goes to a high state, thus comparing for a high address line.

To allow the selection between 8- and 16-bit I/O addressing, the outputs of the gates related to A8-A15 are connected through a shorting plug to the outputs of the gates related to A5-A7. If the shorting plug is installed, then the board decodes the full 16-bit address bus. If the shorting plug is removed, then the upper eight gates will not drive the common output, and thus only the lower three (A5-A7) are used for addressing.

When the address comparator is equal, pin 6 of IC39 will go high. This line is used in conjunction with pin 5 to enable the 74LS138 decoder. Pin 5 is low whenever an I/O command is currently on the bus.

When the 74LS138 is enabled, it uses the next lower three address lines (A4, A3, and A2) to select which USART should be enabled. In this manner, each USART has four consecutive addresses, for both reading and writing. The output of the decoder drives the chip-select pins of the USARTs. The R/W/ line of the USART is used to determine if a read or write operation is to occur when the chip is selected. When this line is high, a write will occur, while in read mode the line will be low.

In summary, the board uses 32 of the I/O ports on the system. The base address for these ports is selected with dip-switches, and the 32 ports are divided equally among the USARTs.

Bus Interface

The bus interface consists of the buffering circuits required from the Multibus, the interrupt driver, and the XACK/AACK generation logic. The bus interface circuitry is also found on sheet 1 of the schematics.

The data bus buffers consist of two 74LS242s, each one buffering four data lines. These are inverting buffers, thus immediately correcting for the inverted data on the bus. Since the directional enable pins of the buffers are of opposite polarity, they can be tied together, and are driven by a signal (pin 8 of IC28) which goes high whenever the board is addressed and an I/O read command is in progress. During all other conditions, this signal is low, thus sending data from the Multibus into the board.

The IORC/ and IOWC/ signals from the Multibus are also buffered (through 74LS08 gates) and used to set the level of the R/W/ line on the USARTs.

The interrupt circuitry takes the two bussed interrupt outputs from the USARTs (explained later) and allows them to be gated to

form an interrupt signal to the processor. If either the TINT/ or the RINT/ signal goes low, and it is jumpered into pin 8 of IC29 (with shorting plugs), it will cause a vectored interrupt to be generated on the selected line.

The board generates two command acknowledge signals. The first, XACK, indicates when a data transfer is complete and the processor can go to the next cycle. The other, AACK, gives the processor advance information related to when a transfer will be complete.

The circuit which generates the acknowledge signals consists of a shift register (74LS164, IC25) which is kept cleared when the board is not active. When an I/O command occurs, the clear input goes high, allowing the register to shift 1's through at the CCLK rate. The eight outputs of the shift register, which go high from 100-800ns after the time a command starts, can be jumpered to the SACK and AACK drivers (IC26). Note that since the command is asynchronous with respect to the bus clock the outputs may vary up to one clock cycle (i.e. the second output can occur anywhere from 100-200ns after command initiation).

The user can also select either acknowledge signal to be returned as soon as the board is selected by tying the driver's input high. The drivers are enabled whenever a command is occurring to this board, thus gating the proper timing onto the bus.

Sheet 1 of also contains the crystal oscillator which is used as a time-base for the USARTs. The oscillator is a simple feedback network, with the resistors used to bias the 7404 gates into their linear region, and the 100pf capacitor used to block any DC voltage to the crystal and to stabilize operation. After buffering, this 5.0688MHz signal is sent to the eight USARTs, whose internal dividers generate baud rate clocks from it.

USART Section

Sheet 2 of the schematics shows the actual interface to the external devices. Note that this sheet is repeated eight times on the board, with the IC numbers listed for ports 0-7, in that order. Also, the signal CSX is referenced with the number 0-7 instead of the trailing "X" to indicate which USART is being used.

The format of the characters being sent and received is determined entirely by the USART and how it is programmed. Details on the programming of the 2651 are provided in the Signetics 2651 data sheet.

Note that there is one strap selection available for each USART. This strap, for CTS selection, is required because the USART will not transmit any characters unless its CTS/ pin is low. Since many serial devices do not drive this line, the strap labeled CTS INT allows the user to drive it from the RTS/ output of the USART. With this arrangement, whenever the RTS/ signal from a USART is low, it will be allowed to transmit. In the other mode, with the CTS EXT strap in place, the external device must drive CTS in order for the board to operate properly.

The USARTs can generate an interrupt on the occurrence of any transmitter empty or receiver full condition. All of these interrupt outputs are wire-ORed together to form a common transmitter interrupt and receiver interrupt signal. These two signals can be gated to any of the eight Multibus vectored interrupt lines (see the Bus Interface section).

All of the RS-232 signals from the external connector are buffered by 1489s. Note that capacitors can be added to slow the rise and fall times on all of the inputs to the USART. Normally, however, these additional capacitors are not needed.

4. Installation/User Selectable Options

The Octal Serial Interface is designed to operate in any standard Multibus system. The board can occupy any card position of the system, since it does not operate as a bus master.

Addressing

The board has a 12-position dip-switch to select the port addresses it will respond to. Each position of the switch corresponds to one address line, from A5 to A15, with the right-most position not used. As marked on the board, A15 is selected by the left-most switch, while A5 is selected by the second from the right. An address line is compared for "0" if the switch is closed (up), as printed on the board. With the switch left open (down), the corresponding address line is compared for "1".

If 16-bit I/O addressing is to be used, a shorting plug must be placed over the two wire-wrap pins marked EXTENDED I/O. For systems where only 8-bit I/O addressing is used, this shorting plug should be left off. Also, for 8-bit systems, the upper eight address switches are not used.

CTS Selection

Since the USART will not transmit any data unless the CTS signal is active, the board allows the user to jumper it to a known state. This option can be used when the board is being connected to a simple device which does not generate this signal.

When the user wants the USART's RTS output to drive its CTS input, then a shorting plug should be placed in the USART's CTS INT position. This will allow the USART to transmit regardless of the state of the CTS signal from the external connector. If the user wishes CTS to be monitored from the device, then the CTS EXT position should be shorted. This will cause the output of the CTS buffer from the external connector to be run to the USART's CTS input.

Interrupt Selections

Each USART has its transmitter and receiver interrupt outputs tied to the others. The resulting two signals can be strapped to one of the vectored interrupt lines of the Multibus.

To allow transmitter interrupts, a shorting plug must be placed over the wire-wrap pins marked T INT. This plug should be left off to disable all transmitter interrupts for the board. Likewise, receiver interrupts are enabled by placing a shorting plug over the pins marked R INT.

Once the proper interrupt types are allowed, a vectored interrupt level must be established. The user can pick any level (0-7) to receive the interrupt by placing a shorting plug on the appropriately marked pins on the board.

XACK and AACK Generation

In order for the board to acknowledge processor commands, two lines are provided to indicate when a data transfer is complete. The XACK (transfer acknowledge) line is driven by the board when the transfer is completely finished, and the processor is allowed to complete the cycle. The AACK (advance acknowledge) is provided to allow systems to operate at their full speed potential (by preventing wait states), since it can be returned before XACK. Only XACK is used to indicate when a cycle can end, with the function of AACK to give advance information concerning the timing of the board.

Both of the lines can be strap selectable to return to the processor from 0-800ns after a command is received, in 100ns increments. The selection of timing for each line is done with shorting plugs placed over wire-wrap pins on the board.

The board has two rows of wire-wrap pins which are used for XACK/AACK generation. The top row is used for XACK, while the bottom row is for AACK. Each row consists of 9 pairs of pins, with each pair being one timing combination. To setup the board, the user needs to place a shorting plug in each row, under the timing number which he desires.

The timing numbers are marked to be the maximum return time for the signal involved (multiplied by 100ns). The minimum time is 100ns below the maximum time. For example, the pins marked "4" will return their signals from 300-400ns after a command is received. The pins marked "0" always return the signal immediately.

Since the XACK timing is tied to the access time of the board, the setting of that plug is suggested to be "3". The setting of the AACK strap will have to be determined by the system designer, using the information presented here.

If pin 25 of your system is being used for LOCK/ (as specified in the IEEE Multibus specifications), then the AACK driver must be disconnected by removing the shorting plug on the pins marked AACK ENBL. Leaving this on causes the board to drive pin 25 with the AACK signal when it is selected. This strap is available only on boards with revision A or greater.

One note--the timing for both acknowledge lines is dependent on the CCLK (constant clock) signal from the Multibus. It is assumed here that this clock is running at 10MHz, so if any other frequency is used on the system, the spacing between strap positions will be the period of the actual clock rather than 100ns. For example, a system with a 9.5MHz CCLK signal will have 105ns strap selection spacing.

5. Specifications

Word Size

8 bits

Addressing

This board requires 32 I/O ports. The base address for these ports can be on any 32 port boundary. Normally, 16-bit addressing is used for port selection. By changing a strap, however, 8-bit addressing can be selected.

Each USART requires four consecutively addressed ports, and their function is described below.

<u>Address</u>	<u>Input Function</u>	<u>Output Function</u>
0	Receiver data reg.	Transmitter data reg.
1	Status register	SYN1/SYN2/DLE regs
2	Mode register	Mode register
3	Command register	Command register

Access Time

350ns maximum

Baud Rates Available

50, 75, 110, 134.5, 150, 300, 600, 1200, 1800, 2000, 2400, 3600, 4800, 7200, 9600, and 19,200.

Interrupt Sources

Any transmitter empty or receiver full condition can trigger an interrupt on any of the eight vectored interrupt lines of the Multibus. Independent straps allow all transmitter interrupts or all receiver interrupts to be disabled.

RS-232 Specifications

The drivers and receivers used on the board are the 1488 and 1489 type. This provides a standard interface for the following lines: TxD, RxD, DTR, RTS, CTS, and DSR.

Interface

All signals meet the IEEE Multibus proposed specification.

12 Pin Edge Connector

Part Number: 345-012-500-201

Manufacturer: EDAC, 20 Railside Road, Don Mills, Ont. M3A1A4

Electrical Characteristics

Vcc= +5V +5%

Vdd= +12V +5%

Vbb= -12V +5%

Icc= 1.2A typ, 2.0A max

Idd= 0.1A typ, 0.2A max

Ibb= 0.1A typ, 0.1A max

Environmental Characteristics

Operating Temperature: 0 C to +55 C

Relative Humidity: 0 to 90% (non-condensing)

Physical Characteristics

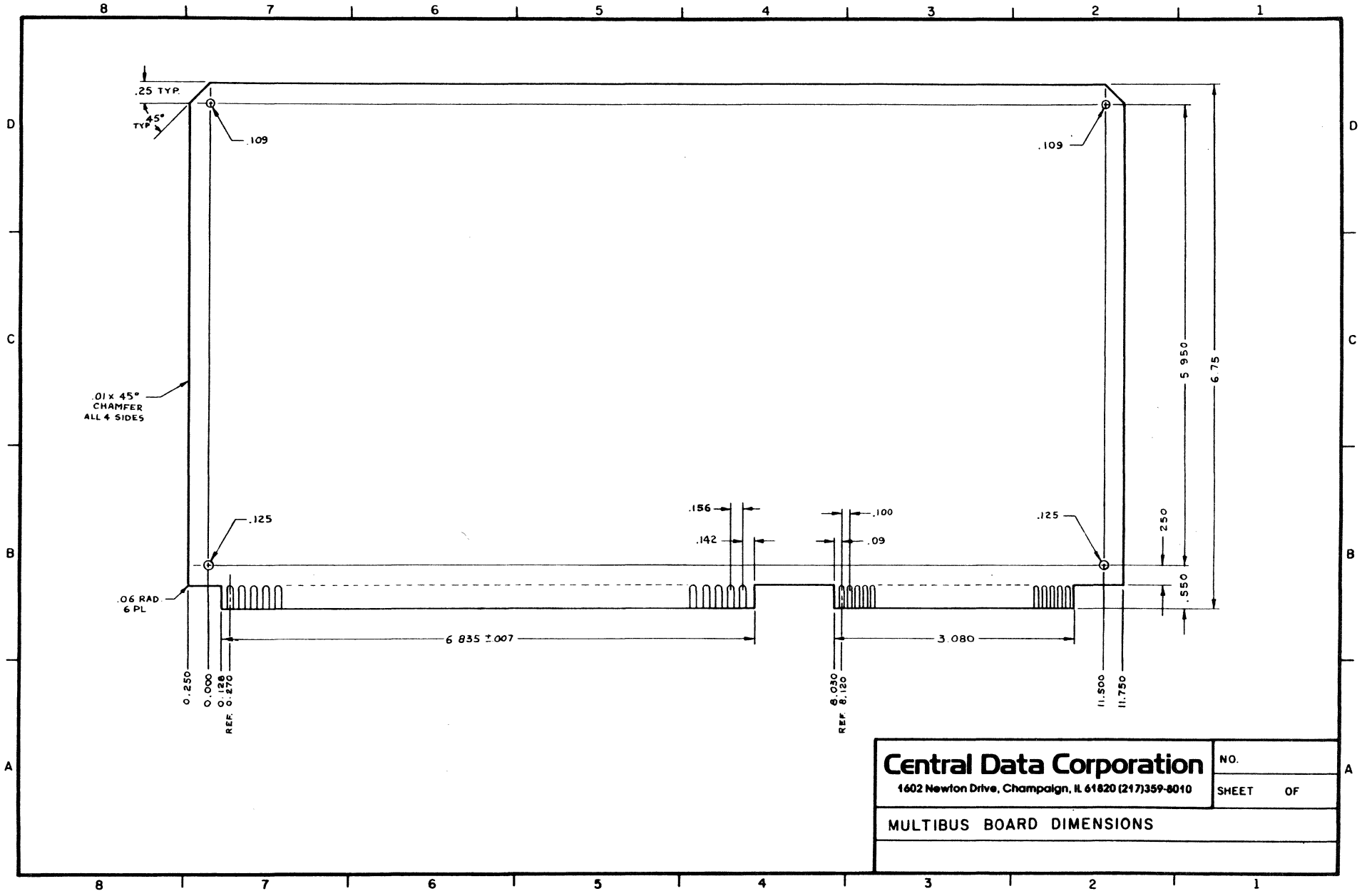
Dimensions: see the basic Multibus dimensions on the following page. Each edge connector is 0.680" wide, with the right edge of each connector being 1.135", 2.360", 3.585", 4.810", 6.010", 7.235", 8.460", and 9.685" from the right-hand reference hole.

Weight: 9oz (255gm)

Ordering Information

Part Number: B1018

Description: Multibus Octal Serial Interface Board



Central Data Corporation		NO.	A
1602 Newton Drive, Champaign, IL 61820 (217)359-8010		SHEET	OF
MULTIBUS BOARD DIMENSIONS			

6. Schematics

The following pages contain the schematics for the Octal Serial Interface board. A full description of the circuitry is given in the Principles of Operation section of this manual.

8 7 6 5 4 3 2 1

D
C
B
A

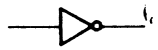
3-INPUT 'AND' GATE



2-INPUT 'OR' GATE



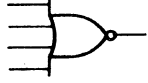
INVERTER



2-INPUT 'NAND' GATE



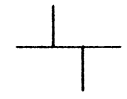
4-INPUT 'NOR' GATE



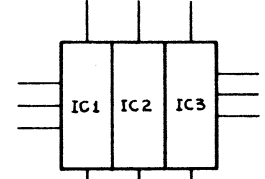
TWO LINES - NO CONNECTION



3 LINES - ALL CONNECTED



GROUP OF SIMILAR PARTS

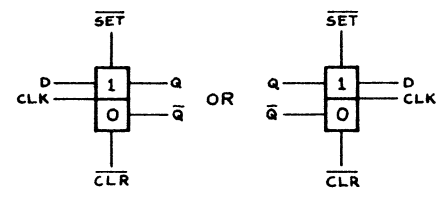


ALL LINES ENTERING ON THE TOP OR BOTTOM ARE SEPARATE FOR EACH CHIP

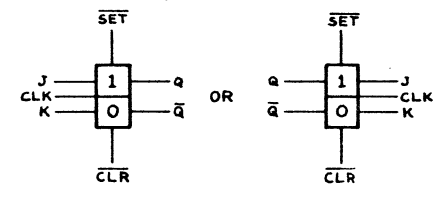
ALL LINES ENTERING ON THE SIDES ARE BUSSED TO ALL CHIPS

UNMARKED ARROWS GO TO +5V

D-TYPE FLIP-FLOP



J/K FLIP-FLOP

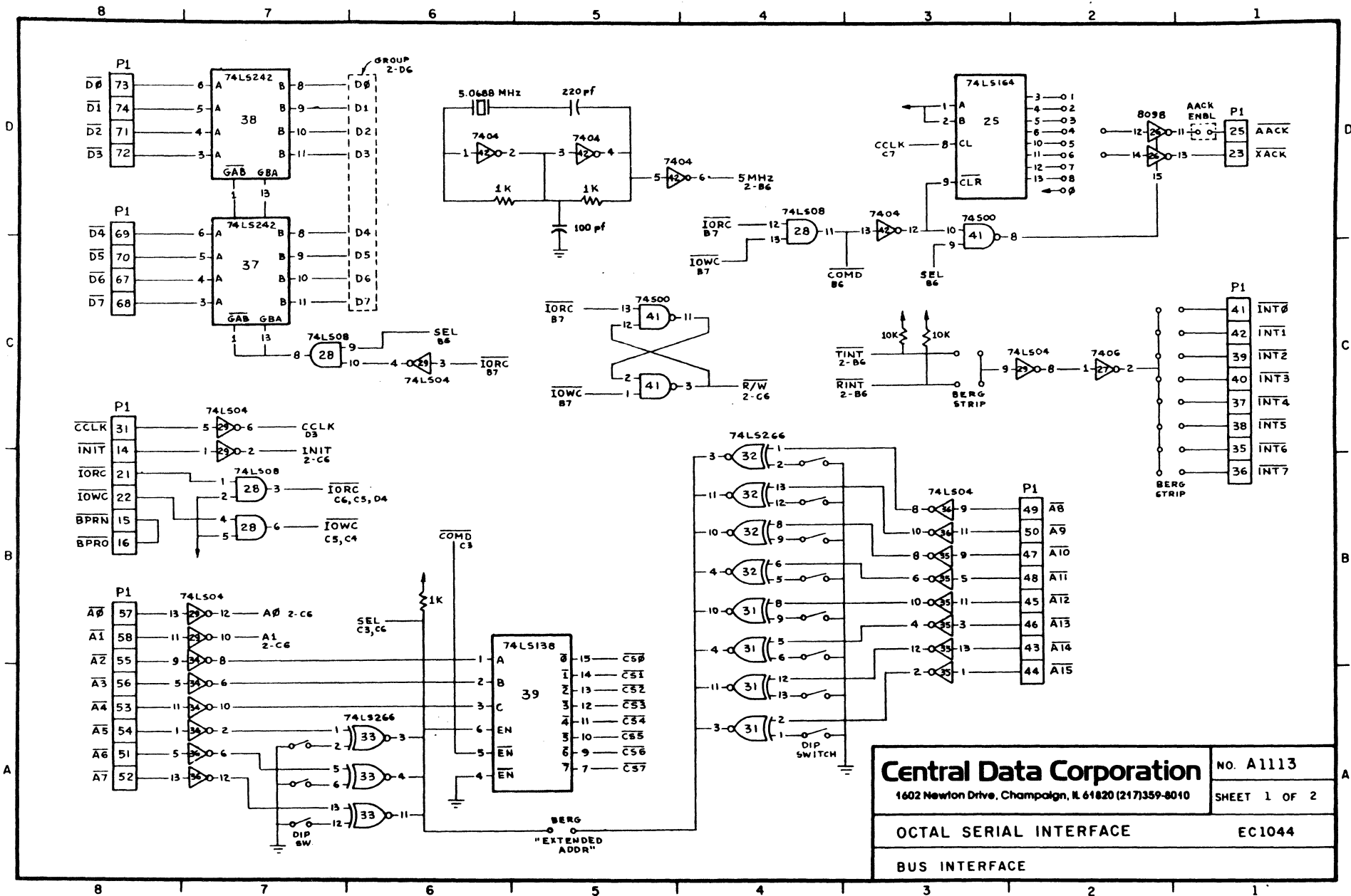


Central Data Corporation
1602 Newton Drive, Champaign, IL 61820 (217)359-8010

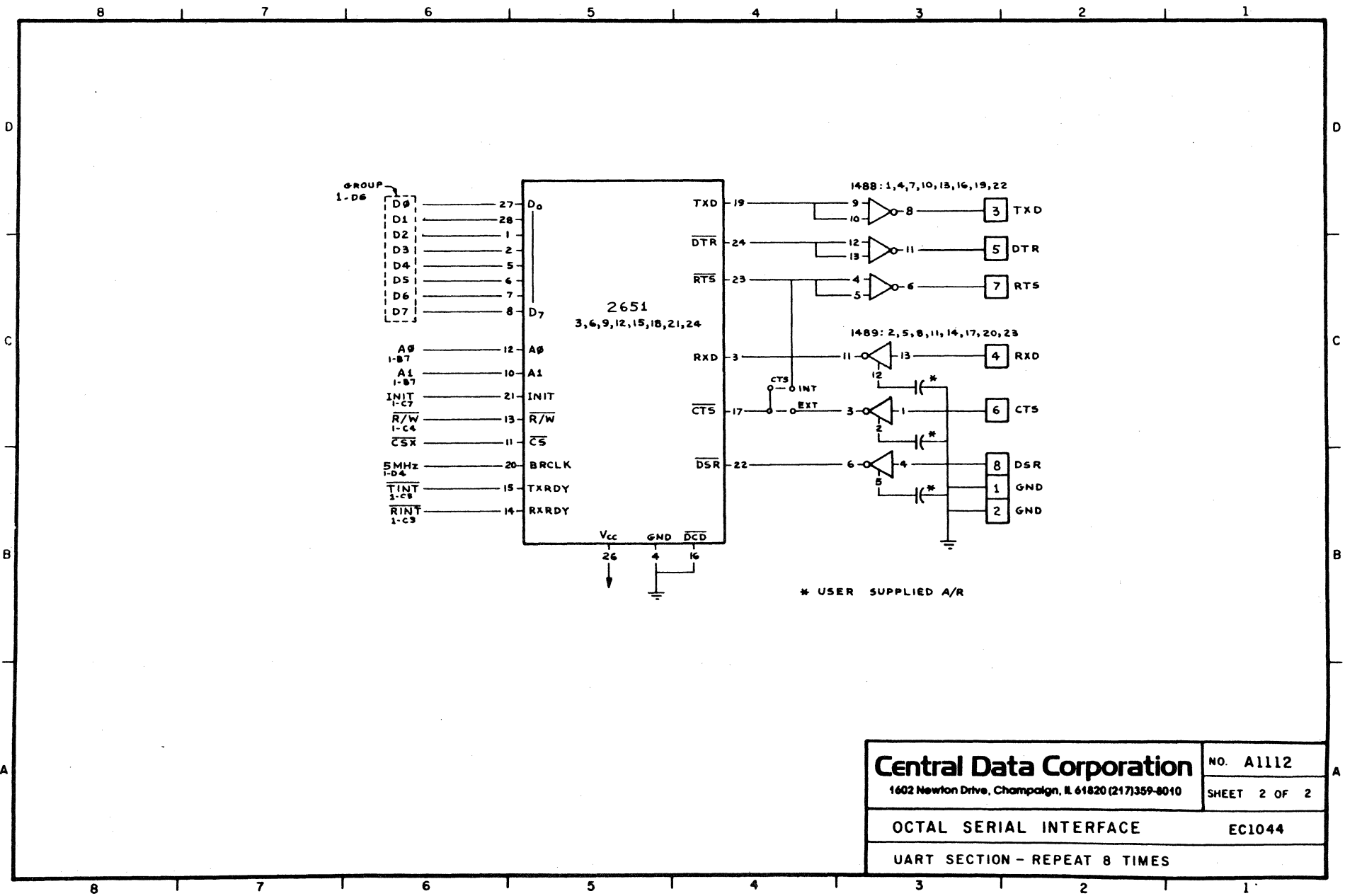
NO. _____
SHEET OF _____

DRAWING CONVENTIONS

8 7 6 5 4 3 2 1



Central Data Corporation		NO. A1113
1602 Newton Drive, Champaign, IL 61820 (217)359-8010		SHEET 1 OF 2
OCTAL SERIAL INTERFACE		EC1044
BUS INTERFACE		



Central Data Corporation 1602 Newton Drive, Champaign, IL 61820 (217)359-8010	NO. A1112
	SHEET 2 OF 2
OCTAL SERIAL INTERFACE	EC1044
UART SECTION - REPEAT 8 TIMES	

Model 3C400
MULTIBUS Ethernet (ME) Controller
Reference Manual
May 18, 1982

ABSTRACT

This document describes the 3Com 3C400 Multibus Ethernet Controller that connects any Multibus compatible system processor to a DEC-Intel-Xerox Ethernet Communication System.

3Com Corporation 1390 Shorebird Way, Mountain View, California 94043 USA
(415) 961-9602 Telex: 345546

7/21/82

This is an addendum to the
Model 3C400
MULTIBUS Ethernet (ME) Controller
Reference Manual
of
May 18, 1982

CORRECTIONS:

Page 4-6: The "address match" status bit is inverted. In other words, the bit is 0 if the destination address of the received packet is equal to the station address, and 1 if it is not.

Page 4-6: The "broadcast" status bit is also inverted. In other words, the bit is 0 if the the destination address is the broadcast address (all ones), and 1 if it is not.

CLARIFICATIONS:

Page 4-5: The JAM bit is cleared by setting it (writing a one into the JAM bit).

Page 4-5: The TBSW bit remains 1 during all JAM processing. It does not become 0 again until the packet has been successfully transmitted.

Page 4-6: There is no status bit indicating a multicast packet which is not a broadcast packet. Such a packet is identified by a broadcast status bit equal to 1 (false) and the multicast bit in the destination address being 1 (true). Multicast packets will only appear in receive buffers if the receiver is enabled to accept them.

Page 4-7: Broadcast is a special case of multicast. Enabling the receiver for multicast (modes 3,4,5) includes the broadcast address.

Page 4-7: The multicast bit of the destination address is the least significant (low order) bit of the first byte of the packet. In a receive buffer, this is the byte immediately following the buffer header word.

Page 4-12: The number of retransmissions normally exceeds 15 only when the Ethernet is broken. When this occurs the only way to get the ME to return the transmit buffer to the processor is to reset the controller by setting bit 8 of MECSR.

TABLE OF CONTENTS

CHAPTER 1 - 3COM ME CONTROLLER SPECIFICATIONS

- 1.1 DESCRIPTION
- 1.2 ME FEATURES
- 1.3 ME SPECIFICATIONS

CHAPTER 2 - BACKGROUND INFORMATION

- 2.1 BASIC ETHERNET SUBSYSTEMS
- 2.2 EXAMPLE FILE TRANSFER
- 2.3 HOW 3COM PRODUCTS IMPLEMENT ETHERNET
FOR MULTIBUS COMPATIBLE SYSTEMS
- 2.4 ETHERNET OPERATION
- 2.5 TRANSMISSION SUBSYSTEM
- 2.6 ME CONTROLLER SUBSYSTEM

CHAPTER 3 - PHYSICAL DESCRIPTION

- 3.1 ENVIRONMENT
- 3.2 PACKAGING
- 3.3 BLOCK DIAGRAM

CHAPTER 4 - 3COM ME PROGRAMMING

- 4.1 MEMORY ALLOCATION
- 4.2 TRANSMIT/RECEIVE
- 4.3 PROGRAMMING ANOMALIES
- 4.4 OPERATION

CHAPTER 5 - INSTALLATION AND CONFIGURATION CONSIDERATIONS

- 5.1 INSTALLATION CHECKLIST
- 5.2 ETHERNET ADDRESSING CONSIDERATIONS

CHAPTER 6 - FACTORY WARRANTY

APPENDICES

- A. ORDERING INFORMATION
- B. BIBLIOGRAPHY
- C. SAMPLE DRIVER

CHAPTER 1

3COM ME CONTROLLER SPECIFICATIONS

1.1 DESCRIPTION

The 3C400 Multibus Ethernet (ME) Controller provides the connection to Ethernet for any Multibus compatible system processor. It consists of one Multibus (IEEE-796) board that plugs into the Multibus. (See Figure 1-1 below).

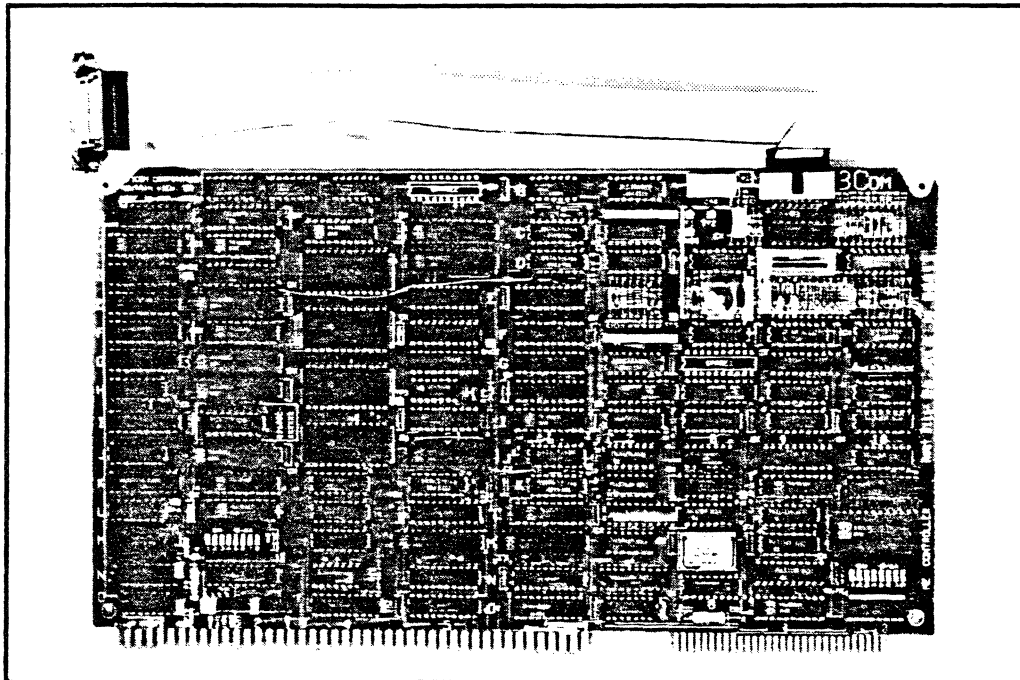


FIGURE 1-1. 3COM MULTIBUS ETHERNET CONTROLLER BOARD SET

Connection from the 3C400 ME Controller to Ethernet is made via the 3Com Ethernet Transceiver and Transceiver Cable, or any other Ethernet compatible transceiver and cable.

The 3C400 Controller, 3C100 Ethernet Transceiver and the 3C110 Transceiver Cable conform to the Ethernet specifications, version 1.0, published by DEC, Intel, and Xerox on 30 September, 1980. When coupled with customer supplied driver software, they implement layers one (physical) and two (data link) of the **International Standards Organization Reference Model for Open Systems Interconnection**. Any **Multibus compatible system processor so equipped will be compatible with any other Ethernet-based system at the physical and data link levels.**

1.2 ME FEATURES

- Compatible with 10 megabit-per-second DEC, Intel, Xerox Ethernet.
- Compatible with Multibus (IEEE-796).
- Connects to Ethernet using 3Com Ethernet Transceiver and Transceiver Cable or any other transceiver and cable that conform to the Ethernet specification.
- Controller, transceiver, and cable together provide a complete hardware implementation of the Ethernet specification except for multicast address comparison and random number generation for retransmission timing.
- Includes 8K byte dual-ported memory which appears in Multibus memory space. Transmission between the dual-ported memory and Ethernet do not consume Multibus cycles, allowing concurrent processing.
- Three 2K byte buffers can each handle maximum packet size allowed by Ethernet specification. One buffer is dedicated to transmission, two to reception of Ethernet packets.
- Under software control, each packet buffer may be independently connected to either the Ethernet or the Multibus.
- Multibus-addressed buffers allow in-place packet assembly, processing and multiplexing.
- Can receive minimally spaced packets.
- Controller can be selectively enabled to recognize packets containing station address, broadcast packets, multicast packets or all packets.
- Ethernet address assigned by 3Com is held in PROM on controller and can be referenced or replaced by software when loading address recognizer.
- Manchester decoding using phase-locked loop circuitry.
- 32-bit CRC generated on transmission and verified on reception.
- Hardware generation and removal of preamble.
- Hardware retransmission timing with random number supplied by software.
- Hardware detection of oversized and undersized packets and alignment errors.
- Functions as 16-bit memory slave and is compatible with 8-bit and 16-bit masters.
- Contained on one Multibus board.

1.3 ME SPECIFICATIONS

Compatibility

Ethernet	Conforms fully to Ethernet specification, version 1.0, published 30 September, 1980, by DEC, Intel, and Xerox.
Multibus	Conforms fully to Multibus specification. Functions as 16-bit memory slave. Compatible with 8-bit and 16-bit masters.
IEEE-796	Compliance is D16 M24 V0 (16-bit transfers, 24-bit addressing, non-bus vectored interrupts).

Functions

Serial/parallel and parallel/serial conversions.
Transmit and receive buffering.
Framing of packets.
Manchester encoding and decoding.
Address recognition
Collision and error detection.
Preamble generation and removal.
Carrier sense and deference.
Backoff and retransmission timing.
Collision fragment filtering.
Frame check generation and detection.
Alignment error and overrun filtering.

Memory

Transmit	One 2K byte dual-ported RAM memory.
Receive	Two 2K byte dual-ported RAM memories.
Control & Status	Registers which occupy 2K bytes of Multibus address space.
Byte Ordering	Low order first or high order first, switch selectable.
Address	Occupies 8K bytes of Multibus memory address space. Starting address set by switches at any 8K byte address boundary in range 0 to 1016K bytes.

Timing

Bit Rate	10 million bits per second
Packet Spacing	9.6 microseconds minimum
Transmit Delay	1 microsecond typical, without deference
Receive Delay	3 microseconds typical
Jam Time	Transmits 32 bits of zeroes when collision is detected.

Ethernet Packet Format

Length	512 bits minimum, 12,144 bits maximum, excluding 64-bit preamble.
Format	Destination Address.....48 bits Source Address.....48 bits Type.....16 bits Data.....8n bits where n=46 minimum, 1500 maximum Frame Check Sequence.....32 bits
Frame check sequence	32 bit CRC generated on transmit, verified on receipt.
Preamble	Generated and removed by controller.

Address Recognition

Ethernet Address Unique 48-bit address assigned by 3Com. Held in socketed PROM on controller, appears in Multibus address space.

Address Recognizer Station Ethernet address held in RAM memory, may be loaded from Ethernet address PROM or with an address supplied by software. Can be selectively enabled by software to recognize packets containing:

- Station or broadcast address
- Station, broadcast, or multicast address
- Any address

Error Handling

Controller can be selectively enabled by software to reject packets with FCS, alignment or range errors.

Interrupts

Interrupt Conditions Selectively enabled by software:

- Transmit done
- Receive buffer A full
- Receive buffer B full
- Collision (jam)

Priority levels All interrupts use a common priority level, jumper selectable from INTO to INT7.

Software Functions

The following functions must be performed by customer software:

- Loading of Ethernet address
- Multicast address comparison
- Random number generation for retransmission timing after collision.

Installation

Size One Multibus-standard board

- 30.5cm X 17.1cm
- 12in X 6.75in

Slots	Requires one slot.
Power	5A at +5V 0.5A at +12V for transceiver
Bus loading	One DC load
Transceiver cable connector	Ethernet-standard female 15-pin "D" subminiature connector attached to the controller via cable.
Transceiver cable	Uses Ethernet-standard transceiver cable which must be ordered separately.
Ethernet address	Unique address supplied by 3Com for each controller, contained in onboard PROM.
Operating Environment	
Temperature	5° to 55° C
Humidity	10% to 90% without condensation

CHAPTER 2

BACKGROUND INFORMATION

This chapter covers some background information about both the Ethernet and the 3Com ME Controller.

In the last decade, computers have grown from a luxury to a necessity in most businesses. Similarly, in the next decade, **inter-computer communication** will grow from a luxury to a necessity.

The "Ethernet" network, developed for machine-machine communication, was pioneered at Xerox Corporation as an appropriate implementation for inter-computer communications. In use since 1974, the Ethernet has evolved to an industry standard, documented in the Ethernet Specification, published September 30, 1980 by DEC, Intel, and Xerox.

The benefits of modern computerized workstations are now magnified as they communicate information to other devices at 10 million bits-per-second over the Ethernet. What's more the Ethernet network can be tailored to end user's needs and workstations once the Ethernet coaxial cable is in place. This means multiple workstations can share

resources such as:

Word processors	Data bases
Printers	Process control stations
Electronic mail systems	Array processors
Graphics stations	Laser printers
Transaction workstations	Etc.

Individual workstations and shared resources are plugged into Ethernet Information Outlets in the wall the same way telephones are plugged into telephone wall-outlets. However, the Ethernet's 15-pin connector is more complex than a telephone connector.

Each Ethernet device is assigned a unique address (like a unique telephone number), therefore, it can be moved around and plugged into any convenient Ethernet information outlet. Further, all devices plugged into the Ethernet can talk to each other, by mutual agreement, similar to two people talking on the telephone.

Ethernet, due to its **standardized** physical and logical protocol, allows users to mix and match equipment from **multiple vendors**.

In the future a voice capability will probably be integrated into Ethernet for store-and-forward voice communications to complement electronic mail.

2.1 BASIC ETHERNET SUBSYSTEMS

The Ethernet is a bus-oriented communication system that supports up to 100 stations using a 50 ohm coaxial cable as the bus.

Figure 2-1 below shows the basic parts of a typical Ethernet system, with workstations connected to the Ethernet coaxial cable.

The **Transmission Subsystem** is made up of 50 ohm coaxial cable, terminators, transceivers, and transceiver cables.

The **Controller Subsystem** is the set of controller boards and the software

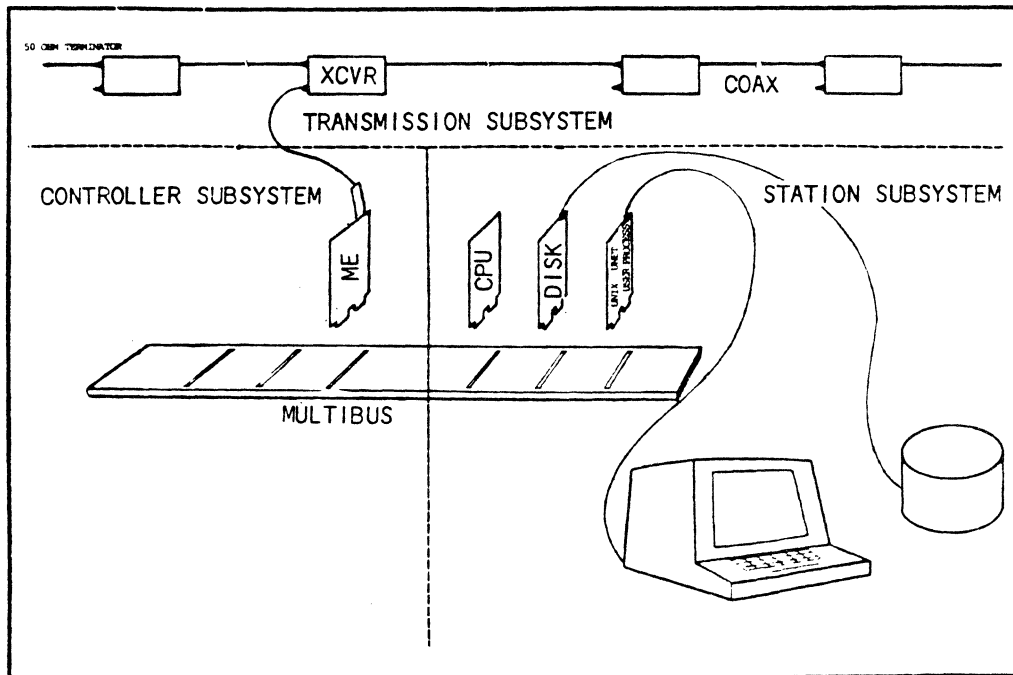


FIGURE 2-1. BASIC ETHERNET SUBSYSTEMS

supporting them.

The **Station Subsystem** is everything else associated with the station, such as, the terminal, processor, disk, and higher level protocol software.

These three subsystems are discussed again later in this chapter, in terms of 3Com product implementation.

Meanwhile, the following example describes how a file of information is transferred from one device to another using Ethernet.

2.2 EXAMPLE FILE TRANSFER

(This is an example text file transfer using a File Transfer Program running on the host processor.)

1. The terminal user runs the File Transfer Program, connects to the receiver, and specifies the file to be transferred.
2. The file's characters are mapped into device-independent virtual characters (by software) to meet protocol specifications.
3. The mapped character stream is then routed to a virtual circuit set up between the two devices.

4. The virtual circuit protocol software breaks the character stream into packets for transmission. (It also retransmits corrupted packets, and limits data rate to avoid overruns.)

5. The packets are then passed to the Ethernet driver software.

6. The Ethernet driver then copies the packet into a packet buffer and tells the controller to transmit it.

7. The controller waits until the coaxial cable is not in use, then transmits the packet.

8. The Ethernet transceiver receives the packet's bit stream and injects it onto the coaxial cable. (If the transceiver detects a collision, it signals the controller to retransmit.)

9. The receiving station recognizes its address and reverses the above procedure: bits are received by the transceiver, fed to the controller, passed to software that reassembles the packets, maps the characters, and stores the data.

2.3 HOW 3COM PRODUCTS IMPLEMENT ETHERNET FOR MULTIBUS COMPATIBLE DEVICES

For a complete local computer network, there are only three additional components needed by Multibus compatible systems:

1. **3Com Ethernet Transceiver** - fully conforms to published Ethernet specifications and connects directly to the Ethernet coaxial cable.

2. **3Com Ethernet Controller** - plugs directly into the Multibus

3. **Higher-level Protocol Software** - providing high-level network protocol services - including data link drivers. 3Com's UNET Software is UNIX compatible and provides the Internet Protocol (IP), Transmission Control Protocol (TCP), file transfer protocol (UFTP), electronic mail protocol (UMTP), virtual terminal protocol (UVTP), etc.

2.4 ETHERNET OPERATION

The Ethernet is a carrier sense, multiple access transmission system with collision detection (CSMA/CD). To transmit a packet, a station waits for quiet on the network (defers). When the network is quiet, it starts to transmit the packet.

During packet transmission, the station also watches for collisions with other transmitters; these may occur within one round-trip time through the network. The station is said to have "acquired the network" if no collision occurs in that time interval. If a collision does occur, the station transmits 4 to 6 additional bytes of data (jam) and the aborts the packet. The extra bytes insure that any other participant in the collision is sure to see it. The station then waits a random amount of time (backoff) before retransmitting (after deferring to packets in progress on the network).

2.5 TRANSMISSION SUBSYSTEM

The transmission subsystem, in the form of a 3Com "starter package",

(Model 3C440) is shown in **Figure 2-2** below. It consists of four types of components: **transceiver cables**, **transceivers**, **coaxial cable**, and **terminators**. These are described below.

Transceiver Cable - The transceiver cable is a 15 meter shielded twisted pair cable that connects the controller to the transceiver. It has 4 pairs, one each for transmit, receive, collision detect, and power. It has a male 15 pin connector with lock posts on the controller end and a female 15 pin D connector with slide lock assembly on the transceiver end. Thus the cables can be concatenated to make a longer cable, up to the maximum length of 50 meters.

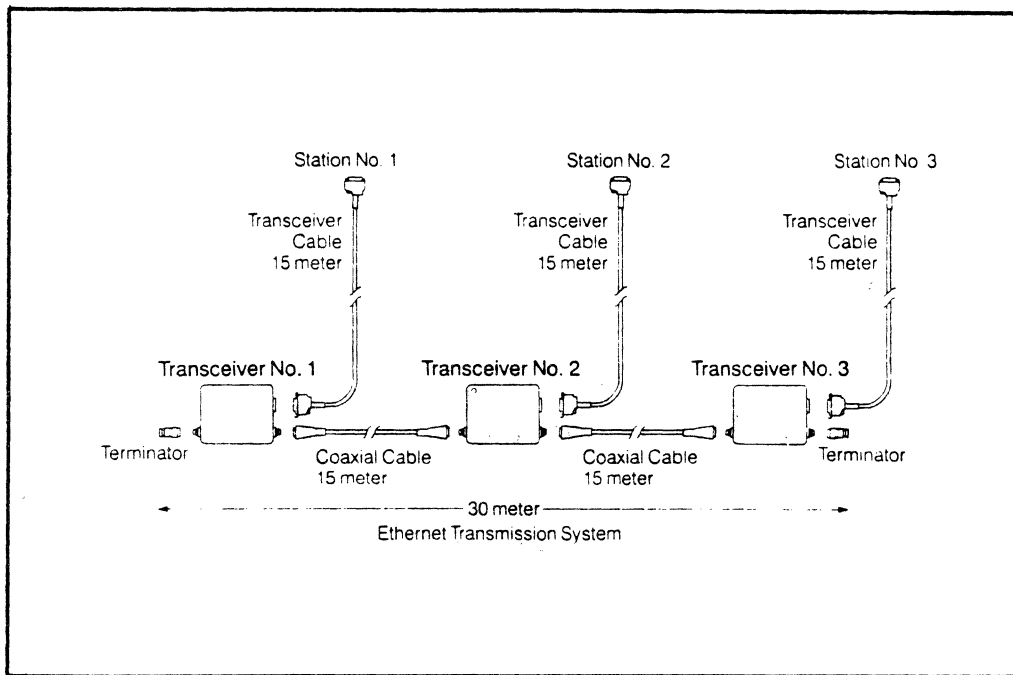


FIGURE 2-2. 3COM ETHERNET TRANSCEIVER STARTER PACKAGE

To minimize EMI (electro magnetic interference), the connectors have internal shields connecting the cable shield to the shell of the connector.

The male cable connector can either be brought out of the wall to the station or mounted on a cover plate providing a bulkhead disconnect at the wall. When mounted on the cover plate, it has been referred to as the "Information Outlet." (see **Figure 2-3** below)

Transceiver - The 3Com transceiver is compatible with the DEC, Intel and Xerox Ethernet specification. It makes a high impedance connection to the common coaxial cable and provides electrical isolation between the

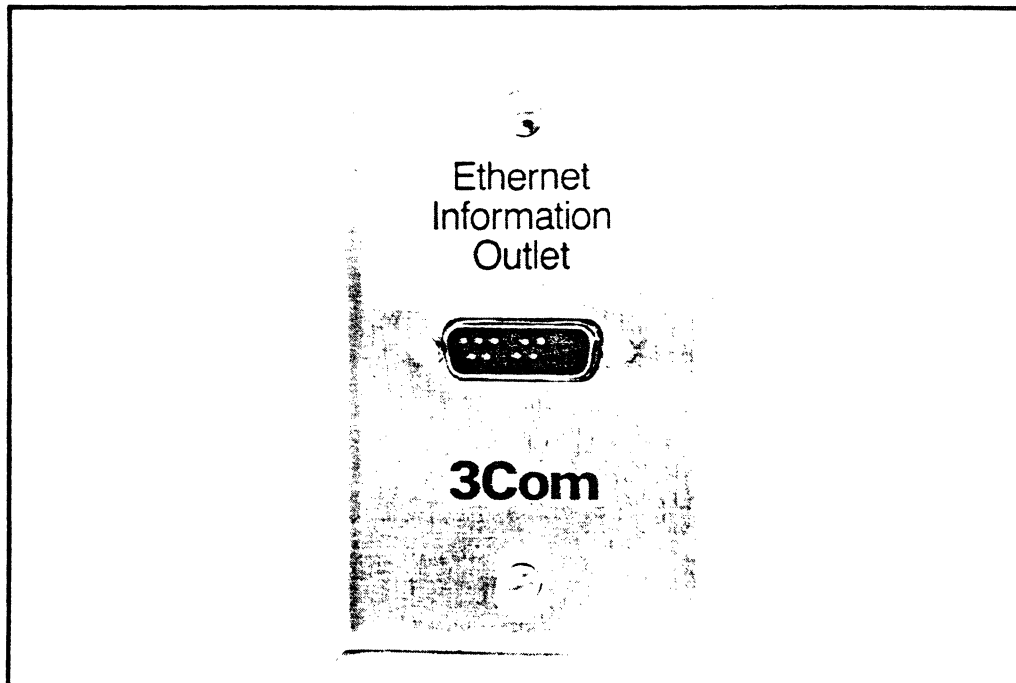


FIGURE 2 - 3. INFORMATION OUTLET

coaxial cable and the twisted pair cable.

The transceiver injects transmit signals from the controller into the coaxial cable. The transceiver also receives signals from the coaxial cable which appear on the receive lead of the transceiver cable with balanced signalling.

The receiver also provides correction for signal distortions caused by traveling through long lengths of coaxial cable.

The collision signal appears if there is a signal present from any other station on the network. When transmitting, this indicates a collision. When **not** transmitting, it indicates the presence of other signals on the network.

Coaxial Cable - The coaxial cable is a 50 ohm cable with multiple shields to minimize susceptibility to strong RF fields.

Cable Connectors - Cable sections are terminated with standard N-series connectors. Rubber boots cover the connectors to prevent multiple connections of the coaxial shield to building grounds - a potential source of ground induced noise into the coaxial shield. Coaxial cable sections are joined by insulated barrel connectors (N-Series female-female adapters).

Terminators - The ends of a coaxial cable segment are terminated with 50

ohm terminators with insulated outside covers.

2.6 ME CONTROLLER SUBSYSTEM

The ME Ethernet Controller interfaces the transceiver to the internal bus of the Multibus system to which it is connected. It performs serial-parallel and parallel-serial conversion, buffering, CRC generation and checking, address recognition, phase encoding and decoding - discussed below. The I/O structure and speed of the processor determine how these functions are partitioned between hardware and software (or microcode) in the Ethernet station.

Buffering: Most processors have bus transfer rates that are unduly stressed by the 10Mbps Ethernet bandwidth, therefore, full packet buffers are provided to keep pace with the bit rate of network traffic.

CRC Generation And Checking: The cyclic redundancy code (CRC) uses the 32 bit polynomial from the U.S. Department of Defense Autodin II system. The CRC function is implemented in hardware on the ME.

Address Recognition: The controller watches every packet that passes to determine whether to accept the packet, based on its destination address. The ME Controller implements address recognition in hardware to minimize CPU overload.

Phase Encoding, Decoding, and Transceiver Interface: Manchester

encoding is used for data transmission on the Ethernet. It has a 50% duty cycle. The first half of a bit cell contains the complement of the bit and the second half of the bit cell contains the bit.

Phase Encoding is done in the controller by exclusive-ORing the clock with the data. (Decoding is also performed in the controller. Partitioning of encode-decode functions into the controller rather than the transceiver minimizes wires to the transceiver while minimizing transceiver size and power dissipation.)

Phase Decoding in the ME Controller is done by an analog phase-locked loop technique. This technique has the advantage of tolerating more phase jitter than alternative techniques in use, twice the tolerance of a typical one-shot decoder and four times the tolerance of a typical digital state-machine decoder.

The **Transceiver Interface** contains line drivers and receivers.

**CHAPTER 3
PHYSICAL DESCRIPTION**

3.1 ENVIRONMENT

The ME Controller interfaces a Multibus compatible processor to an Ethernet transceiver.

The ME Controller plugs into the same Multibus backplane as the processor and resides in the same enclosure with it. An Ethernet transceiver cable (approximately 50 feet long) connects the ME Controller to the Ethernet transceiver. The Ethernet transceiver in turn taps directly into the Ethernet coaxial cable.

According to the International Standards Organization Open Systems Interconnection Reference Model, the ME Controller performs part of both the physical and link layer services, the first and second of seven

layers of service (see Figure 3-1 below).

The two main functions of the ME in the link layer of the ISO Model are:

1. Data Encapsulation

- o framing (frame boundary delimitation)
- o addressing (handling of source and destination addresses)
- o error detection (detection of physical channel transmission errors)

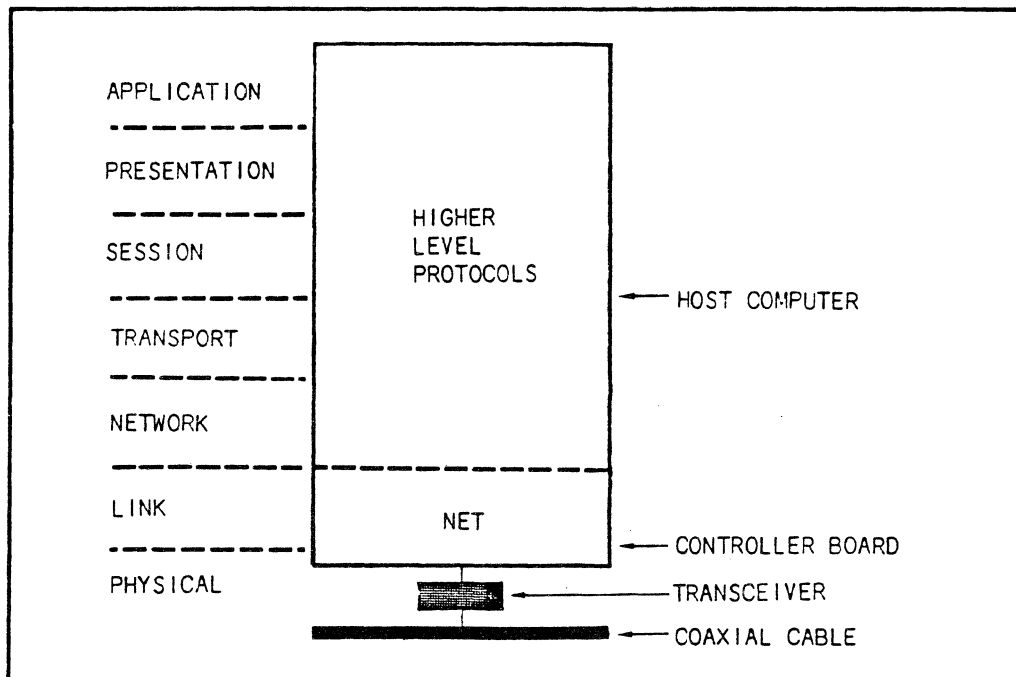


FIGURE 3-1. ISO PROTOCOL HIERARCHY

2. Link Management

- o channel allocation (collision avoidance)
- o contention resolution (collision handling)

In the physical layer of the ISO model, the controller performs preamble generation/removal and bit encoding/decoding (between binary and phase-encoded form).

3.2 PACKAGING

The ME Controller is a single multibus compatible PC board whose overall dimensions are 12 inches x 6.75 inches. It uses one Multibus

The PC layout design rules followed are:

- o Ten mil traces
- o Ten mil air gaps
- o At most two traces between IC leads
- o Four layer pcb
- o Intact internal voltage and ground planes

The ME Controller can be used with any transceiver that conforms to the DEC-Intel-Xerox Ethernet specifications.

3.3 BLOCK DIAGRAM

The major ME Controller components are shown in Figure 3-3 below.

The ME Controller presents to the Multibus a full 16 bit interface. All bus transfer instructions to a 16 bit slave device are implemented on this controller.

The 3Com Multibus Ethernet Controller is memory-mapped and therefore appears as 8K bytes of memory on the Multibus; 4K is used for two receive buffers and 2K is used for one transmit buffer. There are four bytes of

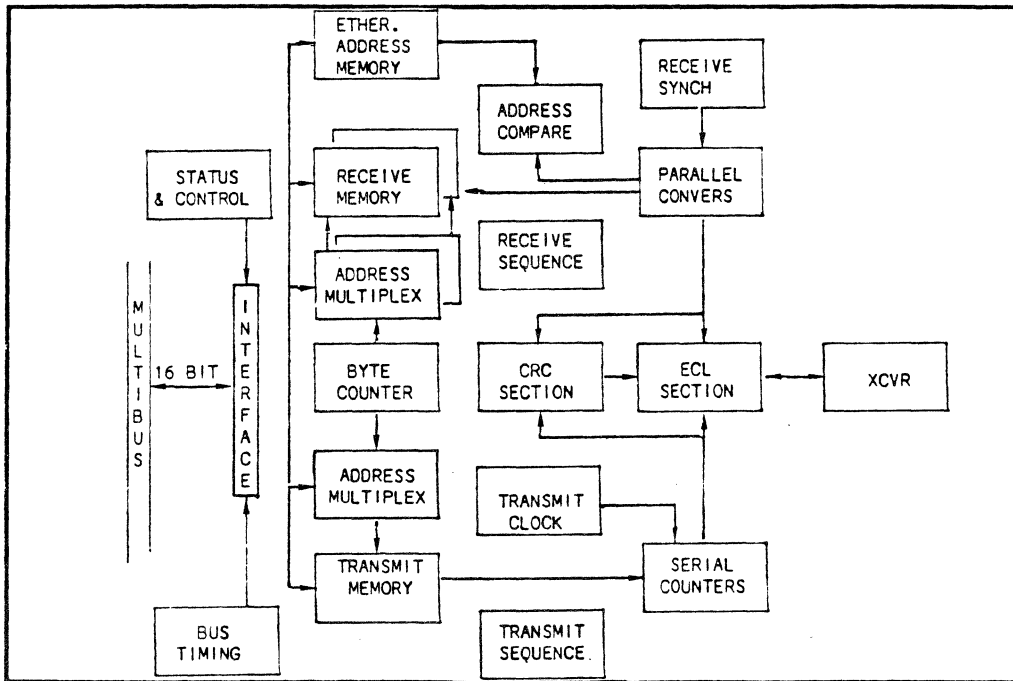


FIGURE 3-3. ME CONTROLLER FUNCTIONAL BLOCK DIAGRAM

control and two words of status in the lowest 2K of memory. The actual device address is switch selectable on any 8K boundary.

On transmit, the processor loads the packet to be sent into the TRANSMIT MEMORY (see Figure 3-3 above). Once the transmit bit is set in the control byte no further processor intervention is required. Successive bytes are taken from the TRANSMIT MEMORY and placed into the SERIALIZER where they are converted to a serial bit stream. The serialized data is fed to both the CRC (Cyclic Redundancy Check) and to the ECL SECTION which does a level conversion to be compatible with the Ethernet Transceiver. The CRC is automatically appended to the end of the transmit data packet.

On receive, the processor must first make available to the ME one or both of the RECEIVE MEMORIES available (see Figure 3-3 above) by setting the proper control bits. Since hardware address recognition is incorporated into this design, the receive packet is further qualified. The ME can be hardware configured to receive only packets with correct physical, multicast, and/or broadcast addresses. After address recognition, the Ethernet data enters the ECL SECTION where a phase lock loop generates a proper synchronizing clock from the incoming data. Also the ECL SECTION does a level conversion to TTL signals. The serial data is sent to the PARALLEL CONVERSION logic (and converted to a byte format) and to the CRC SECTION. The CRC is calculated and compared to the CRC code appended to the end of the packet by the transmitting station. If a CRC error occurs, (Frame Check error) a flag is set in the first byte of the RECEIVE MEMORY.

CHAPTER 4

ME PROGRAMMING

The following chapter is important for development of software drivers.

4.1 MEMORY ALLOCATION

The ME occupies 8K bytes of the Multibus 24 bit address space. See Figure 4-1 below. Switches on the controller select the base address of the ME memory called MEBASE. MEBASE must be aligned on an 8K byte

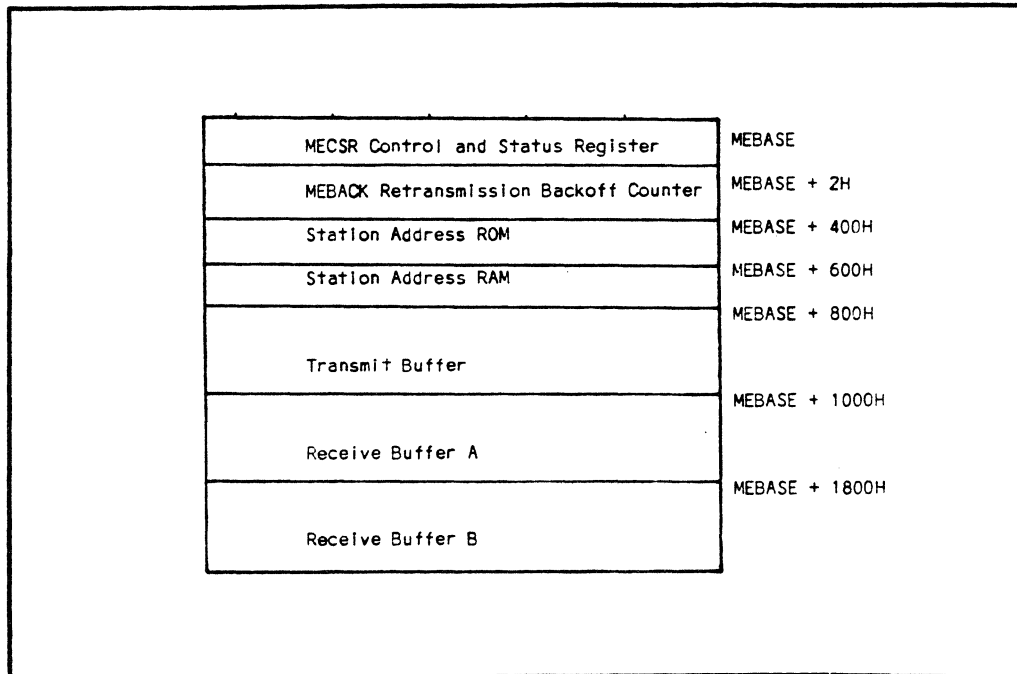


FIGURE 4-1. ME BUFFER ORGANIZATION 8K BYTES, ADDRESSES IN HEX

boundary. The controller partitions its memory into six regions as follows: (Also see Figure 4-1 above.)

- | | |
|----------------------------------|------------------|
| 1. Control region | (MEBASE) |
| 2. Station address ROM | (MEBASE + 400H) |
| 3. Station address RAM | (MEBASE + 600H) |
| 4. One 2K byte transmit buffer | (MEBASE + 800H) |
| 5. First 2k byte receive buffer | (MEBASE + 1000H) |
| 6. Second 2k byte receive buffer | (MEBASE + 1800H) |

Software uses the registers in the control region to manipulate the controller and read its status. There are two registers in the control region:

1. MECSR, the control and status register, at MEBASE
2. MEBACK, the retransmission backoff counter, at MEBASE+2.

NOTE: The byte-ordering switch, TRB (See section 5.1) will affect the ordering of the bytes in both MECSR and MEBACK. If the TRB switch is OFF, MECSR and MEBACK are as shown in Figure 4-2. If the TRB switch is ON the bytes of MECSR and MEBACK are reversed.

See Figure 4-2 below.

The station address ROM, at MEBASE+400H, holds the six byte station address of the controller. The address recognition circuitry uses the station address RAM, at MEBASE+600H, to compare the destination address of packets from the ether when deciding whether to accept a packet. Software must write the station address into the RAM and "give" the address to the controller by writing one into AMSW. Any further references to the address memory are ignored until reset.

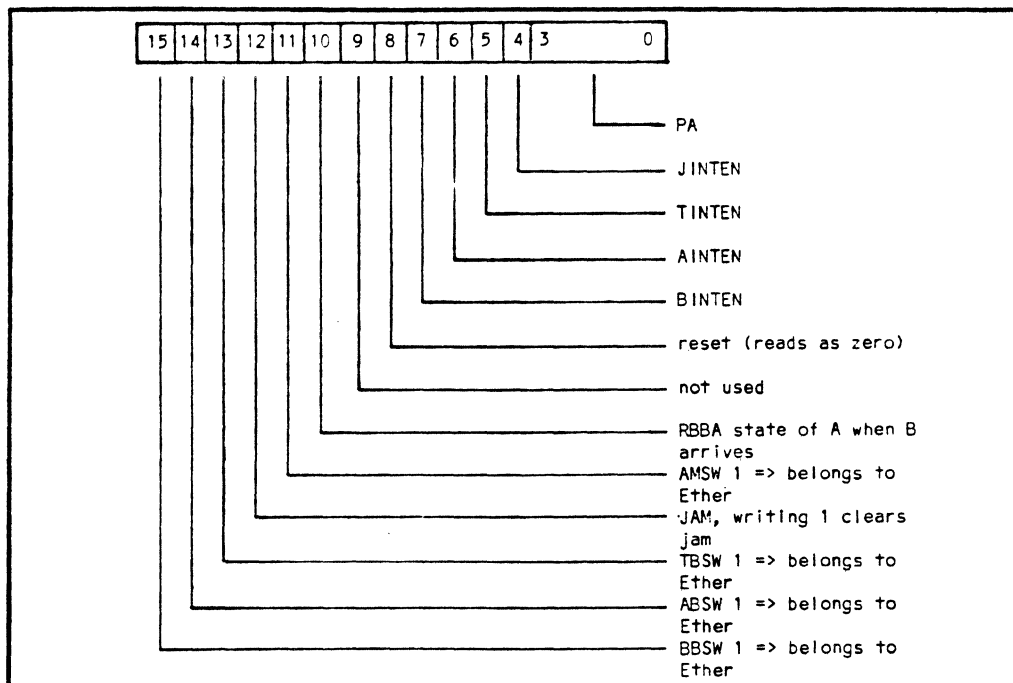


FIGURE 4-2. MECSR=MEBASE, THE MULTIBUS ETHERNET CONTROL AND STATUS REGISTER

4.2 TRANSMIT/RECEIVE

The transmit buffer starts at $MEBASE+0x800$ (see Figure 4-1). The first word of the buffer is MEXHDR, the transmit buffer header, as shown in Figure 4-3 below. To transmit, align the packet in the buffer so the last byte of the packet coincides with the last byte of the transmit buffer. Set up MEXHDR to contain the offset of the first byte of the packet. Finally, set TBSW to one. As long as TBSW remains one, the transmit buffer is busy and belongs to the controller; (any reference to it is ignored). When TBSW becomes zero, transmission is complete and the transmit buffer is available to software once again. NOTE: The TBSW bit

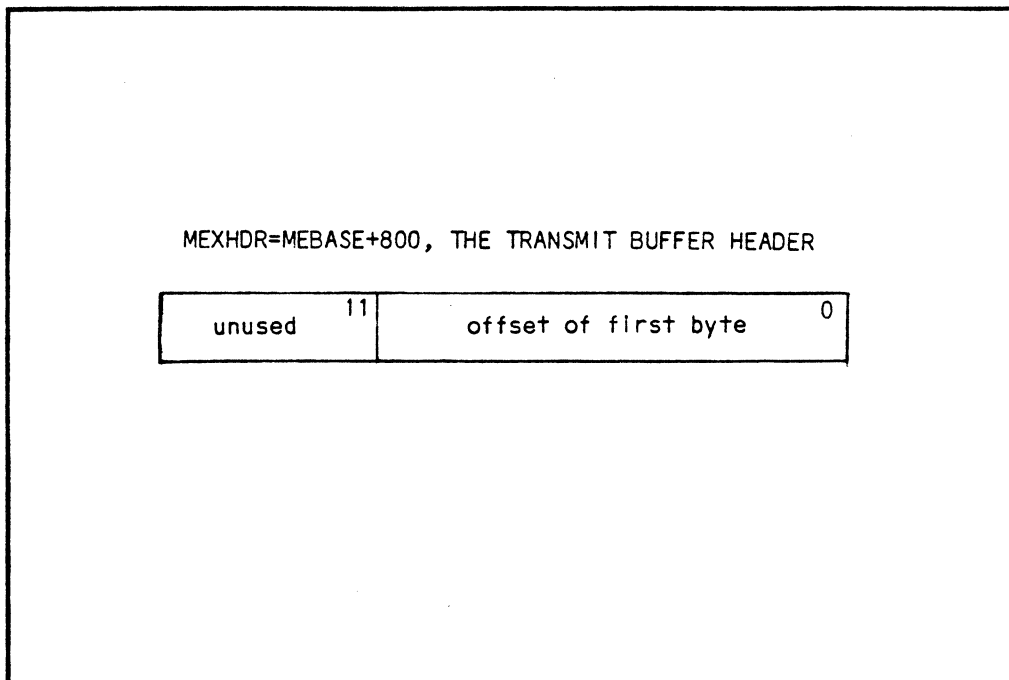


FIGURE 4-3. MEXHDR=MEBASE+800, THE TRANSMIT BUFFER

remains "1" during all JAM processing and will only become "0" again after the packet is successfully transmitted.

In the event of a collision, the controller sets JAM to one. To retransmit the packet, software must write the two's complement of the number of slot times to delay into MEBACK; then write a one back into JAM (the JAM bit is cleared by writing a one into it). Writing into MEBACK when JAM is not set produces unpredictable results.

The ME provides two buffers to receive packets from the ether. Software controls these buffers with two bits in MECSR; one for each buffer. ABSW controls receive buffer A which starts at MEBASE+1000H. BBSW controls receive buffer B which starts at MEBASE+1800H. (See Figure 4-1 on Page 4-1) To receive a packet in A, set ABSW to one. While ABSW remains a one, any reference to A will be ignored. ABSW will remain one as long as the buffer belongs to the controller; when ABSW becomes zero the buffer contains a packet and belongs to the software. To receive a packet in B, the software manipulates BBSW similarly. If both ABSW and BBSW are zero after giving both receive buffers to the controller, RBBA helps the software decide which buffer has the oldest packet: If RBBA is zero, then the packet in A is older than the packet in B. If RBBA is one the packet in B is older than the packet in A.

The header (first) word of both receive buffers are called, MEHDR

and MEBHDR, see **Figure 4-4** below. The received packet minus the preamble starts immediately after the buffer header. After a packet arrives, the controller writes a status word in the buffer header. The low order eleven bits specify the offset of the first free byte after the packet. The high order bits contain various status flags, starting with the sign bit: fcs error, broadcast, range error, station address match, and framing error.

NOTE: In **Figure 4-4** the "address match" status bit is "inverted". That is, the status bit is zero (0) if the destination address matches the station address, or one (1) if it does not match.

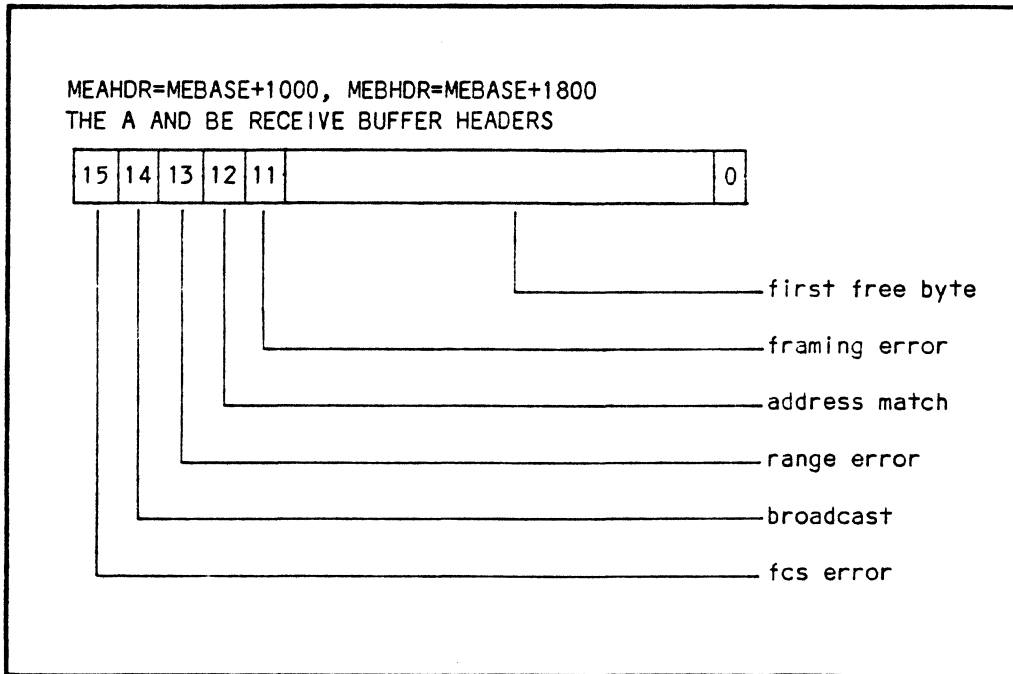


FIGURE 4-4. A AND B RECEIVE BUFFER HEADERS

Similarly, the "broadcast" status bit is zero (0) if the destination address matches the broadcast address, or one (1) if it doesn't match.

PA allows the software to select which packets the controller will accept on receive. The controller classifies all packets on the ether, as follows:

All	the universe of packets on the ether
Range	runt and oversized packets
FCS	packets with frame check sequence errors
Frame	packets with alignment errors
Mine	packets with destination address = station address
Multi	multicast packets ¹
Broad	broadcast packet ²
Errors	range+fcs+frame

¹NOTE: There is no status bit indicating a multicast packet that is not a broadcast packet. Such a packet is identified by a broadcast status bit equal to "1" (false) and the multicast bit in the destination address being "1" (true). Multicast packets will only appear in receive buffers if the receiver is enabled to accept them. The multicast bit of the destination address is the least significant (low order) bit of the first byte of the packet. In a receive buffer, this is the byte immediately following the buffer header word.

²NOTE: Broadcast is a special case of multicast. Enabling the receiver for multicast (modes 3,4,5) includes the broadcast address.

The software sets PA to receive only selected classes of packets.

- 0 all
- 1 all - errors
- 2 all - fcs - frame
- 3 mine + multi
- 4 mine + multi - errors
- 5 mine + multi - fcs - frame
- 6 mine + broad
- 7 mine + broad - errors
- 8 mine + broad - fcs - frame

Jumpers on the board select a single interrupt level for the ME controller. Setting JINTEN enables interrupts when JAM is set. Setting TINTEN enables interrupts when TBSW is zero. Setting AINTEN and BINTEN enables interrupts when ABSW and BBSW are zero respectively. The ME interrupts whenever any of the masked bits meet the conditions stated above; in particular, interrupts are not edge triggered, but level triggered. An interrupt routine should not leave an interrupt enabled unless it turns around the buffer that caused the interrupt. If the interrupt routine does not turn around the associated buffer and leaves the interrupt enabled, the controller will immediately reinterrupt as soon interrupt routine completes.

4.3 PROGRAMMING ANOMALIES

MECSR and MEBACK are replicated all through the control region. There is no method to read the value of MEBACK. If the software attempts to read MEBACK, it will read MECSR instead.

The station address ROM is the first six bytes of an eight byte block; this eight byte block is replicated throughout the region.

The station address RAM is the first six bytes of an eight byte block; this eight byte block is replicated throughout the region.

The interrupt enables for the transmit or receive buffers must NOT be set until that buffer has been assigned to the Ether. Enabling the interrupts before that time will result in an immediate interrupt going to the processor. Part of the interrupt service routine must include disabling the interrupt enables; otherwise, the processor will stay interrupted. The JAM interrupt can be left in the enabled state as long as desired.

NOTE: The packet stored into the packet buffer by the processor does not include the preamble or FCS, but does include the Ethernet data link layer header fields (destination address, source address, and type field) along with the data. Since the maximum legal packet size on the

Ethernet is 1514 (excluding the frame check sequence (FCS) field), at least 532 bytes of each transmit packet buffer will always be unused. Conversely, since the minimum legal packet size is 60 bytes (again excluding the FCS field), at most 1986 bytes of each transmit packet will be left unused. It is the responsibility of the driver software to insure that minimum and maximum packet size requirements are observed.

4.4 OPERATION

The transmitter and receiver operate independently, giving the programmer the illusion that the Ethernet is a full-duplex device. In fact, only one packet may exist on the coaxial cable at a time.

Whenever the Ethernet channel is not in use, the packet supplied to the transmitter is transmitted as quickly as possible. The ME hardware separates packets by the minimum packet spacing (9.6 - 10.2 microseconds).

Whenever a packet appears on the Ethernet, (except while transmitting) it is read into a buffer previously supplied to the receiver.

One aspect of the transmit process, the **binary exponential backoff** algorithm, is implemented by the ME hardware and requires software support only for random number generation. When the controller detects a collision it sets the JAM bit in the transmit control register which causes a 32-bit jam signal to be transmitted and, if the JINTEN bit is set, an interrupt occurs. The software must store a random number into the MEBACK register to cause a delay (back off) of the appropriate time after a collision. After the delay, the same packet is retransmitted. If a collision happens again, the cycle repeats.

The delay time is an integral multiple of a slot time (512 bit-times

or 51.2 microseconds). The integral multiple is chosen as a uniformly distributed random integer greater than or equal to zero and less than 2^k (2 to the k^{th} power), where k is either the number of retransmission attempts for the packet being transmitted or 10, whichever is less. This algorithm doubles the mean of the delay time each time a collision occurs, ensuring the stability of the Ethernet even under extreme loading.

If the number of retransmission attempts exceeds 15 (probably a transmission subsystem malfunction), an error should be reported. The number of retransmissions normally exceeds 15 only when the Ethernet is broken. When this occurs, the only way to get the ME to return the transmit buffer to the processor is to reset the controller by setting bit 8 of MECSR. NOTE: This reset merely gives the memory buffers back to the Multibus (resetting TBSW, ABSW, and BBSW), reset does not affect the contents of any ME memories.

The processor overhead to support backoff in software is minimal. Studies show that Ethernet packets typically experience collisions less than 0.03% of the time.

CHAPTER 5

INSTALLATION AND CONFIGURATION CONSIDERATIONS

5.1 INSTALLATION CHECKLIST

- BEFORE OPENING THE SHIPPING CARTON, inspect it for damage or water stains, if so,
 - Write a brief description of the damage on the bill of lading, and
 - Request that the carrier's agent be present when the carton is opened.
 - Save the carton and packing materials to show the carrier in case the controller was damaged. The carrier is liable for shipping damage.

- Unpack the ME Controller module gently by removing the foam packing material.

___ Verify that all components are present. (See Figure 5-1 below.)

One PC Board

One Cable

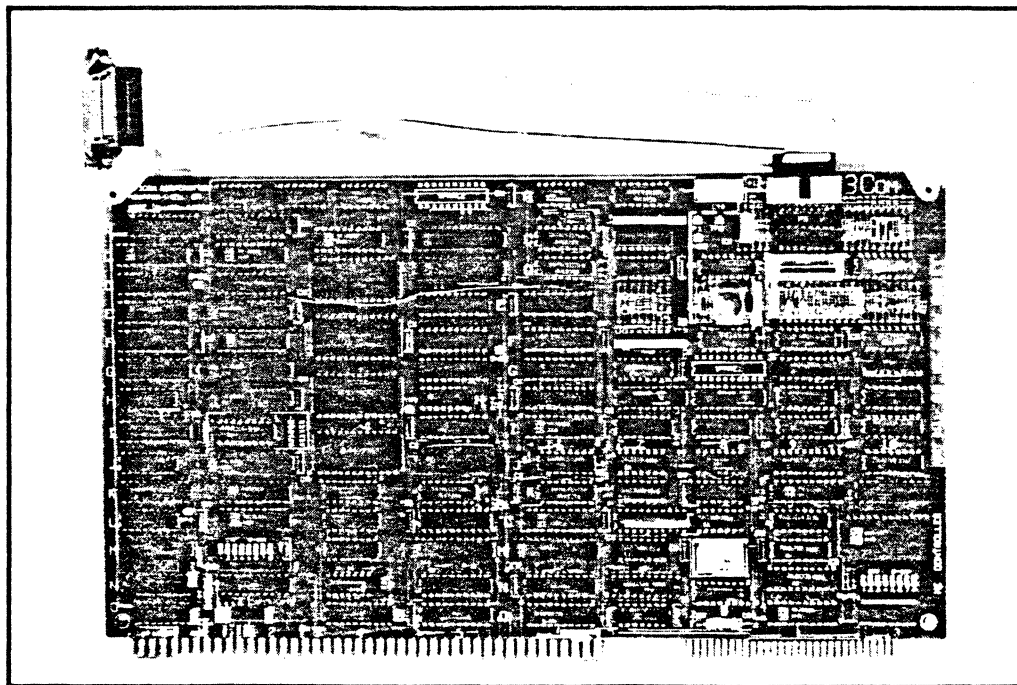


FIGURE 5-1. ME CONTROLLER PARTS

configure the ME Controller on the I/O side, connect the I/O jumpers, IORC and IOWC.

--- Locate the address switches labeled ADR13/ and ADR17/. (See Figures 5-2, 5-3, 5-4). The switches labeled ADR13/ specify the most significant bit for 20-bit addressing. Switch ADR17/ specifies the most significant bit for 24-bit addressing. If the factory default settings (80000H) are satisfactory, go the next step, otherwise change the switch settings to meet your specific needs.

--- Locate the interrupt jumpers labeled INTERRUPTS. (See Figures 5-2, 5-3). Set the interrupt switches to the interrupt level desired: From 0=highest, to 7=lowest.

(Installation procedures continued after figures and tables on following pages.)

TABLE OF SWITCH AND JUMPER SETTINGS

Description, Switches	Factory Default	Switch Name
Buffer Memory Base Address	80000 H	Memory ADR17/-ADR0D/
Byte ordering	OFF	TRB
	(Most significant byte at EVEN address)	
	e.g. UNLIKE the 8086!	

Description, Jumpers	Factory Default	Jumper Name
24-bit Addressing	Inserted	JP1
20-bit Addressing	Inserted	JP2

Memory Configuration

	JP1	JP2
24-bit Addressing	Inserted	Inserted
20-bit Addressing	Out	Inserted
16-bit Addressing	Out	Out

- ___ Turn the DC power to the backplane OFF.

- ___ Plug the serial cable supplied into the 14-pin connector on the back edge of the ME. The Notched Position of the Serial Cable 14-pin leader has "component-side" orientation.

- ___ Plug the Multibus board into the Multibus backplane.

- ___ Plug the Ethernet transceiver cable (not supplied) into the subminiature - 0 end of the serial cable.

- ___ Attach the lug at the end of the pigtail on the serial cable (near the board) to any convenient chassis ground. PROPER GROUNDING IS CRITICAL TO THE PROPER OPERATION OF THE ETHERNET.

5.2 ETHERNET ADDRESS CONSIDERATIONS

Each station on an Ethernet has a 48-bit Ethernet address associated with it that is unique across all Ethernets in the universe. The station address is assigned by the manufacturer of the station. For stations assembled from components from multiple manufacturers, the assignment of Ethernet addresses is ambiguous. This section describes the conventions to be followed to maintain unique Ethernet addresses under various configurations of components.

Each ME Controller manufactured by 3Com is shipped with a unique Ethernet address. The address is contained in a special address recognition PROM, and is also printed on the PC board in indelible ink. For stations that contain only a single ME Controller, the station address is the one supplied with the ME Controller. For stations that contain multiple ME Controllers, such as gateways, one of the controllers should be arbitrarily selected to contribute its Ethernet address as the station address. The hardware address recognition in the station must be programmed to respond to the chosen Ethernet address in all the ME Controllers attached.

It may become necessary in order to analyze a hardware problem to swap boards between stations at a user site. In that case it is the user's option either to have the Ethernet address to follow the board, or to disassociate the board from the address in software. The advantage of the former is that the address recognized always matches the address

printed on at least one of the controllers in the station. The advantage of the latter is the ability to swap hardware around without changing any station name/station address directory entries for the network.

If a board is sent back to the factory for repair, the board that comes back may not have the same Ethernet address printed on it. The customer has the option of using either the new Ethernet address or the old. Both addresses are allocated to the customer and are under the customer's control.

It is the customer's responsibility to manage the allocated Ethernet addresses. Addresses can be assigned in any manner as long as none are duplicated on more than one station. The recommended practice is to add each ME Controller Ethernet address to a site-wide pool of addresses maintained independently of the hardware. That way, ME hardware can be moved around among stations or sent back to 3Com for repair without affecting any software.

CHAPTER 6

FACTORY WARRANTY AND REPAIR POLICY

Any 3Com product which fails within 90 days from date of shipment will be repaired or replaced by 3Com free of charge providing, in the opinion of 3Com, the product has not been subject to improper electrical, mechanical, or thermal stress. The customer should send the defective item to 3Com stating the nature of the fault and quoting the date and invoice number of the original shipment. Freight charges from customer to 3Com must be paid by the customer. Freight charges from 3Com to U.S. customers are paid by 3Com using UPS. Foreign customers must pay freight both ways. Replacement or repair under warranty will normally be completed within seven days of receipt at 3Com.

After 90 days, a factory repair service is available for the 3C400 MULTIBUS Ethernet Controller. The charge is fixed at \$175 for each controller regardless of the extent of the repairs concerned. If testing of the controller shows no failure, or if the controller has been modified or damaged by the customer attempting repairs or if it has been subject to improper electrical, mechanical, or thermal stress, it will be returned to the customer in the same condition as received and a \$100 charge assessed. Controllers repaired under the fixed-price repair service will be warranted for 90 days from the date of shipment of the repaired controller.

In order to achieve a quick turnaround, 3Com may ship a controller different from that received. In all cases, the controller shipped from 3Com will be given a new Ethernet address to avoid any possible duplication of addresses. The customer may use the Ethernet address originally shipped with the controller, or with the repaired controller, or both.

APPENDIX A
ORDERING INFORMATION

Model Number	Description
3C100	Ethernet transceiver
3C110-015	Transceiver cable, 15 meter
3C400	Multibus Ethernet Controller
3C420	Reference Manual
3C440	Multibus Ethernet Starter Package

Warranty

Any item which falls within 90 days of shipment will be repaired or replaced by 3Com free of charge. After 90 days a billable factory repair service is available.

APPENDIX B
BIBLIOGRAPHY

1. DEC-INTEL-XEROX, "The Ethernet, A Local Computer Network, Data Link Layer and Physical Layer, Version 1.0", September 30, 1980.
2. Robert M. Metcalfe and David R Boggs, "Ethernet: A Distributed Packet Switching for Local Computer Networks", CACM, 19:17, July 1976, pp. 395 - 404.
3. John F. Shoch and Jon A. Hupp, "Performance of an Ethernet Local Network - A Preliminary Report", Local Area Communications Network Symposium, Mitre and NBS, Boston, May 1979.

3Com

computer communication compatibility

APPENDIX C

SAMPLE DRIVER

The following driver was written to support the 3Com Ethernet Multibus board in the Intel 86330 machine. 3Com makes no warranties, either express or implied, with respect to this software, its quality, performance, or fitness for any particular purpose. It is included here "as is" for example purposes.

```

/* *****
 * UNET driver for the 3COM Ethernet Multibus board
 *
 * The following routines are defined:
 *
 * ethinit - called by IP when TCP (or another protocol) is started.
 * ethfin - called by IP when all protocols have terminated.
 * ethgetb - called by IP to obtain a buffer for transmission.
 * ethsend - called by IP to transmit a packet.
 * ethisend - called by the IP router to transmit a packet.
 * ethrint - called by the kernel when a packet is received.
 *
 * ethasend - called by ethsend to fill the transmit buffer.
 * etharcv - called by ethrint to get bytes from a receive buffer.
 * ethout - called to output a word to the controller board.
 * ethin - called to input a word from the controller board.
 * random - compute a random number
 * ethdebug - called to print a debug message.
 * *****/

#include "param.h"
#include "dir.h"
#include "user.h"
#include "map.h"
#include "buf.h"
#include "intr.h"
#include <UNET/unetconfig.h>
#include <UNET/unetio.h>
#include <UNET/sys/unetpio.h>
#include <UNET/uversion.h>

#define ETHDEBUG 1
int _ethdebug;
#if ETHDEBUG
#define ethdebug(m,n) if (_ethdebug) ethprint(m,n); else
#else
#define ethdebug(m,n)
#endif
long eth_bad; /* count of bad packets received */
long eth_notip; /* count of good non-IP packets received */
long eth_good; /* count of good IP packets received */
int eth_repeat; /* debugging repeat count */
int eth_xlength; /* debugging transmit length */

/* ME controller addresses */
#define MEBASEPN 0x1E0 /* ME memory base page number (= 900K) */
#define MMPGSIZE 2048 /* 86/330 mmu page size */
#define MEBASE 0 /* ME memory base address */
#define MECR (MEBASE) /* control and status register */
#define MEBACK (MEBASE+2) /* retransmission backoff counter */
#define MEADROM (MEBASE+0x400) /* station address (in ROM) */
#define MEADRAM (MEBASE+0x600) /* station address (in RAM) */
#define MEXHDR (MEBASE+0x800) /* transmit buffer header */
#define MEAHDR (MEBASE+0x1000) /* receive buffer A header */
#define MEBHDR (MEBASE+0x1800) /* receive buffer B header */

```

```

/* Bits in status and control register MECSR */
#define BBSW 0x8000 /* write 1 to release, read 0 when packet received */
#define ABSW 0x4000 /* write 1 to release, read 0 when packet received */
#define TBSW 0x2000 /* write 1 to transmit, read 0 when complete */
#define JAM 0x1000 /* read 1 if collision, write 1 to retransmit */
#define AMSW 0x0800 /* write 1 to give station address to controller */
#define RBBA 0x0400 /* read 1 if B older than A (only if ABSW=BBSW=0) */
#define RESET 0x0100 /* write 1 to reset controller */
#define BINTEN 0x0080 /* write 1 to interrupt when packet received in A */
#define AINTEN 0x0040 /* write 1 to interrupt when packet received in B */
#define TINTEN 0x0020 /* write 1 to interrupt when transmission complete */
#define JINTEN 0x0010 /* write 1 to interrupt when collision occurs */
#define PA 0x0007 /* receive criteria:
                                0 all
                                1 all - errors
                                2 all - fcs - frame
                                3 mine + multi
                                4 mine + multi - errors
                                5 mine + multi - fcs - frame
                                6 mine + broad
                                7 mine + broad - errors
                                8 mine + broad - fcs - frame
*/
/* Bits in receive packet header MEAHDR or MEDHDR */
#define FCSerr 0xB000 /* data CRC error */
#define BROADCAST 0x4000 /* broadcast packet */
#define RANGerr 0x2000 /* range error */
#define ADRmatch 0x1000 /* address match */
#define FRAMerr 0x0800 /* framing error */
#define FREEmask 0x07FF /* free pointer mask */
#define STATmask 0xFB00 /* status mask */
#define NORMALhdr 0x4000 /* normal receive status */

#define INTlevel 4 /* Multibus interrupt level for the board */

#define SENDHEAD 14 /* dest(6) + source(6) + type(2) */
#define RECHEAD 14 /* dest(6) + source(6) + type(2) */
#define RECTAIL 4 /* CRC(4) */
#define SENDLIM SENDHEAD+NMAXSEG /* transmit buffer length */
#define RECLIM RECHEAD+NMAXSEG+RECTAIL /* receive buffer length */

char *mapwork();
dev_t ethdev;
int (*ethrcv)();
int (*ethwin)();
int ethadr[3];
int ethIplen;
int ethcntl;
char *ethsndptr;
int ethsndcnt;

char IPtype[2] = {0x08, 0x00}; /* IP packet type */

/*****
 * ethinit - called by IP when TCP (or another protocol) is started.
 *****/
ethinit (dev, rcv, win)

```

```

register dev_t dev;
int (*rcv)(), (*win)();
{
    ethdebug ("Ethinit, dev =", dev);
    ethdev = dev;
    ethrcv = rcv;
    ethwin = win;
    ethout (MECSR, RESET); /* reset controller */

    /* Transfer station address from controller ROM to RAM */
    ethout (MEADRAM, ethadr[0] = ethin(MEADROM));
    ethout (MEADRAM+2, ethadr[1] = ethin(MEADROM+2));
    ethout (MEADRAM+4, ethadr[2] = ethin(MEADROM+4));
    ethout (MECSR, AMSW);

    /* Allow reception of "mine + broad - error" packets */
    ethout (MECSR, PA | ABSW | BBSW);
    ethout (MECSR, ethcntl = (PA | AINTEN | BINTEN));

    return (NMAXSEG*2); /* advertise 2 buffers */
}

/*****
 * ethfin - called by IP when all protocols have terminated.
 *****/
ethfin (dev)
register dev_t dev;
{
    ethdebug ("Ethfin, dev=", dev);
    ethout (MECSR, RESET);
    ethrcv = 0;
    ethwin = 0;
    return (1);
}

/*****
 * ethgetb - called by IP to obtain a buffer for transmission.
 *****/
caddr_t
ethgetb (dev, count)
register dev_t dev;
register int count;
{
    if (count > NMAXSEG) {
        u.u_error = E2BIG; /* Packet too big */
        return ((caddr_t)0);
    }
    /* Pad the packet with zeros if it has fewer than 46 data bytes */
    if (count < 46)
        count = 46;
    /* Pad the packet with an extra byte if length is odd */
    if ((count & 1) != 0)
        count++;
    /* Transmit the packet */
    count += SENDHEAD;
}

```



```

    ethsndcnt = count;
    ethsndptr = MEXHDR+0x800-count;
    ethout(MECSR, PA); /* turn off receive interrupts */
    ethsndptr = mapwork(MEBASEPN+(((int)ethsndptr)>>11)) +
        ((int)ethsndptr&(MMPGSIIZE-1));
    return (&ethsndptr[SENDHEAD]);
}

/*****
 * ethsend - called by IP to transmit a packet
 *****/
ethsend (dev, buffer, count, lohigh, loclow)
dev_t dev;
char *buffer;
int count;
long lohigh;
long loclow;
{
    register int ps, collide = 0;

#ifdef ETHDEBUG
    if (buffer != &ethsndptr[SENDHEAD]) {
        u.u_error = EIO; /* bad buffer address */
        ethout (MECSR, ethcnt1); /* restore rcv interrupts */
        return (1);
    }
#endif

    /* Stuff dest, src, and type at the beginning of the packet */
    {
        char *dad = ethsndptr;
        char *sad = (char *)&lohigh+2;

        dad[0] = sad[1];
        dad[1] = sad[0];
        dad[2] = sad[3];
        dad[3] = sad[2];
        dad[4] = sad[5];
        dad[5] = sad[4];
    }
    bcopy (ethadr, &ethsndptr[6], 6);
    bcopy (IPtype, &ethsndptr[12], 2);
    if (eth_xlength)
        ethout (MEXHDR, 0x800-eth_xlength); /* point to transmit buf */
    else
        ethout (MEXHDR, 0x800-ethsndcnt); /* point to transmit buf */
    /* pxbk("snd", ethsndptr, 10); */
    ps = spl7 ();
    for (;;) {
        ethout (MECSR, ethcnt1 | TBSW);
        /* If a collision occurs, re-transmit */
        while ((ethin(MECSR)&TBSW) != 0) {
            if ((ethin(MECSR)&JAM) != 0) {
                if (++collide > 10) {
                    ethdebug ("Excessive collisions", ethin(MECSR));
                    u.u_error = EIO;
                    splx (ps); /* interrupts OK */
                }
            }
        }
    }
}

```

```

        return(1);
    }
    ethdebug ("Collision!", collide);
    printf ("Collision!");
    ethout (MEBACK, -random(1<<collide));
    ethout (MECSR, ethcnt1 ! JAM);
}
}
if (eth_repeat == 0)
    break;
else
    --eth_repeat;
}
/* Tell UNET that the packet was transmitted OK */
ethIPlen = ((ethsndptr[16]<<8)&0xFF00) | (ethsndptr[17]&0x00FF);
/* printf ("%d.", ethIPlen); */
(*ethwin) (ethdev, NMAXSEG*2);
splx (ps);          /* interrupts OK */
return (0);
}

/*****
 * ethisend - called by the IP router to transmit a packet.
 *****/
ethisend (dev, buffer, count, lohigh, loclow)
{
}

/*****
 * ethrint - called by the kernel when a packet is received.
 *****/
ethrint ()
{
    register int mecsr = ethin(MECSR);

    ethout(MECSR, PA);          /* turn off receive interrupts */
    /* Get data bytes */
    if ((mecsr&RBBA) == 1) {
        if ((mecsr&ABSW) == 0) ethr1 (MEAHDR, ABSW);
        if ((mecsr&BBSW) == 0) ethr1 (MEBHDR, BBSW);
    } else {
        if ((mecsr&BBSW) == 0) ethr1 (MEBHDR, BBSW);
        if ((mecsr&ABSW) == 0) ethr1 (MEAHDR, ABSW);
    }
}

/*****
 * ethr1 - called by ethrint to process a received packet
 *****/
ethr1 (hdr, bsw)
int hdr;
int bsw;
{
    register char *ethrcvptr;
    int hdrvalue = ethin(hdr);

```

```

if ((hdrvalue&STATmask) != NORMALhdr) {
    ethdebug ("Bad packet, hdr =", hdrvalue&STATmask);
    ethout (MECSR, ethcntl | bsw);
    ++eth_bad;
    return;
}
ethrcvptr = hdr+2;
ethrcvptr = mapwork(MEBASEPN+(((int)ethrcvptr)>>11)) +
    ((int)ethrcvptr&(MMPGSIze-1));
/* pxbllk("rcv", ethrcvptr, 10); */

/* Filter out non-IP packets */
if ((ethrcvptr[12] != IPtype[0]) ||
    (ethrcvptr[13] != IPtype[1])) {
    ethdebug ("Non-IP packet, IPtype =",
        (ethrcvptr[13]<<8) | ethrcvptr[12]);
    ethout (MECSR, ethcntl | bsw);
    ++eth_notip;
    return;
}
/* Give packet to UNET */
ethIPlen = ((ethrcvptr[16]<<8)&0xFF00) | (ethrcvptr[17]&0x00FF);
/* printf ("%d.",ethIPlen); */
(*ethrcv) (ethdev, &ethrcvptr[RECHHEAD], ethIPlen);
ethout (MECSR, ethcntl | bsw); /* release buffer to cntlr */
++eth_good;
}

```

```

/*****
 * ethin - called to input a word from the controller board
 *****/
ethin (adr)
register int adr;
{
    switch (adr) {

        /* Swap bytes if handling control information */
        case MECSR:
        case MEBACK:
        case MEXHDR:
        case MEAHDR:
        case MEBHDR:
            return (swapb(xin(adr)));

        default:
            return (xin(adr));
    }
}

```

```

/*****
 * ethout - called to output a word to the controller board
 *****/
ethout (adr, word)
register int adr;
register int word;

```

```

switch (adr) {

/* Swap bytes if handling control information */
case MECSR:
case MEBACK:
case MEXHDR:
case MEAHDR:
case MEBHDR:
    xout (adr, swapb(word));
    break;

default:
    xout (adr, word);
    break;
}

}

/*****
 * random - called by ethsend to generate a random number
 *         between 0 and limit
 *****/
random (limit)
{
    return (limit); /* be lazy for now */
}

/*****
 * ethprint - called by above routines to print a debug message
 *****/
#ifdef ETHDEBUG
ethprint (message, number)
register char *message;
register int number;
{
    char ucbuf[40];
    char unbuf[20];

    udlog (LGKN, 0, uconcat(ucbuf,message,
                           unum(number,10,unbuf),NCP,NCP,40),1);
}
#endif

/*****
 * xin - input from mapped ME memory
 *****/
short
xin(p)
register char *p;
{
    short w;

    w = *(short *) (mapwork (MEBASEPN+(((int)p)>>11))+((int)p&(MMPGSIZE-1)));
    return (w);
}

```

```
/*
*****
*   xout - output to mapped ME memory
*****
xout(p, w)
register char *p;
short w;
{
    int s;

    *(short *) (mapwork(MEBASEPN+(((int)p)>>11))+((int)p&(MMPGSIZE-1))) = w;
}

/*
pxblk(s, a, c)
char *s;
short a[];
int c;
{
    char unbuf[8];
    register i;

    printf(s);
    for (i = 0; i < c; i++)
        printf(" %s", unum(a[i], 16, unbuf));
    printf("\n");
}
*/
```