MOSTEK®

Z80 MICROCOMPUTER SYSTEMS HARDWARE

Operations Manual

SDB-80E SOFTWARE DEVELOPMENT BOARD

SDB-80E

SOFTWARE DEVELOPMENT
BOARD

OPERATIONS MANUAL

MOSTEK®

MICROCOMPUTER SUPPORT

Z80 Software Development Board (SDB-80E)

HARDWARE FEATURES

- ☐ Available as board or complete system
- ☐ 4K bytes of RAM, expandable on board to 16K Bytes
- ☐ Four 8-bit I/O ports with handshake lines
- ☐ Serial ASCII interface (110-9600 BAUD)
- ☐ Fully buffered for system expandability
- ☐ Four counter/timer channels
- ☐ On board capacity from 5K bytes of PROM to 20K bytes of ROM
- ☐ Double euro-card format

SOFTWARE FEATURES

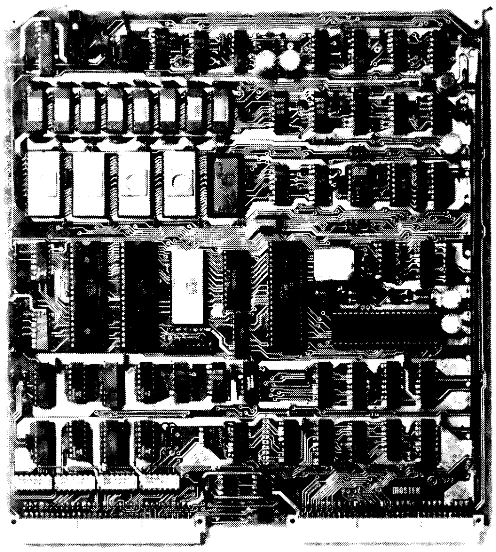
- ☐ 2K x 8 Operating System in ROM (DDT-80)
- ☐ 8K x 8 assembler/editor in ROM (ASMB-80)
- ☐ Channeled I/O for user convenience
- ☐ Double euro-card format

GENERAL DESCRIPTION

The SDB-80 is a stand-alone microcomputer designed by MOSTEK around the advanced Z80 microprocessor familiy. It contains more on-board firmware and RAM memory than any previously offered single board microcomputer, plus all the features of the industrieś most sophisticated microprocessor. This board represents the very latest in state-of-the-art technology by utilizing MOSTEK's new 16K Dynamic RAM memories. The SDB-80 also is the first single board microcomputer to offer a complete package of software development aids in ROM. This 10K byte firmware package is included with the SDB-80 and provides the ability to generate, edit, assemble, load, execute, and debug Z80 programs for all types of applications.

USING THE SDB-80

In addition to functioning as a stand-alone development aid, the SDB-80 is fully expandable through the addition of optional add-on circuit boards. It may also be utilized directly in OEM applications by inserting custom programmed ROM or PROM memories into the sockets provided on the board. For these OEM applications, partially populated versions of the SDB-80 (designated OEM-80) are available without the standard system firmware, and with quantity discounts.



SYSTEM FIRMWARE

A standard feature of the SDB-80 is a complete package of development software aids which are resident in the five MK 34000, 2k x 8 ROM memories located on the board. This firmware includes a sophisticated operating system, debug package, assembler, and text editor. The presence of this software in ROM provides instant access to these development aids, eliminating the time-consuming requirement of loading the software from some perpheral device into RAM.
Another key feature of having the development aid software in ROM is that entire RAM space is available for the user's programs.

Debug (DDT-80) includes:

- ☐ object program Load/Dump
- ☐ Memory or Port Examine/Change
- ☐ Breakpoint/Execute
- ☐ Logical/Physical I/O mapping (with user expandable drivers)
- ☐ Drivers for Standard Peripherals

The Assembler (ASMB-80) includes:

- ☐ 1, 2 or 3 pass operation
- ☐ conditional Assembly
- ☐ Relocatable object module generation
- ☐ Relocatable linking loader
- ☐ Drivers for Silent 700 Cassette

The Text Editor (EDIT-80) includes:

- ☐ Line or character operation
- ☐ Macro commands

ELECTRICAL SPECIFICATIONS

Operating Temperature Range . . . 0 °C to 50 °C

Power Supply requirements (Typical)

+12V ± 5%	175 mA
+ 5V ± 5%	1.5 A
−12V ± 5%	100 mA

Interface Levels . . . TTL Compatible

MECHANICAL SPECIFICATIONS

Extended double Eurocard

Board Size: 250 mm x 233.4 mm x 18 mm

Connector: Dual 64 pin Eurocard Connector DIN 41612 form D; A and C pinned.

COMPATIBLE ADD-ON BOARDS

RAM-80AE Add-on RAM card for the SDB-80E. This card supplies 16k bytes of MK 4027 dynamic RAM Memory.

RAM-80BE Add-on RAM/IO card for the SDB-80E. This card supplies 16k (expandable to 64k) bytes of MK 4116 dynamic RAM Memory, plus 4 fully buffered I/O ports using 2 MK 3881 PIO's. On-card bank switching allows expansion of SDB-80E memory space beyond 64k bytes.

XRAM-80 Expansion Kit for RAM-80BE. Consists of 8-MK 4116 RAMs.

AIM-80E In-circuit-emulation capability is added to the SDB-80 by using the AIM-80 board also provides other debugging capatibilities such as TRACE and SINGLE STEP.

FLP-80E The FLP-80 interfaces the SDB-80 to two soft-sectored floppy disk drives. Full file handling software and firmware is provided with the card.

RIO-80E The RIO-80E includes 2-buffered PIO's, I-UART, I-CTC, and sockets for 16k bytes of MK 2708 PROM.

NON RESIDENT SOFTWARE AVAILABLE

XFOR-80 Fortran IV Cross Assembler. Assembles Z80 programs but is written in Fortran IV. It is useful for persons desiring to perform Z80 assembly in mini-computers such as the PDP-11. It is furnished in Fortran IV source as a card deck or paper tape.

XMDS-80 8080A Cross Assembler. Performs the same function as the Fortran IV Cross Assembler, except that it is designed to be used with an Intel MDS system. It is furnished as an object tape in Intel compatible Hex format.

XMDS-80D This is identical to the XMDS-80 except that it is compatible with Intel MDS systems which use floppy disk. It is furnished as object code on an MDS compatible floppy diskette.

OTHER ACCESSORIES AVAILABLE

PPG-08 PROM Programmer module for programming MK 2708 UV erasable PROM memories. Interfaces directly with the SDB-80. Enclosure included.

OEM USERS CARDS AVAILABLE

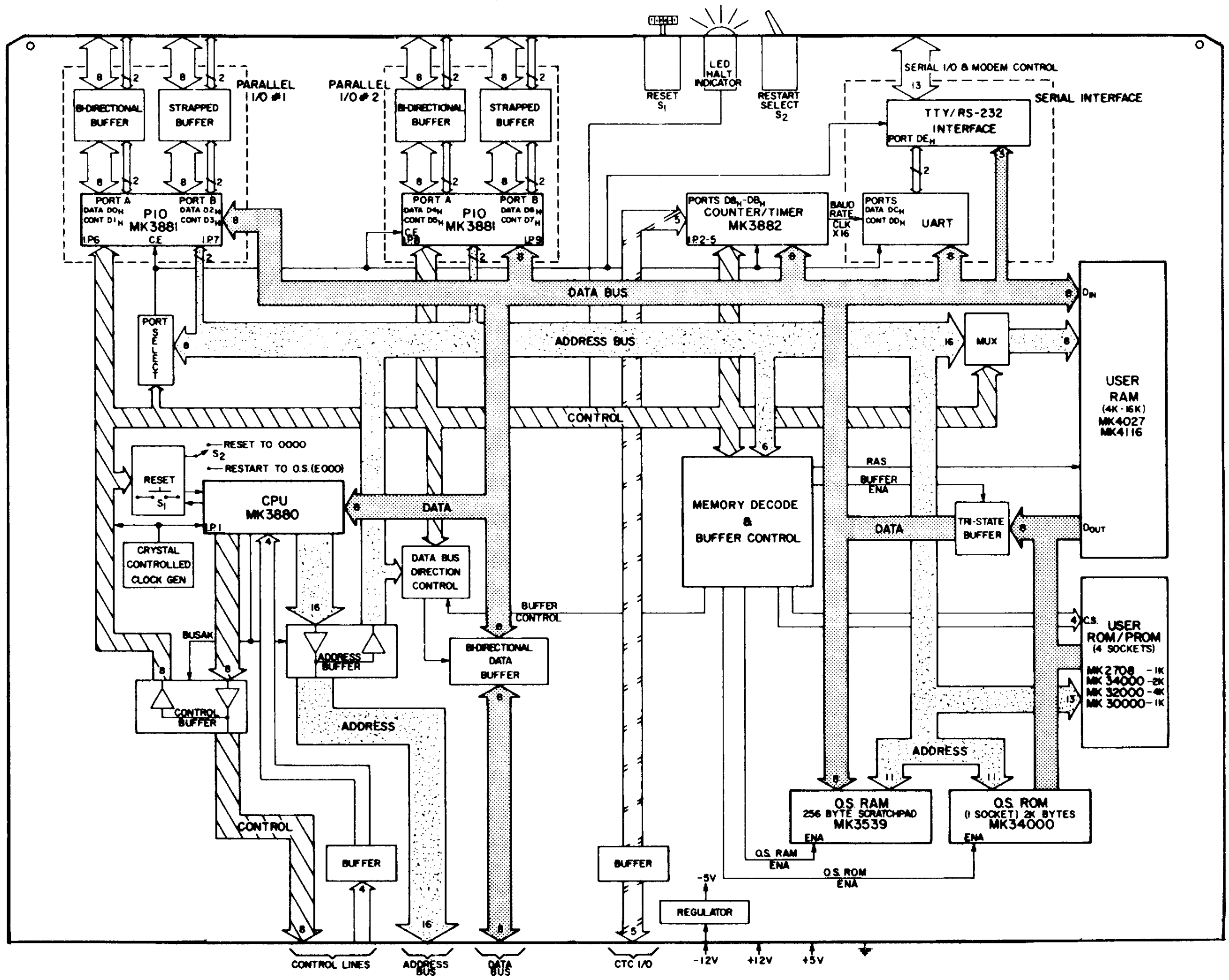
OEM-80E SDB-80E without Software. Available with 4 or 16k RAM Memory. 5 PROM/ROM sockets are free for user programs.

Parallel Universal Display Interface UDI-P This double Eurocard CRT/Keyboard Interface is bus compatible with the SDB-80E. A MK 3881 PIO on the card allows writing to the CRT Display at up to 3.300 characters per second.The CRT-80E provides 24 lines of 80 characters. The standard ASCII 96 character font is provided, other fonts may be programmed using MK 2708 PROM's. The command set includes TAB and cursor control.

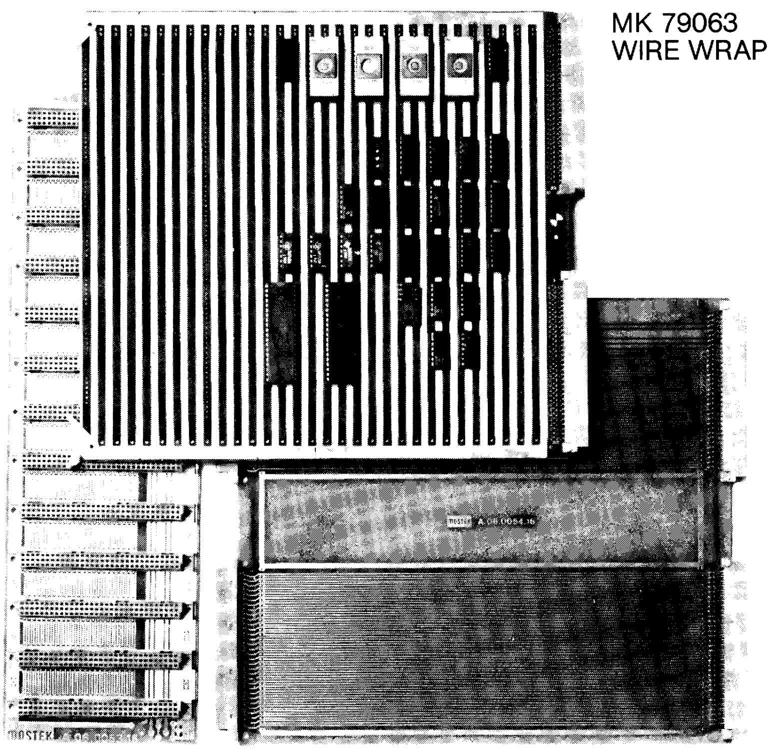
Serial Universal Dislplay Interface UDI-S This is identical to the parallel except it operates over a 4 wire serial current loop connection. Useful in remote terminal applications.

BACK-80E 12 slot prewired printed circuit backplane for the SDB-80E family. This card greatly simplifies system construction.

SDB-80 FUNCTIONAL BLOCK DIAGRAM



MK 79063
WIRE WRAP CARD



EXTENDER CARD
MK 79062

BACKPLANE CARD
MK 79054

Z80 SYSTEM SUPPORT

SYS-80E	SDB-80E with MK 78039	
	4k byte	MK 78040
	16k byte	MK 78041
MOSTEK TERMINAL	Complete Video Display Unit	MK 78037

Z80 PROCESSOR ELEMENTS

OEM-80E	with 4k bytes RAM with 16k bytes RAM	MK 78122 MK 78124
DDT-80	Debug 2k Byte ROM	MK 78118
ASMB-80 EDIT-80	Assembler four 2k Byte ROM's (including the Editor)	MK 78119
SDB-80E	with OEM-80 + DDT-80 + ASMB-80 + EDIT-80 + documentation	
	4k byte RAM	MK 78103
	16k byte RAM	MK 78104

Z80 HARDWARE SUPPORT

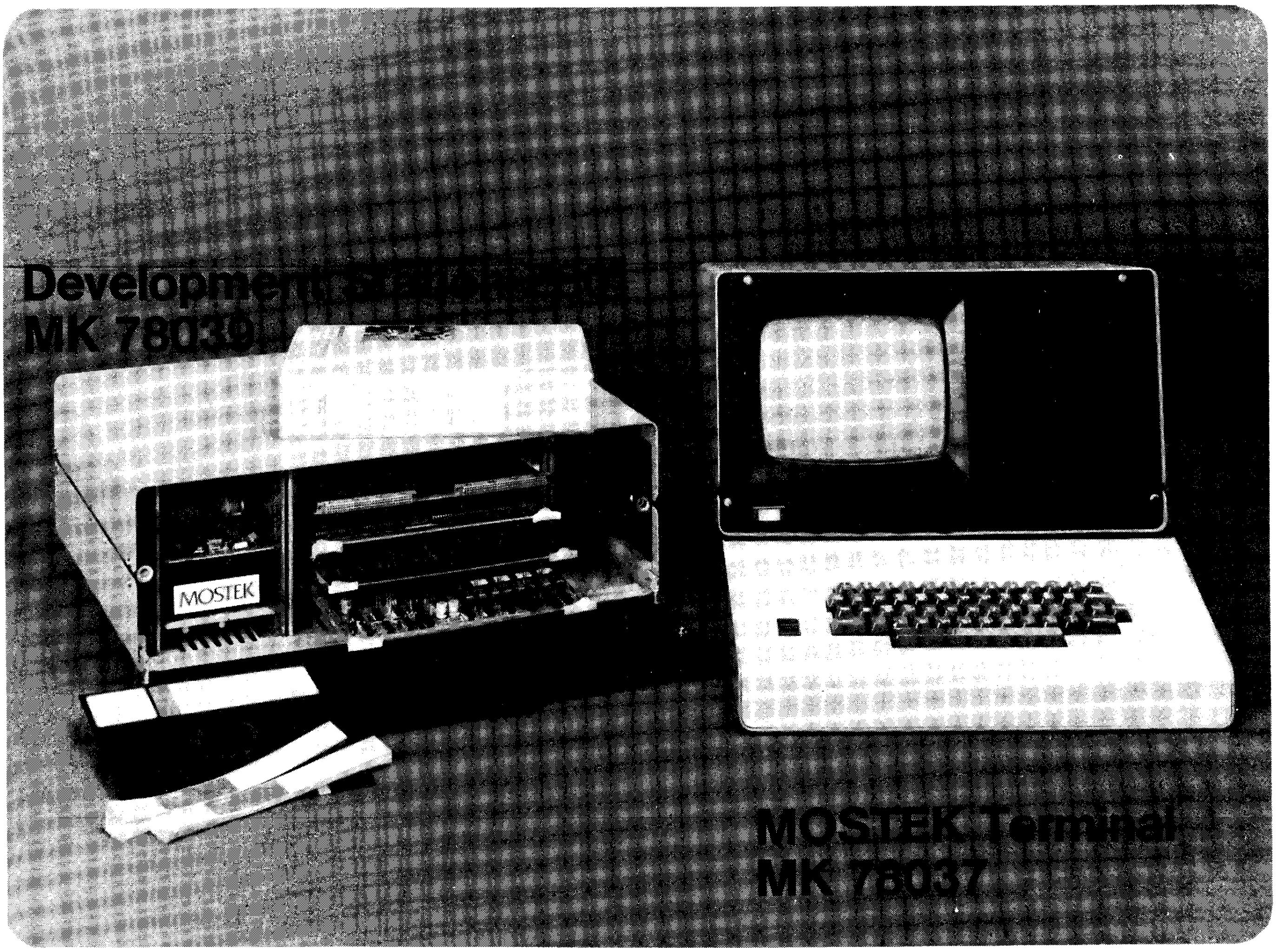
RAM-80AE	16k RAM	MK 78109
RAM-80BE	16k RAM, 2 PIO	MK 78110
XRAM-80	16k expander for RAM-80BE	MK 78126
AIM-80E	I.C.E. (In-Circuit-Emulation)	MK 78106
FLP-80E	Floppy Interface	MK 78112
RIO-80E	16k PROM, 2 PIO, 1-CTC, UART	MK 78128
Universal Display Interface	Serial UDI-S Parallel UDI-P Screen read option	MK 78033 MK 78035 MK 78036
	EIA Interface Cable For SDB-80E	MK 79058
	TTY Cable for SYS-80E	MK 79059
PPG-08	PROM Programmer for MK 2708 1kx8 UV PROM's with enclosure (requires MK 79060)	MK 79033
	PPG-08 Cable for SYS-80E	MK 79060
	Wire Wrap Card	MK 79063
	Extender Card	MK 79062
BACK-80E	Backplane Card, 6 slot	MK 79054
	Development Station Z80 6 total slots, power supply, no cards	MK 78039

Z80 SOFTWARE SUPPORT

XFOR-80	Fortran IV Cross Assembler requires 20k, 16 bit words	
	Card Deck	MK 78117C
	Paper Tape	MK 78117P
XMDS-80	8080 MDS Cross Assembler Paper Tape	MK 78115
XMDS-80D	8080 MDS Cross Assembler Soft sectored diskette	MK 78116
	Listing for DDT-80	MK 78534
	Listing for ASMB-80	MK 78536

DOCUMENTATION

Z80 CPU	Manual	MK 78070
Z80 PIO	Manual	MK 78071
SDB-80E	Manual	MK 78548
RAM-80E	Manual	MK 78545
AIM-80E	Manual	MK 78546
SDB-80E	Literature Package includes CPU, PIO, SDB-80 manuals plus data sheets	MK 78549
PPG-08	Manual	MK 78532
Z80	Pocket reference manual	MK 78516
Z80	Programming manual	MK 78515



MOSTEK TERMINAL MK 78037

FEATURES

- ☐ Self contained visual terminal
- ☐ 24 line, 80 character per line display
- ☐ Baud rate selection 110–9600
- ☐ Current loop or V24
- ☐ Comprehensive commands

GENERAL

A keyboard and a monitor provide together a "teletype" replacement video terminal for MOSTEK development systems, that can also be used in other applications. The terminal is completely self contained with its own power supply and electronics, requiring only the serial communication lines to the computer.

KEYBOARD

The keyboard obtains its power from the display unit, the coded keyboard information and power connections are made over a 25 pin type D connector.

DISPLAY ELECTRONICS

The display electronics uses the MOSTEK universal display interface board (MK 78033), power being provided within the terminal itself. The set of available functions is fully described in the MK 78033 data sheet; the key features are:

- ☐ 24 lines with 80 characters per line
- ☐ Cursor movements, absolute and relative
- ☐ Serial communication, 110–9600 baud
- ☐ Upper and lower case characters
- ☐ Clear screen, clear line etc.
- ☐ Tabulate
- ☐ A 9" diagonal display is used.

MECHANICAL

The display and keyboard are separate units connected by a cable. The display dimensions are:

B 43 cm	H 26 cm	T 32 cm

The keyboard dimensions are:

B 43 cm	H 4,5 cm H 9,0 cm	T 24 cm

DEVELOPMENT STATION Z80 MK 78039

FEATURES

- ☐ Accepts upto 6 total boards
- ☐ Protected power supplies
- ☐ 11 I/O connectors for Peripheral equipment
- ☐ Double Europaformat boards

GENERAL

The MOSTEK development station has been designed to house and provide power for all MOSTEK boards with the double european format as in the detailed description of the SDB-80E. When used in conjunction with the MOSTEK terminal it forms a powerful development system.

POWER SUPPLIES

Plug-in power supplies are used, the supplies being one card with +5 volt 7 ampere and one card with ±12 volt 1 ampere. All supplies have overvoltage protection and current limiting.

MECHANICAL

The housing has the following dimensions:

B 52 cm   H 18 cm   T 36 cm

The front panel has quick release fastners to give free access to the boards.

INPUT/OUTPUT

11 connectors, 25 pin type D, are available for peripheral equipment. For the SDB 80E some of these are already commited as listed here below:

Connector	SDB 80 E function
1	Terminal
2	CTC
3	Floppy disc controller (1)
4	Uncommitted.
5	Uncommitted.
6	Paper tape reader
7	Paper tape punch
8	Line printer (2)
9	Uncommitted.
10	PROM programmer-PPG08
11	Uncommitted.

(1) with FLP 80E
(3) with RAM 80BE (or RIO80E or use PROM prog. Connector)

MOSTEK Z80 DEVELOPMENT SYSTEM

Includes:

MK 78103	SDB-80 Package A or	(4k byte RAM)
MK 78104	SDB-80 Package B with 256 byte static RAM DDT 80 Operating System ASMB 80 Resident Assembler and Text Editor and documentation	(16k byte RAM)
MK 78037	MOSTEK Terminal	
MK 78039	Development Station Z80	

TABLE OF CONTENTS

TABLE OF CONTENTS (CON'T)

TABLE OF CONTENTS (CON'T)

LIST OF FIGURES

LIST OF FIGURES (CON'T)

LIST OF FIGURES (CONT'T)

SDB-80 INTRODUCTION

Mostek's SDB-80 computer board was developed for either software development or OEM board use. This approach is unique in the industry and offers several advantages.

A development system configuration is illustrated in the following figure. Here we have a system that is expandable along the system bus while conveniently connecting to various peripherals through a separate connector. Driver routines are included in the operating system software that allows direct communication to the peripheral units shown. Development system firmware is located in 5 ROM/PROM sockets on the SDB-80 board.

This same board, minus the development system firmware serves as an OEM board. The OEM-80 board is a powerful stand alone computer that has uses in application areas as:

Test Equipment
Remote Data Log
Medical Electronics
Process Controllers

By using one board for both OEM and development system configurations one gains the following advantages:

Z-80 DEVELOPMENT STATION WITH SDB 80E

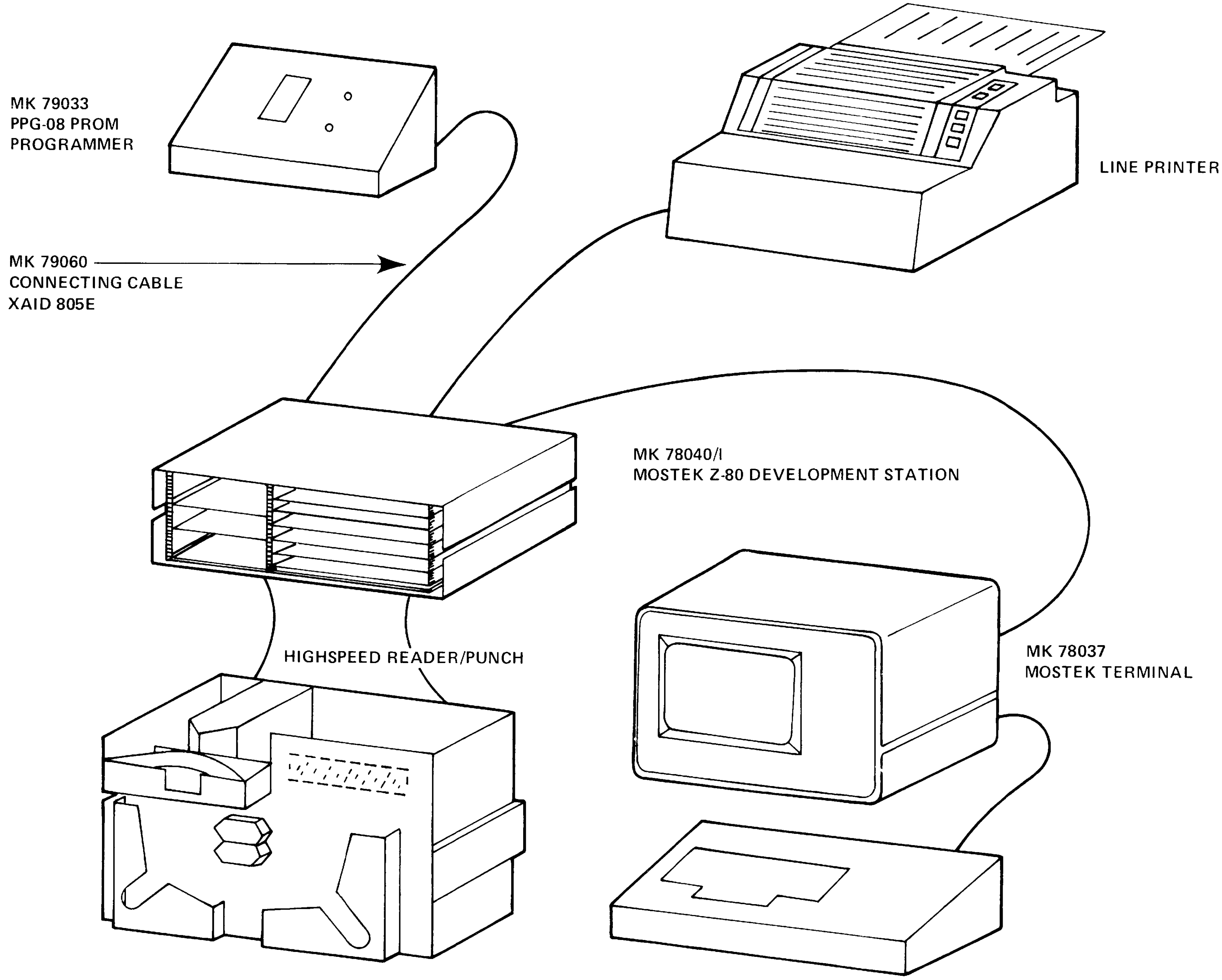


Figure 1

1) Versatility

For the user who builds only a few systems, the SDB-80/ OEM-80 combination can be ideal. One SDB-80 can be used to develop software which will operate on the OEM-80. If necessary, the SDB-80 operating system can re removed and replaced by PROMs containing the user's program.

2) Initial Investment is Limited

With a single card and a terminal, a complete development system can be constructed. With the Resident Assembler, Editor, Debug and Linking Loader, programs may be comfortably debugged at minimal expense.

3) Total Expandability

The SDB-80 can be expanded to more comfortable development systems by the addition of Mostek support products, AIM-80 or FLP-80 for example. The user can configure the exact system he requires, and the same hardware used during development can be used in production.

Functional Description
SDB-80 Board

Refer to the block diagram of Fig. 1 for an overall view of the card.

I. Board and Connectors

The SDB-80 was placed on Mostek's standard size Eurocard based development board (233.4MM x 250MM).

Bus/Control and peripheral lines are routed to separate 64 pin, polarized, fully protected male connectors. This functional separation facilitates system expansion in a card cage environment. Figure 2 is a photograph of the board showing the physical location of the various functional areas.

Two switches and one LED are located at the board top:

S_1: initiates a reset

S_2: determines the reset or restart location

LED: halt instruction indicator

II. Microprocessor Components

The Microprocessor complement is made up of 1 CPU (MK3880), 2 parallel I/O (MK3881), and 1 Control and Timing Circuit (MK3882). Each PIO provides two 10-bit parallel I/O Ports (8 data + 2 handshake) with bi-directional capability. The PIO is programmable under software control so as to make it as versatile as possible. In addition the chip has sophisticated interrupt capability for fast response situations.

SDB-80 FUNCTIONAL BLOCK DIAGRAM

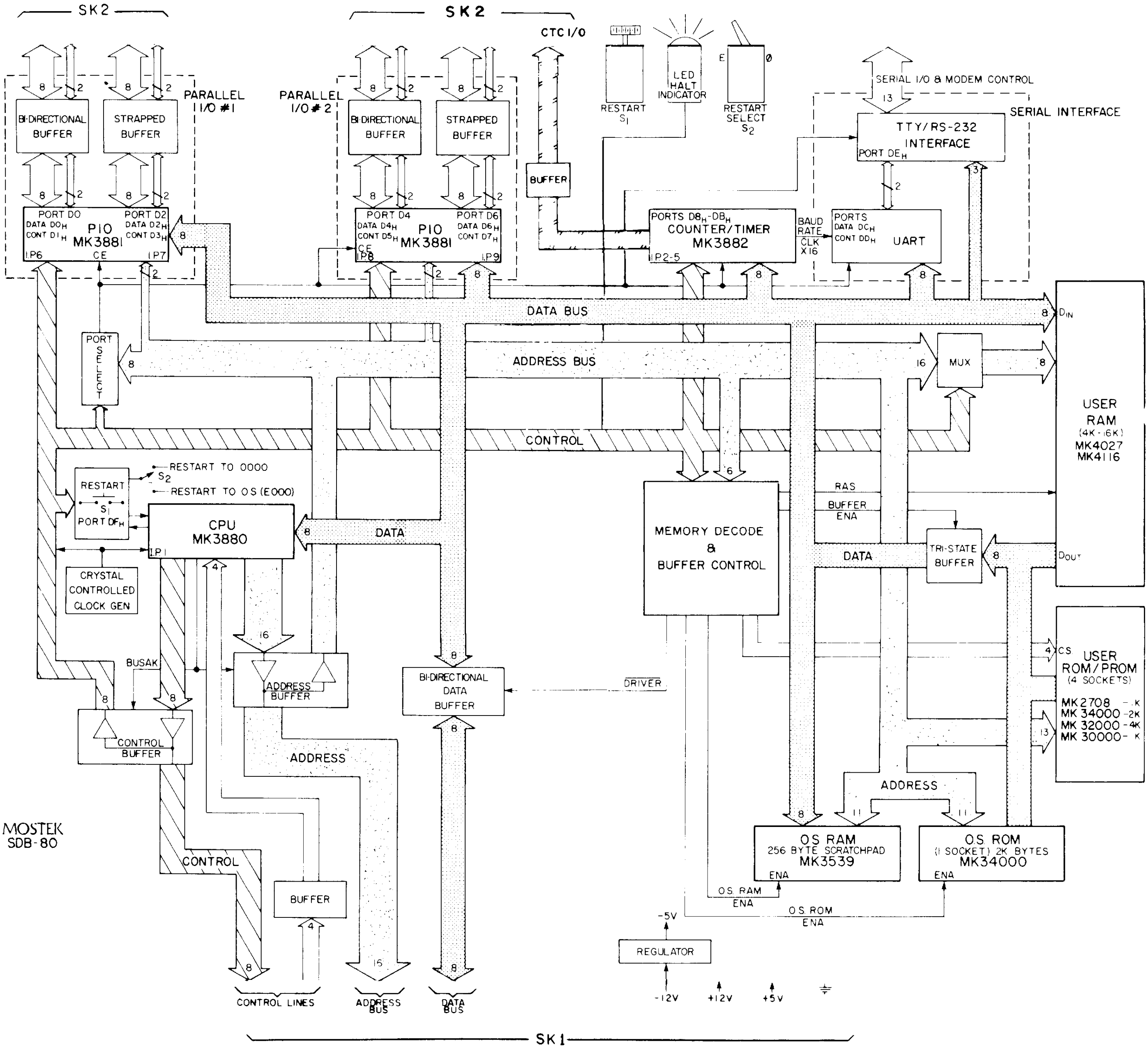
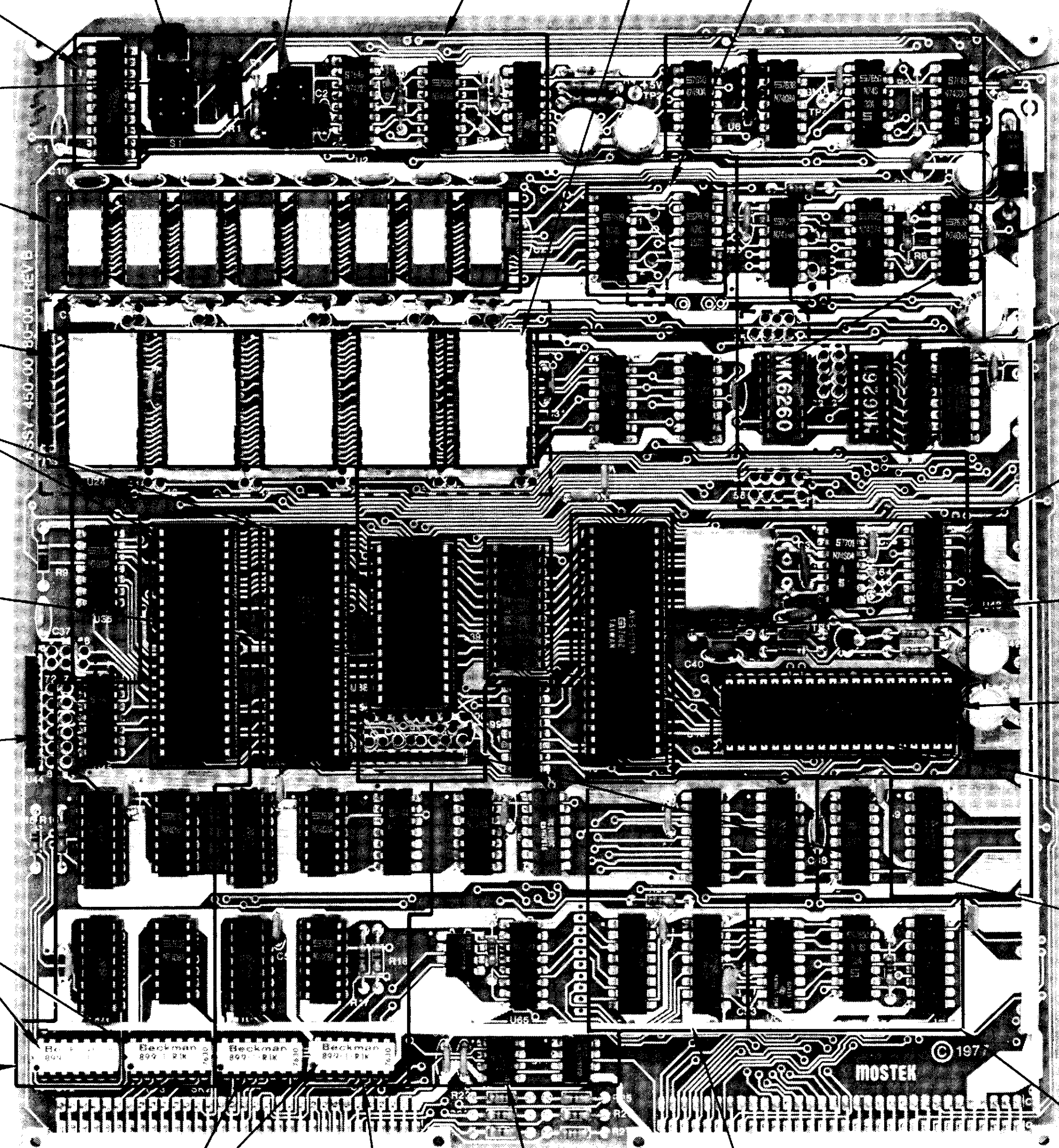


Figure 1



PHYSICAL LOCATION OF MAJOR COMPONENT AREAS

Figure 2

One four channel CTC is included on the SDB-80. Three of the channels are available to the user through either direct input/output or through its interrupt capability. The fourth channel is used as a BAUD Rate Clock Generator for the on-card UART.

The Z80 CPU chip with a 158 unit instruction set, single phase clock input, automatic dynamic memory refresh, and an advanced set of addressing modes is one of the most sophisticated microprocessors available today. It forms the processor for the SDB-80 computer board. This CPU generates the address and control signals, communicates with memory, I/O and peripherals, fetches and executes instructions, and provides most of the timing signals for proper operation of the board.

III. Memory

The SDB-80 has been designed to accommodate a large quantity of memory; specifically 4K or 16K bytes of dynamic RAM, 256 bytes of static RAM, and up to 5K bytes of PROM or 20K bytes of ROM. Numerous options in the memory decoding allow the placement of this memory on any 4K boundary (subdivided into 1K boundaries) within the 65K memory map. The following is a brief description of each portion of the memory section.

User RAM

Eight 16 pin sockets are provided for either 4K dynamic (MK4027) or 16K dynamic parts (MK4116). On board refresh is handled by the memory decode section in conjunction with the refresh signal from the CPU. Upgrading RAM memory size in the field from 4K to 16K bytes is a simple matter of exchanging memory parts and modifying three jumper locations.

By providing software in ROM (as opposed to loading into RAM) and through the use of a separate scratch RAM, all of the 4K or 16K USER RAM is available to the user with no restrictions.

ROM/PROM

Five 24 pin DIP sockets are included on the SDB-80 for development station firmware expansion and/or user program storage space. In a development system configuration one ROM socket is allocated to (and decoded as) the 2K DDT-80 operating system (MK34000 type) while the remaining four are free for accessories such as a Text Editor or Assembler. In an OEM situation all five are available to the user. Each socket can accommodate either a 1K byte PROM or a 1K, 2K, or 4K byte ROM.

a) O. S. ROM

The operating system firmware is contained in 2K of ROM. The starting address for the O. S. is fixed at $E000_H$. This is entered by activating the reset switch (S1) while S2 is in the restart (TO $E000_H$) position.

b) User ROM/PROM

Each socket can be independently decoded anywhere within the 65K memory map. The ROM and/or PROMs can be contiguous with each other and the USER RAM or they may be separated by undecoded memory blocks. Due to the commonality of pin out, memory devices of from 1K bytes to 4K bytes can be inserted in each socket.

c) O. S. RAM

A separate 256 x 8 static RAM is included to handle all scratchpad operations of the operating system. This leaves

4K/16K RAM totally free for the user and his application.

To keep the scratch RAM out of the way in a development system configuration it is decoded at the very top of the memory map; from $FF00_H$ to $FFFF_H$ (location 65,279 to 65,535 in decimal).

In an OEM application this RAM is available to the user and may be decoded elsewhere in the memory map.

d) Memory Decode and Buffer Control

The basic information for decoding memory within the SDB-80 memory map is contained within two bi-polar Read Only Memory parts. The first ROM divides the map into 16 - 4K memory blocks while the second divides a given 4K memory block into 4 - 1K segments. Jumpers are then used to determine which 1K and 4K blocks are actually decoded.

Two control lines are generated in this logic section.

$\overline{\text{DRIVEB}}$:	Indicates Data Bus Drive by the SDB-80 during write or interval read cycles.
$\overline{\text{BUFFER ENA}}$:	Determines whether the memory data buffer is in the active or tri-state (high impedance) condition.

IV. Parallel I/O Interface

Two parallel I/O chips provide four 10 bit (8 data plus 2 handshake) ports. These are available at connector SK2. Each PIO has one bi-directional port and one port that can be hard wired either as an input or as an output. Each line is TTL buffered and has providion for a termination network. Logically each port pair looks like one of the on-board PIO divices.

V. Serial I/O Interface

The serial I/O interface consists basically of a UART, a baud rate clock generator, and the I/O buffering to connector SK2. Asynchronous data rates of from 110 to 9600 baud can be accepted. The baud rate clock is programmable by the CPU, a direct input to the CPU from the serial data allows the CPU to measure the baud rate of a terminal. Direct interface is provided to both teletype (20 mA current loop) and to standard RS-232 terminals (voltage drive).

VI. Counter and Timing Circuit (CTC)

The SDB-80 provides a four channel counter chip that operates under softward control. Interfacing to and from the CTC can be accomplished by input lines (either count or timer mode), by output signals (zero count detect), or through the Z-80 interrupt structure. All four channels are available to the user. However, channel "0" (zero) is generally used as the on board baud rate generator when the serial I/O port is used. Features such as selectable prescaler, count modules, readable down counter, and programmable interrupt are all under software control.

VII. Bus Lines - Data, Address, Control

All Lines going on to and off of the board are TTL buffered and/or terminated. The address and data bus lines are buffered with bidirectional divices so that data can go in both directions. Bus lines are brought out to connector SK1 for ease in system expansion.

VIII. Clock Generator

The SDB-80 comes with a crystal controlled clock generator. This clock drives all the necessary components on the board, is buffered to drive off the board and is gated to accept an external clock. The crystal frequency of 4.916 MHz is divided and supplied to the system as 2.458 MHz. This frequency is a multiple of all the desired baud rates.

IX. Reset/Restart

The SDB-80 can be reset to the bottom of memory at location 0000_H or restarted to the operating system at $E0000_H$. Toggle switch S2, located at the top edge of the board, is used for this function. Push button switch S1 creates the reset.

The top edge LED indicates when the CPU is executing a HALT instruction (OP code 76_H).

X. Port Select

Two dual 2 to 4 line decoders are used for port selection and decode. This logic decodes the address and control lines for PIO, CTC, and the serial port selection. Of the possible 256 ports available, 16 are used on this board (D0 through DF).

INITIAL SDB-80 CHECK OUT PROCEDURE

The SDB-80 board has been fully tested by MOSTEK prior to shipment. The following initial checkout procedure is simply a functional check of the board. If the board fails to perform any of the steps outlined, turn off the power, recheck the board hook up, and try the checkout procedure again. If the problem persists, contact MOSTEK for assistance.

Refer to the Figures on pages 2-3, 15-5 and 15-6 for physical layouts, pin-outs, etc.

I) Minimum Equipment Needed (not supplied by MOSTEK) for SDB-80 Checkout.

1) A serial ASCII terminal i.e., TTY, Silent 700, or CRT equipment.

2) Power Supplies

+ 12V @ 480 mA max
+ 5V @ 2.6 A max
- 12V @ -180 ma max

UNDER NO CIRCUMSTANCES SHOULD 115 VAC EVER BE APPLIED DIRECTLY TO THE SDB-80 BOARD!

II. Board Hookup

Power should be supplied to the board per the following pin out:

POWER	CONNECTOR	PINS
+ 12V	SK1	a6, c6
+ 5V	SK1	a4, c4, a5, c5
GND	SK1	a1, c1, a2, c2
- 12V	SK1	a3, c3

The serial I/O should be connected to the appropriate peripheral as described in Section 6.

III) Initial Check Out Procedure *

The following terminology is used in this section in reference to the terminal:

↲ indicates carriage return
^ indicates carat or up arrow (ASCII Sen).
· indicates period
XXXX, WXYZ indicates hex digits

The underlined portion of the commands is entered by the user. The non-underlined portion is the system response.

1) Set S2 to position E (O.S. reset at 0E000H)

2) Apply power

   2a) Press reset (on SDB-80): Restart to OS at $E000_H$

3) Depress ↲ (carriage return) : Restart to operating system at memory location $E000_H$

4) Terminal now shows a "." as the response. A "." indicates the system is in the command mode and is ready to accept another command. The baud rate has been automatically calculated and adjusted to the proper value.

5) . M to 0, 20 ↲

```
0000 XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
0010 XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
0020 XX
```

: The display or tabulate memory command displays contents of memory from locations 0000_H through 20_H. Each two digit hex number corresponds to one 8 bit byte (binary).

* Applies to Non-Floppy Disk Systems.

6)	.M 0 ↲	
	0000 XX	: Contents of location 0000_H displayed
	0000 XX 76 ^	: Contents of location 0000_H modified to contain OP code 76_H. OP code 76 is the HALT instruction.
	0000 76	: System responds by showing that 76_H is now stored in location 0000_H.
7)	0000 76 .	: Period entered
	.	: System returns to command mode
8)	.E 0 ↲ (No response on terminal LED lights.)	: An EXECUTE command causes the CPU to execute the instruction in memory location 0000_H - a HALT instruction. This is indicated by the LED. Notice there is complete loss of control by the keyboard.
	Depress restart button S1	: LED goes out
	↲	: Carriage return after restart causes the baud rate to be calculated.
	.	: System has been returned to the command mode
9)	.M 0 ↲	: Memory location 0000_H is interogated.
	0000 76	: Memory location 0000_H still has OP code 76_H stored. Reset does not destroy memory contents.
	0000 76 .	: A period returns the system to the command mode.
10)	Set S2 to 0 Depress restart button S1 (no response on terminal; LED lights).	: System has been reset to memory location 0000_H. From 9) above one sees that 76_H is stored at 0000_H. Returning to location 0000_H initiates the HALT instruction which lights the LED.
	Set S2 to E Depress restart button S1 ↲	: System has been returned to command mode.

11) .R 1

```
 PC   AF  I IF  DE   HL  A'F' B'C' D'E' H'L'  IX   IY   SP
XXXX WXYZ XXXX WXYZ XXXX WXYZ XXXX WXYZ XXXX WXYZ XXXX WXYZ XXXX
```

:The REGISTER DISPLAY command prints out the register header designation along with the contents of each CPU register.

12) .M FFE6

:Contents of memory location $FFE6_H$ displayed. This location is very near the top of the memory map which resides in the scratch pad RAM. This is the location where the register information is stored. Each time   is depressed the display indexes to the next memory location. Notice that the memory locations in this part of memory are given mnemonics rather than in hex digits.

```
:SP WXYZ
:IY WXYZ
:IX WXYZ
:L' XX
:H' XX
:E' XX
:D' XX
:C' XX
:B' XX
:F' XX
:A' XX
:L XX
:H XX
:E XX
:D XX
:C XX
:B XX
IF XX
```

:I XX ↵

:F XX ↵

:A XX ↵

:PC WXYZ ↵

: The program counter information is located in memory locations $FFFE_H$ and $FFFF_H$ (two bytes). $FFFF_H$ is the very top of memory. The next ↵ rolls over the boundry of memory map to the start of memory - location 0000_H.

0000 XX

At this point the SDB-80 has been shown to be functional.

To determine the version number of the Operating System firmware, display contents of memory location $E065_H$.

PORTS, PERIPHERALS, AND DRIVERS

The SDB-80, through its I/O ports, can communicate with many types of peripheral devices. Drivers have been included in the O.S. to make this board as easy to use and interface with as possible. The following data lists all the peripheral devices supported by a software driver in the O. S. along with specific interface port information.

	Peripheral	Port	Connector
1)	Teletype	Serial	SK2
2)	Teletype tape reader/punch	Serial	
3)	RS-232 peripheral	Serial	
4)	Silent 700 (keyboard & printer)	Serial	
5)	High speed paper tape reader	D0, Parallel	
6)	High speed paper tape punch	D2, Parallel	
7)	Line printer	D6, Parallel	
8)	PROM programmer (Software available on paper tape)	D4 and D6, Parallel	SK2

Figure 1 shows a map of all possible 256 ports available for the SDB-80. The bottom 48 ports are committed now or reserved for the over-all Z-80 development system.

SDB-80 PORT MAP

MSD ↓ / LSD →	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0																
1																
2																
3																
4																
5																
6																
7																
8																
9																
A																
B																
C																
D	PIO D0 Data	PIO D0 Control	PIO D2 Data	PIO D2 Control	PIO D4 Data	PIO D4 Control	PIO D6 Data	PIO D6 Control	CTC Channel 0	CTC Channel 1	CTC Channel 2	CTC Channel 3	UART Data	UART Control	System Control	System Control
E	RESERVED						FOR FUTURE									
F													EXPANSION			

Figure 1

JUMPER OPTIONS ON SDB-80
AS SHIPPED FROM THE FACTORY

Figure 1 shows the locations and grouping of wire wrap posts that allow the SDB-80 user the flexibility of installing jumpers to achieve the format options available.

Figure 2 illustrates how that option is connected when the board is shipped from the factory.

Additional wiring options will be detailed in later sections of this manual.

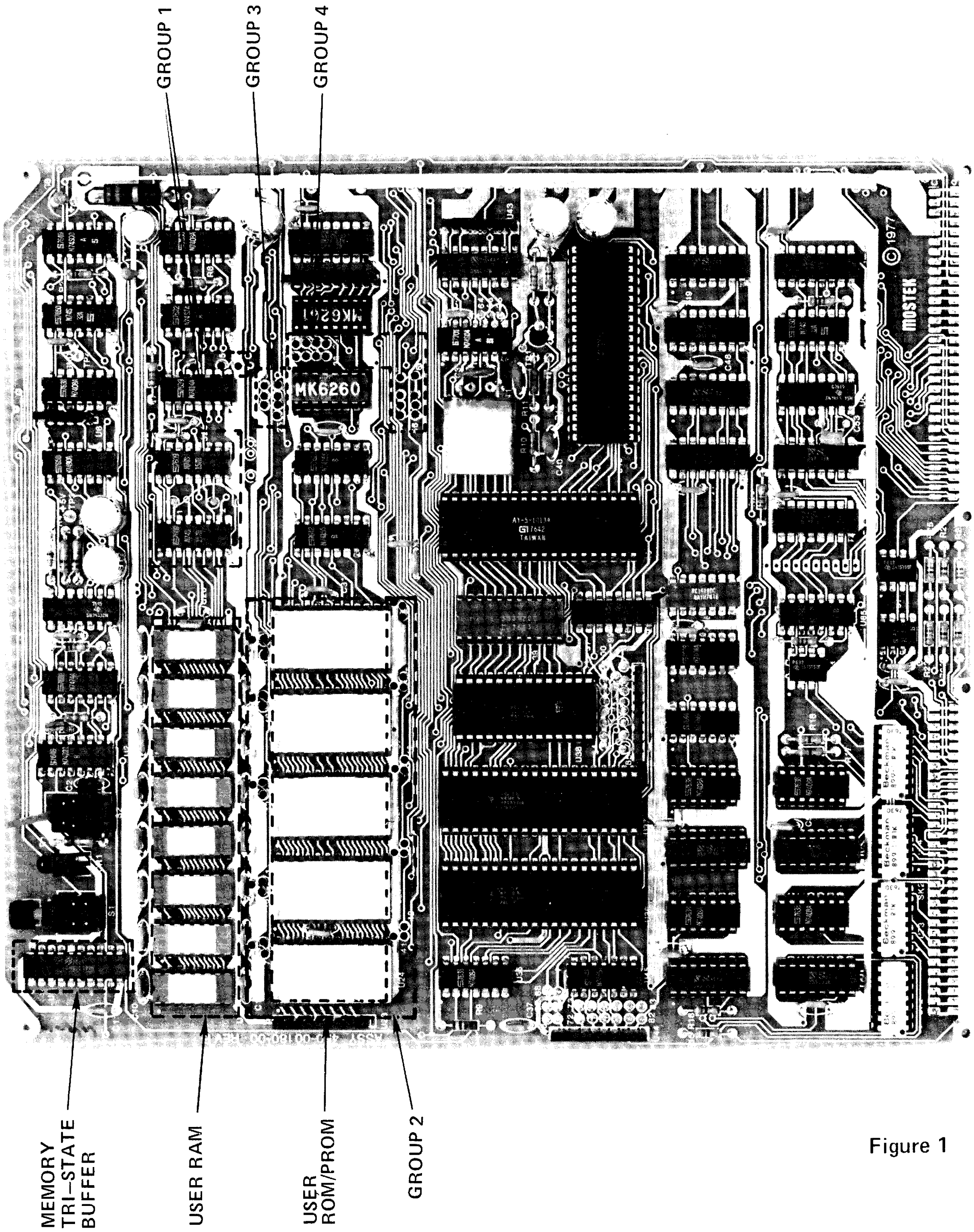

GROUP 1
GROUP 3
GROUP 4
MK6260
MOSTEK
MEMORY
TRI–STATE
BUFFER
USER RAM
USER
ROM/PROM
GROUP 2

Figure 1

SDB-80 PRODUCTION VERSION AS SHIPPED FROM FACTORY

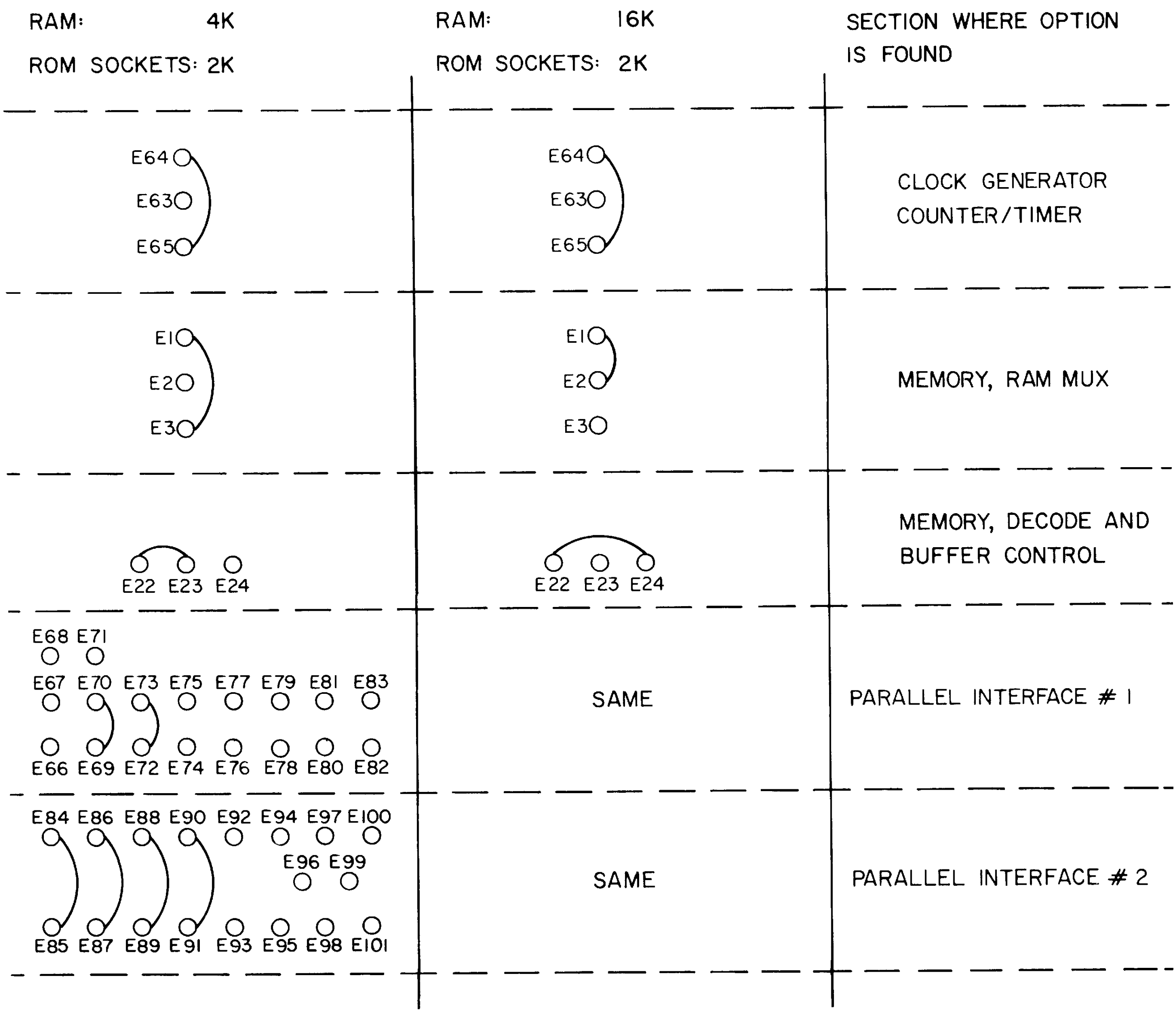


Figure 2

SDB-80 PRODUCTION VERSION AS SHIPPED FROM FACTORY (Cont'd)

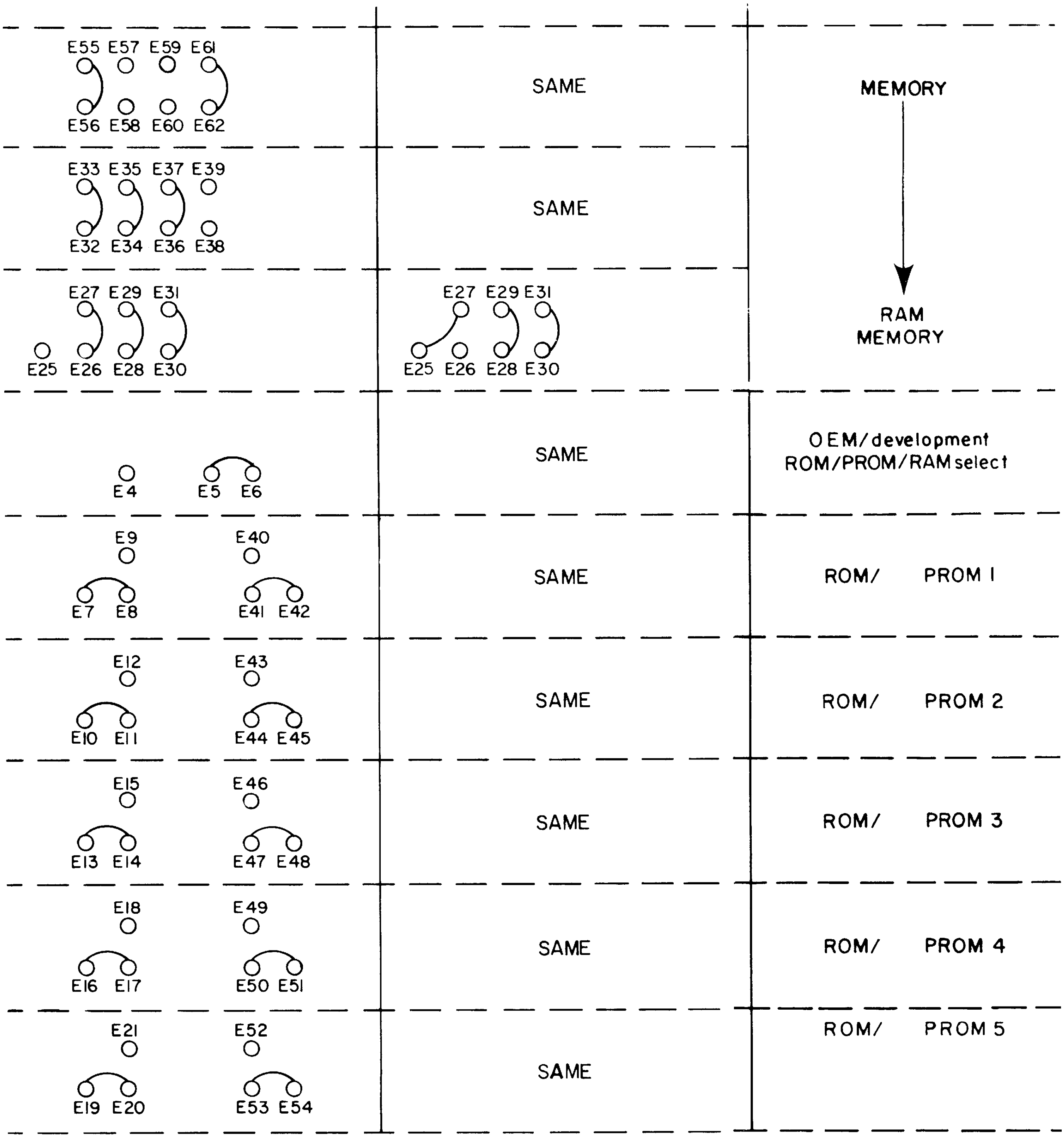


Figure 2

SDB TO PERIPHERALS

Interface Information

The SDB-80 will interface directly with a number of different peripheral units. This section describes the interconnect cableing and the jumper options (on the SDB-80) necessary for the interface to a number of typical peripherals.

The peripherals covered in this section are:

TERMINALS - - PORT: SERIAL

1) Teletype model ASR-33 without reader step control
2) Teletype model ASR-33 with reader step control
3) Texas Instruments Silent 700 model 733 ASR (or any RS232 terminal)

PAPER TAPE READER/PUNCH - - PORT: PARALLEL

1) Plessy model PM-750 Paper Tape Reader	Port D0
2) Plessy model PM-1000 Paper Tape Reader	Port D0
3) Plessy model PM C4020 Paper Tape Reader/Punch	Ports D0, D2
4) EECO model RPF9360B0AAA Paper Tape Reader/Punch	Ports D0, D2

PRINTERS - - PORT: PARALLEL

1) Data Products model 2310	Port D6
2) Data Printer model CT 1334	Port D6

PROM PROGRAMMER - - PORT: PARALLEL

1) Mostek PPG-08	Ports D4, D6

I. SDB-80 INTERFACE TO SERIAL TERMINALS

For applications not requiring a reader step control, an ASR-33 teletype may be connected (TTY plug #2) directly to the SDB-80. No jumper options or modifications are required on the SDB-80. The interconnect schematic is shown in Figure 1.

The resident assembler requires a controlled reader, for this and other applications requiring reader step control, TTY plugs # 2 and # 3 are used in conjunction with a solid state relay. No jumper options or modifications are required on the SDB-80. The interconnect schematic is shown in Figure 2. Other connections are possible but require a more thorough knowledge of the ASR-33.

A Texas Instruments model 733 ASR (silent 700) terminal will interface directly to the SDB-80. No jumper options or modifications are required on the SDB-80.

TTY CONNECTIONS

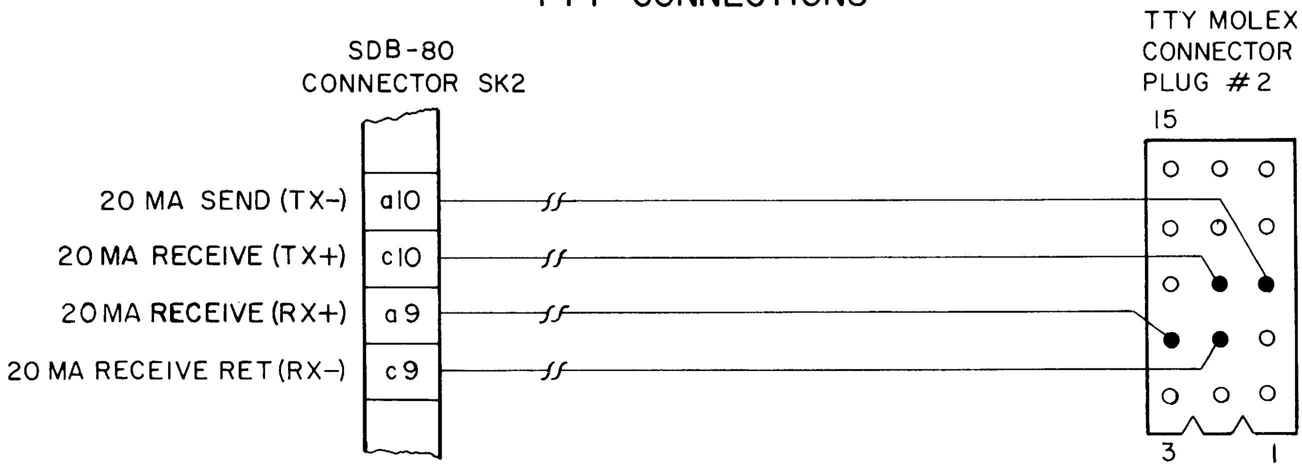


Figure 1

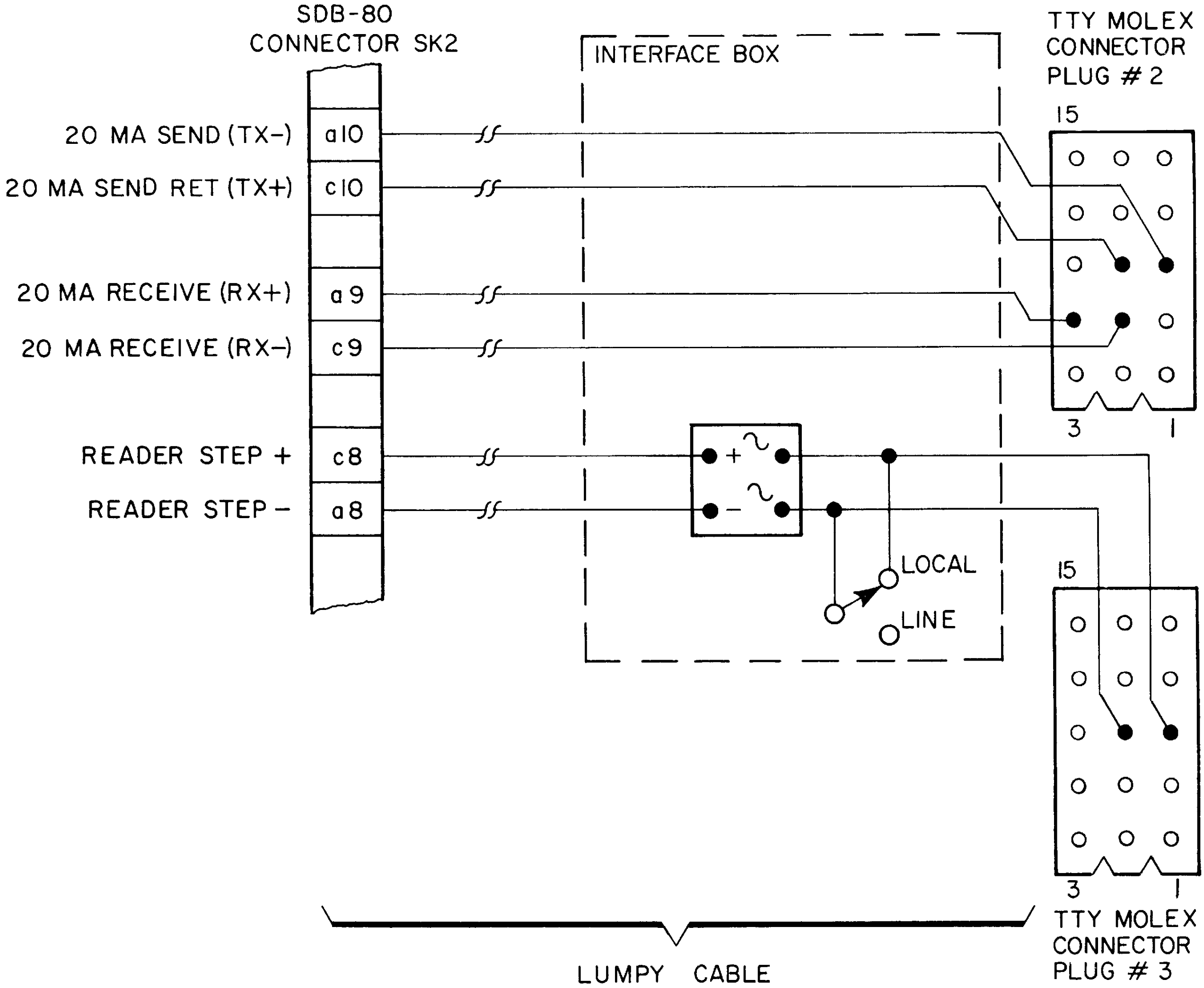


Figure 2

The cable wiring list is given below:

CABLE WIRING LIST

SDB–80 CONECTIOR SK2 PIN	RS–232 SIGNAL	SILENT 700 CONNECTOR PIN
		1
		14
a9	TRANSMITTED DATA	2
		15
c3	RECEIVED DATA	3
c10	20mA + SEND	16
a7	REQUEST TO SEND	4
a10	20mA – SEND	17
c7	CLEAR TO SEND	5
a8	RDR STEP –	18
c6	DATA SET READY	6
c8	RDR STEP +	19
a1	GND	7
a6	DATA TERMINAL READY	20
c10	CARRIER DETECT	8
		21
		9
		22
c9	20mA – REC	10
		23
		11
a9	20mA + RECEIVE	24
		12
		25
		13

Figure 3

II. SDB-80 Interface to Paper Tape Reader/Punches

Plessy models PM 750 and PM 1000 paper tape readers will interface to the SDB-80 as shown in wire list below.

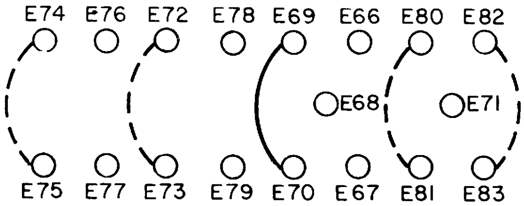
SDB–80 SIGNAL	CONNECTOR SK2	(25 PIN "D" CONN. FEMALE)	
		PLESSY 750	PLESSY 1000
$\overline{\text{SDB (DO)}}$	c28	24	25
RDY (DO)	a28	22	17
DO (0)	c32	1	1
DO (1)	c31	2	2
DO (2)	c30	4	4
DO (3)	c29	5	5
DO (4)	a29	7	7
DO (5)	a30	8	8
DO (6)	a31	10	10
DO (7)	a32	11	11

The jumper options on the SDB-80 should be configured as shown in Figure 4. Jumpers shown with SOLID lines must be installed (Port DO). Jumpers shown with DASHED LINES (Port D2) do not affect the operation of the reader. Jumpers NOT SHOWN must not be installed.

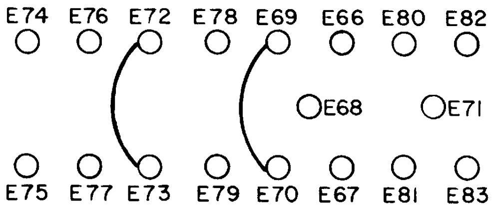
The reader should be configured such that READER DATA and the READER RUN (RUN) and SPROCKET (SPK) signals are active low.

INTERFACE JUMPERS

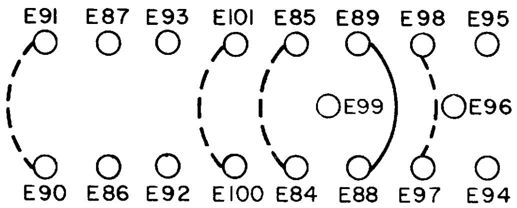
SECTION WHERE THE JUMPERS ARE LOCATED



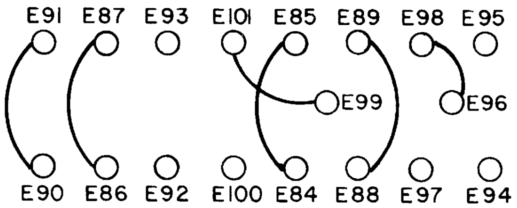
PARALLEL I/O # 1 — FIGURE 4



PARALLEL I/O # 1 — FIGURE 5



PARALLEL I/O # 2 — FIGURE 6



PARALLEL I/O # 2 — FIGURE 7

A Plessy model PM-C4020 or EECO model RPF9360BOAA Reader/Punch will interface to the SDB-80 as shown in wire list.

SDB–80 SIGNAL	CONNECTOR SK2	PLESSY PM-C4020	EECO PPF9360
$\overline{\text{STB (DO)}}$	c28	J1–9	J1–9
RDY (DO)	a28	J1–16	J1–5
DO (0)	c32	J1–1	J1–11
DO (1)	c31	J1–2	J1–12
DO (2)	c30	J1–3	J1–13
DO (3)	c29	J1–4	J1–14
DO (4)	a29	J1–5	J1–15
DO (5)	a30	J1–6	J1–16
DO (6)	a31	J1–7	J1–17
DO (7)	a32	J1–8	J1–18
GND	a1	J1–11, 12, 13, 18, 24	J1–22, 23, 24, 25
RDY (D2)	a23	J2–11	J2–12
$\overline{\text{STB (D2)}}$	a23	J2–12	J2–13
D2 (0)	c27	J2–1	J2–3
D2 (1)	c26	J2–2	J2–4
D2(2)	c25	J2–3	J2–14
D2 (3)	c24	J2–4	J2–15
D2 (4)	a24	J2–5	J2–2
D2 (5)	a25	J2–6	J2–1
D2 (6)	a26	J2–7	J2–5
D2 (7)	a27	J2–8	J2–6
GND	c1	J2–14, 15, 18, 23, 25	J2–20, 25

The jumper options on the SDB-80 should be configured as shown in Figure 5.

The reader should be configured so the READER DATA, the READER RUN SIGNAL ($\overline{\text{RUN}}$) and the SPROCKET SIGNAL ($\overline{\text{SPKT}}$) are active low.

The punch should be configured so the PUNCH DATA and the PUNCH READY (PRDY) are ACTIVE HIGH. The PUNCH COMMAND ($\overline{\text{PCMD}}$) should be ACTIVE LOW.

The timing for the reader and punch data transfer is shown in Figure 8.

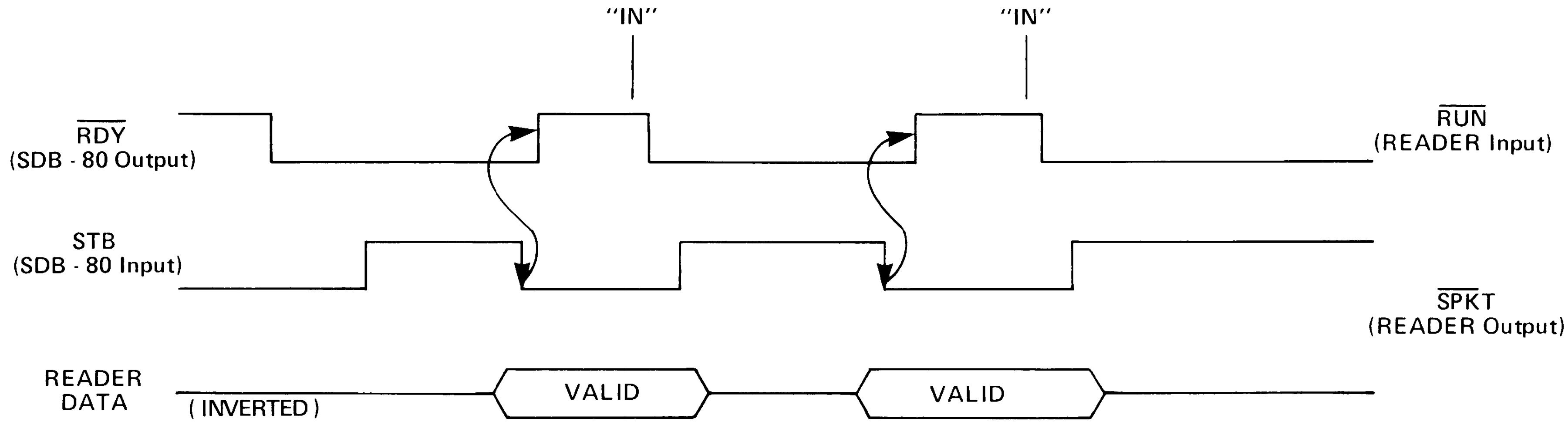
Reader Timing

An "IN" instruction causes the $\overline{\text{RUN}}$ signal to go low requesting a character. The $\overline{\text{SPKT}}$ signal goes high indicating that data is not valid. After data becomes valid $\overline{\text{SPKT}}$ goes low causing the $\overline{\text{RDY}}$ line to go high. The negative edge of $\overline{\text{SPKT}}$ generates an interrupt to the processor. The processor then reads or inputs the next word.

Punch (or Printer) Timing

An "OUT" instruction outputs data to the paralled I/O port causing the $\overline{\text{PCMD}}$ line to go low. The PUNCH READY (PRDY) goes low indicating the punch is busy and has not accepted data. When the punch has accepted data from the port and is ready for another character, the PRDY signal goes high causing $\overline{\text{PCMD}}$ to go high. This action generates an interrupt to the processor. The processor then outputs the next word.

PAPER TAPE READER TIMING



PAPER TAPE PUNCH AND LINE PRINTER TIMING

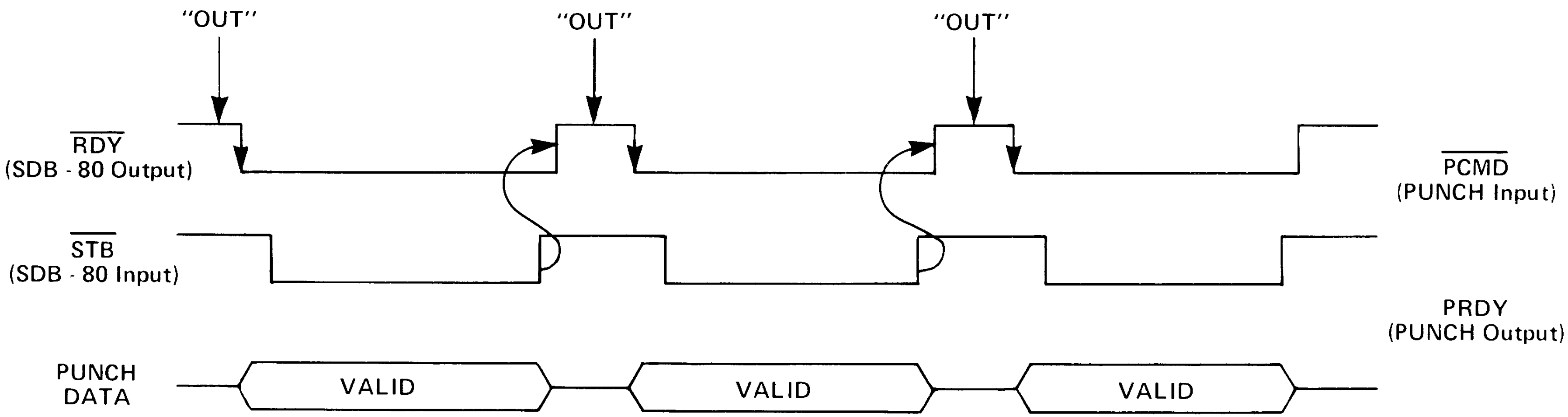


Figure 8

III. SDB-80 Interface to Line Printers

In this example Data Products line printer model 2310 is interfaced to the SDB -80. The jumper options on the SDB-80 should be configured as shown in Figure 6. Jumpers shown with SOLID lines must be installed (Port D6). Jumpers shown with DASHED lines (Port D4) do not affect the operation of the printer. Jumpers NOT SHOWN must not be installed.

The line printer should be configured so that the DATA and ACKNOWLEDGE lines are active high and the character strobe is negative edge triggered.

The following wire list defines the interface cable for the Data Products line printer.

SDB–80 SIGNAL	CONNECTOR SK2	DATA PRODUCTS WINCHESTER CONNECTOR
D6 (0)	c15	B
D6 (1)	c14	F
D6 (2)	c13	L
D6 (3)	c12	R
D6 (4)	a12	V
D6 (5)	a13	Z
D6 (6)	a14	(N)
RDY (D6)	a11	(J)
STB (D6)	c11	E
GND	a1	D,J,N,T,X,(B),(K),(M), C

Timing for the line printer is the same as the punch and is shown in Figure 8.

IV. SDB-80 Interface to PROM Programer

A Mostek PROM Programmer (PPG-08, MK 79033) for MK 2708 PROMs will interface to the SDB-80 with the PROM Programmer interface cable XAID-805 (MK79041).

The jumper options on the SDB-80 should be configured as shown in Figure 7.

No jumper options are required on the PROM Programmer.

The wire list defines the Prom Programmer interface cable.

SDB-80 to PPG-08

SDB–80 SIGNAL	CONNECTOR SK2	PPG–08 J1
$\overline{\text{STB (D4)}}$	c16	2
RDY (D4)	a16	4
D4 (0)	c20	6
D4 (1)	c19	8
D4 (2)	c18	10
D4 (3)	c17	12
D4 (4)	a17	14
D4 (5)	a18	16
D4 (6)	a19	18
D4 (7)	a20	20
RDY (D6)	a11	22
$\overline{\text{STB (D6)}}$	c11	24
D6 (0)	c15	26
D6 (1)	c14	28
D6 (2)	c13	30
D6 (3)	c12	32
D6 (4)	a12	34
D6 (5)	a13	36
D6 (6)	a14	38
D6 (7)	a15	40
GND	a1, a2	All Odd

SERIAL INTERFACE

The Serial I/O Interface consists basically of a UART (AY-5-1013), a baud rate clock generator, and I/O buffering to connector SK2. Asynchronous data rates of from 110 to 9600 baud can be handled by the SDB-80. Logic is provided to allow the CPU to monitor the serial data input in order to permit automatic Baud rate calculation by software. Direct interface is provided to both teletype 20mA current loop and to standard RS-232 terminals (voltage drive). Four handshake lines are available for serial communication links. Figure 1 shows a block diagram of the serial port.

I. Baud Rate Clock

Channel 0 of the Counter/Timer circuit (U38) has been allocated as the baud rate clock generator. The CTC is driven by the master clock whose frequency (2.568 MHz) has been selected as a multiple of the baud rate frequencies. The following table lists the baud rates, baud rate clocks (16X the baud rate), CTC modulus, and percent error from the desired baud rate.

BAUD RATE	SYSTEM CLOCK	CTC MODULUS (÷by)	ACTUAL BAUD RATE CLOCK (16X BAUD RATE)	DESIRED BAUD RATE CLOCK	% ERROR FROM
110	( )	1392	1766	1760	+ 0.34%
300	( )	512	4801	4800	+ 0.2 %
600	( )	256	9602	9600	+ 0.2 %
1200	(2.458)	128	19203	19200	+ 0.2 %
2400	(MHz)	64	38406	38400	+ 0.2 %
4800	( )	32	76813	76800	+ 0.2 %
9600	( )	16	153625	153600	+ 0.2 %

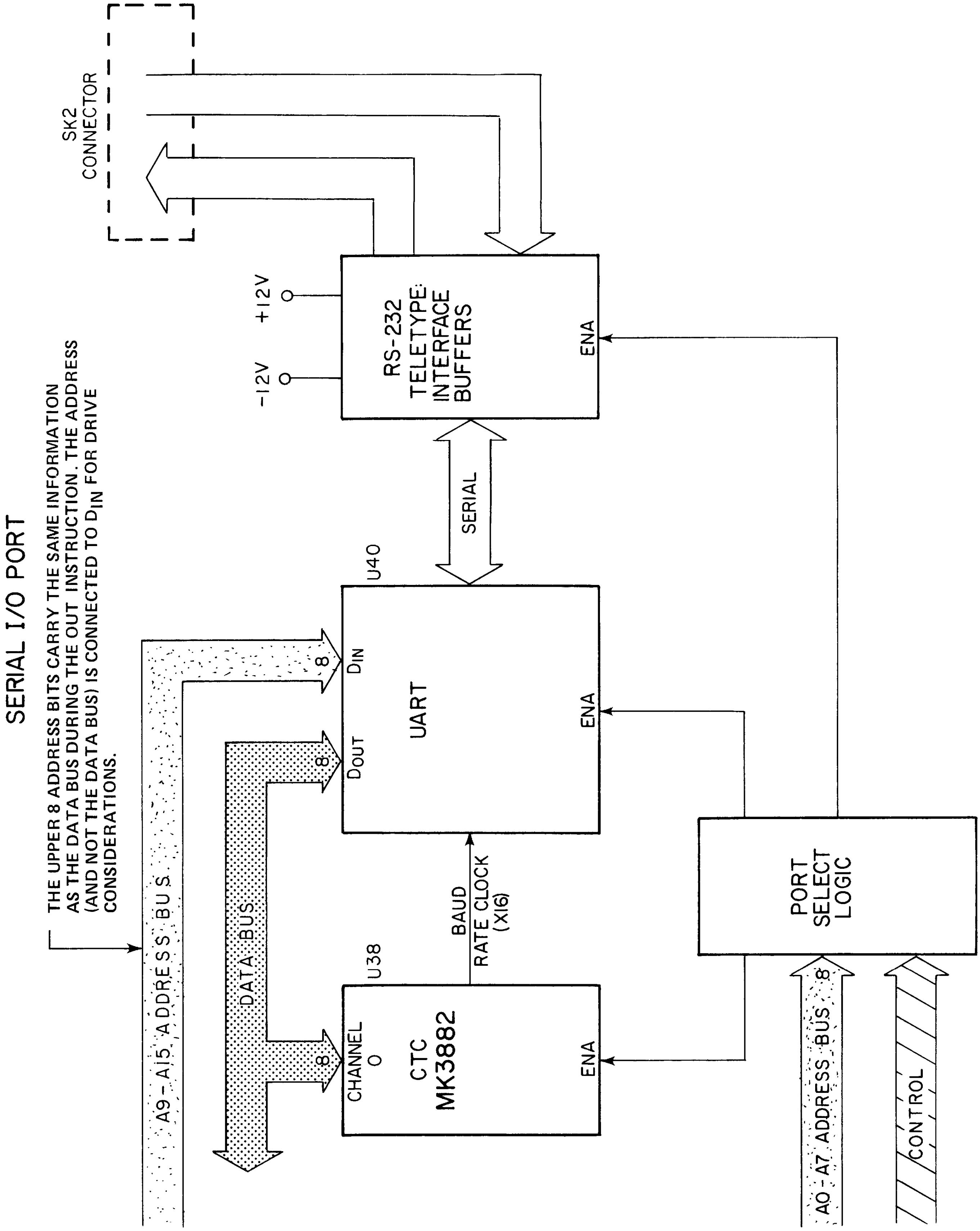

SERIAL I/O PORT
THE UPPER 8 ADDRESS BITS CARRY THE SAME INFORMATION AS THE DATA BUS DURING THE OUT INSTRUCTION. THE ADDRESS (AND NOT THE DATA BUS) IS CONNECTED TO D_{IN} FOR DRIVE CONSIDERATIONS.
SK2 CONNECTOR
-12V
+12V
RS-232 TELETYPE INTERFACE BUFFERS
ENA
SERIAL
U40
UART
D_{IN}
D_{OUT}
ENA
8
8
A9 - A15 ADDRESS BUS
DATA BUS
U38
BAUD RATE CLOCK (X16)
CTC MK3882
CHANNEL 0
8
ENA
PORT SELECT LOGIC
A0 - A7 ADDRESS BUS
8
CONTROL

Figure 1

The CTC modulus is under software control. Contained within the DDT-80 O.S. is a routine which calculates the baud rate of any peripheral interfacing to the serial port and adjusts the count modulus correspondingly.

Both transmit and receive clocks on the UART are tied together so operation is restricted to one common frequency.

II. UART

A full duplex UART is used to receive and transmit data at the serial port. Operation and UART options are under software control. Once the unit has been programmed no further changes are necessary unless there is a modification of the serial data format. Features of the UART include:

- Full duplex operation
- Start bit verification
- Data word size variable from 5 to 8 bits
- One or two stop bits may be selected
- Odd, even, or no parity option
- One word buffering on both transmit and on receive

Transmit and receive clock rates (baud clock rate) must be 16 times the desired baud rate. This clock has been brought to connector SK1. Through a jumper option (E63, 64, 65) an external clock may be supplied to the SDB-80. (See Counter/Timer Section).

A word needs to be said about the UART data interface to the CPU (Din, Dout). Refer to Figure 1 and to the main schematic. During I/O operations, address information is carried directly by the lower 8-bit address lines. This is shown by A0 - A7 going directly to the Port Select Logic block. The upper 8 bits are left free to carry information from the accumulator; the same information that is on the DATA BUS (which is connected to Dout). For loading and drive considerations the ADDRESS BUS (and not the DATA BUS) has been connected to the DATA IN port of the UART.

In addition to current loop terminals, the SDB-80 was designed to communicate with RS-232 terminals and therefore the serial interface looks like a receiving modem or computer port rather than a transmitting terminal port (such as a Silent 700). The effect of this is to exchange three pairs of signals.

SERIAL PORT TO RS-232 CONNECTOR

SK2 CONNECTOR PIN	SIGNAL	RS-232 CONNECTOR PIN
2a 1	Chassis GND	1
2a 9	Transmitted data (RS-232) from terminal	2
2c 3	Receive data (RS-232) at terminal	3
2a 7	Request to send	4
2c 7	Clear to send	5
2c 6	Data set ready	6
2a 1	GND	7
2c10	Carrier detect	8
		9
2c 9	20 mA receive	10
		11
		12
		13
		14
		15
2c10	20 mA send +	16
2a10	20 mA send –	17
2a 8	Reader step –	18
2c 8	Reader step +	19
2a 6	Data terminal ready	20
		21
1a12	RESET	22
		23
2a 9	20 mA receive +	24
		25

Figure 2

For Example:

1) Transmitted Data (RS-232) from Terminal (a9) is an output signal at the terminal but it is shown as an input signal at the serial port.

Receive Data (RS-232) at Terminal (c3) is an input signal at the terminal but is shown as an output signal at the serial port.

2) Request To Send (a7) is an output signal at the terminal but is shown as an input signal at the serial port.

Clear To Send (c7) is an input signal at the terminal but is shown as an output signal at the serial port.

3) Data Terminal Ready (a6) is an output signal at the terminal but is shown as an output signal at the serial port.

Data Set Ready (c6) is an input signal at the terminal but it is shown as an output signal at the serial port.

To change the "sense" of this port i.e., to make it look like a transmitting terminal (as might be required in some OEM applications) the two signals in each pair above needs to be interchanged. DDT-80 will not necessary support such a change. However, such a change is normally made together with custom software.

IV. I/O Buffering

The serial port has been designed to interface directly to both RS-232 and 20mA current loop terminals with no jumper changes. Also brought out to the connector SK2 are four handshake control lines for modem use or general control functions. Each line making up the I/O port is catagorized below:

READER STEP (RS^+,RS^-):

Two lines are allocated for the "Reader Step" or "Tape Reader Control" drive on a teletype machine. This is the mechanism that advances the punch tape. The interface schematic is shown in Figure 3. These lines are capable of supplying + 12V at up to 120mA for driving interface adapters which in turn control the 115VAC reader step directly. The 12V signal can drive an isolating relay or an optically isolated coupler/solid state AC switch. These adapters may be incorporated directly into a "lumpy" cable, although in many cases the isolator (which isolates 115VAC from the board) is located in the teletype machine. In this case, direct connection of RS^+ and RS^- to the teletype is the convenient way to interface. UNDER NO CIRCUMSTANCES SHOULD 115 VOLTS EVER BE APPLIED TO THE SDB-80 DIRECTLY!

TRANSMIT SERIAL DATA (TX, TX^+, TX^-):

The outgoing serial data communication link uses these three lines. The interface schematic of both the voltage and current drive situation is shown in Figure 4.

1) Voltage Drive- RS-232 (TX)

A 75150 RS-232 buffer drives this line directly. Voltage excursions are above and below ground and meet the RS-232 specification.

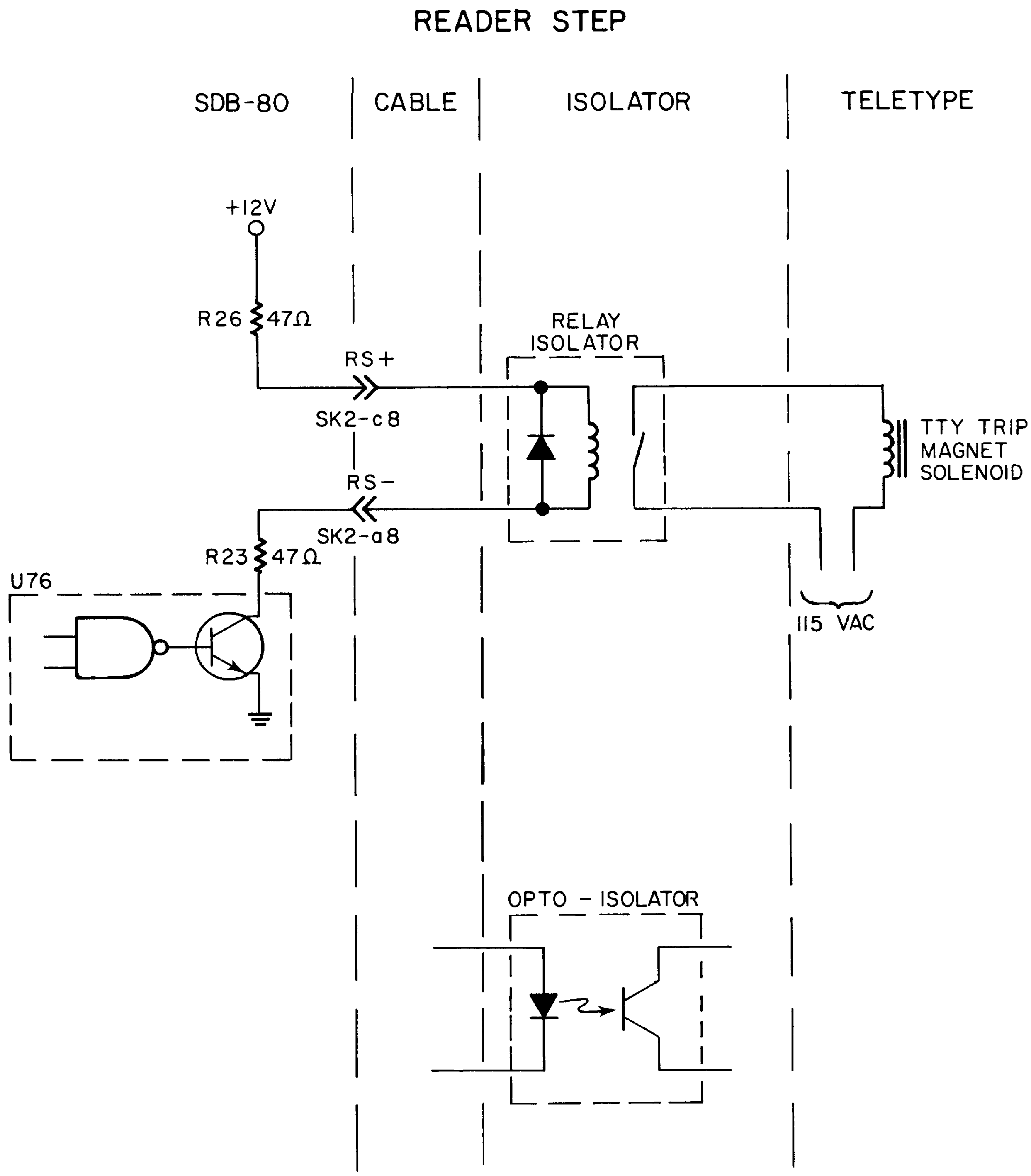

READER STEP
SDB-80
CABLE
ISOLATOR
TELETYPE
+12V
R26
47Ω
RS+
SK2-c8
RS−
SK2-a8
R23
47Ω
U76
RELAY
ISOLATOR
TTY TRIP
MAGNET
SOLENOID
115 VAC
OPTO - ISOLATOR

Figure 3

SERIAL DRIVE TO THE PERIPHERAL FROM THE SDB – 80

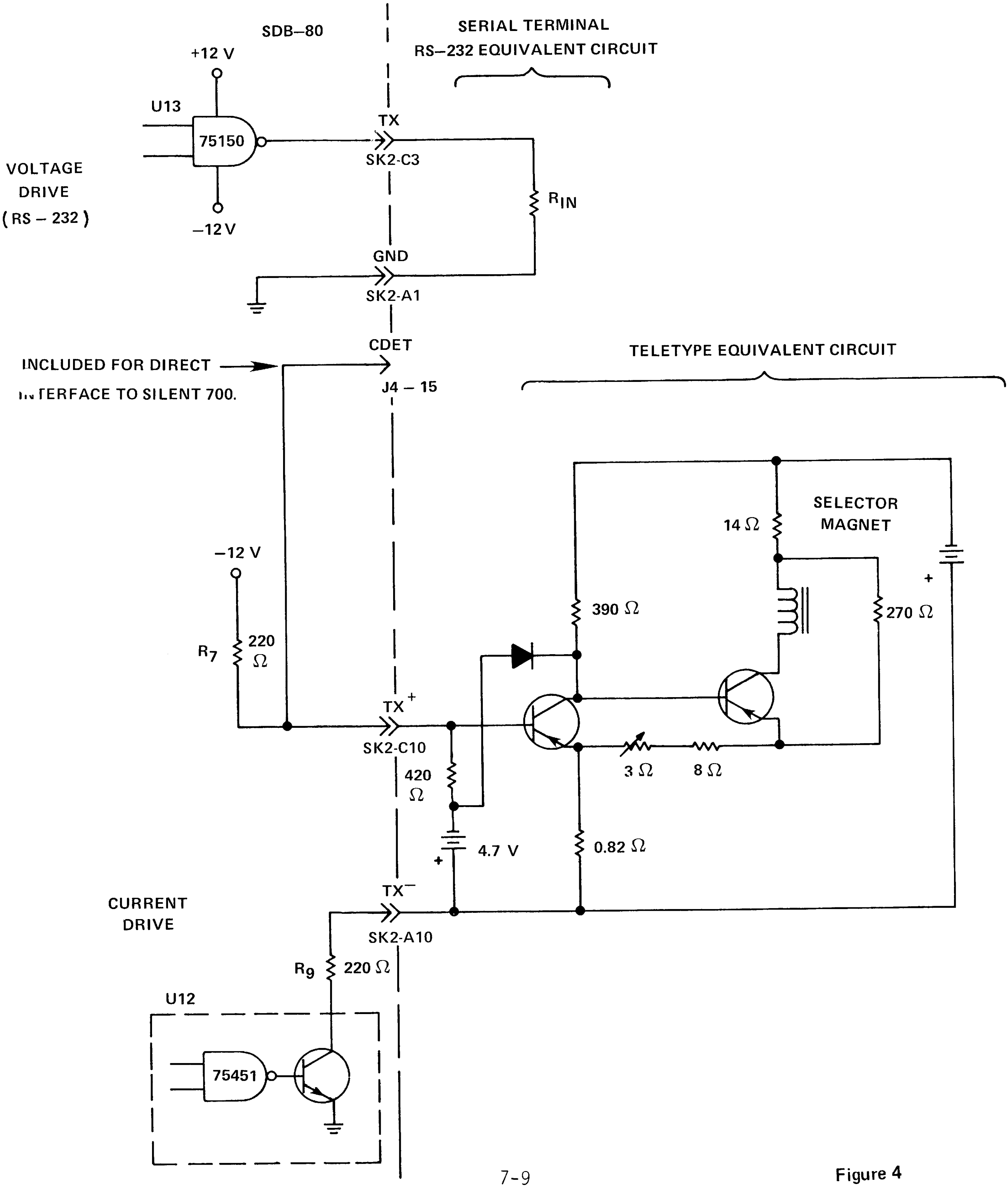


Figure 4

SERIAL DRIVE TO THE SDB-80 FROM THE PERIPHERAL

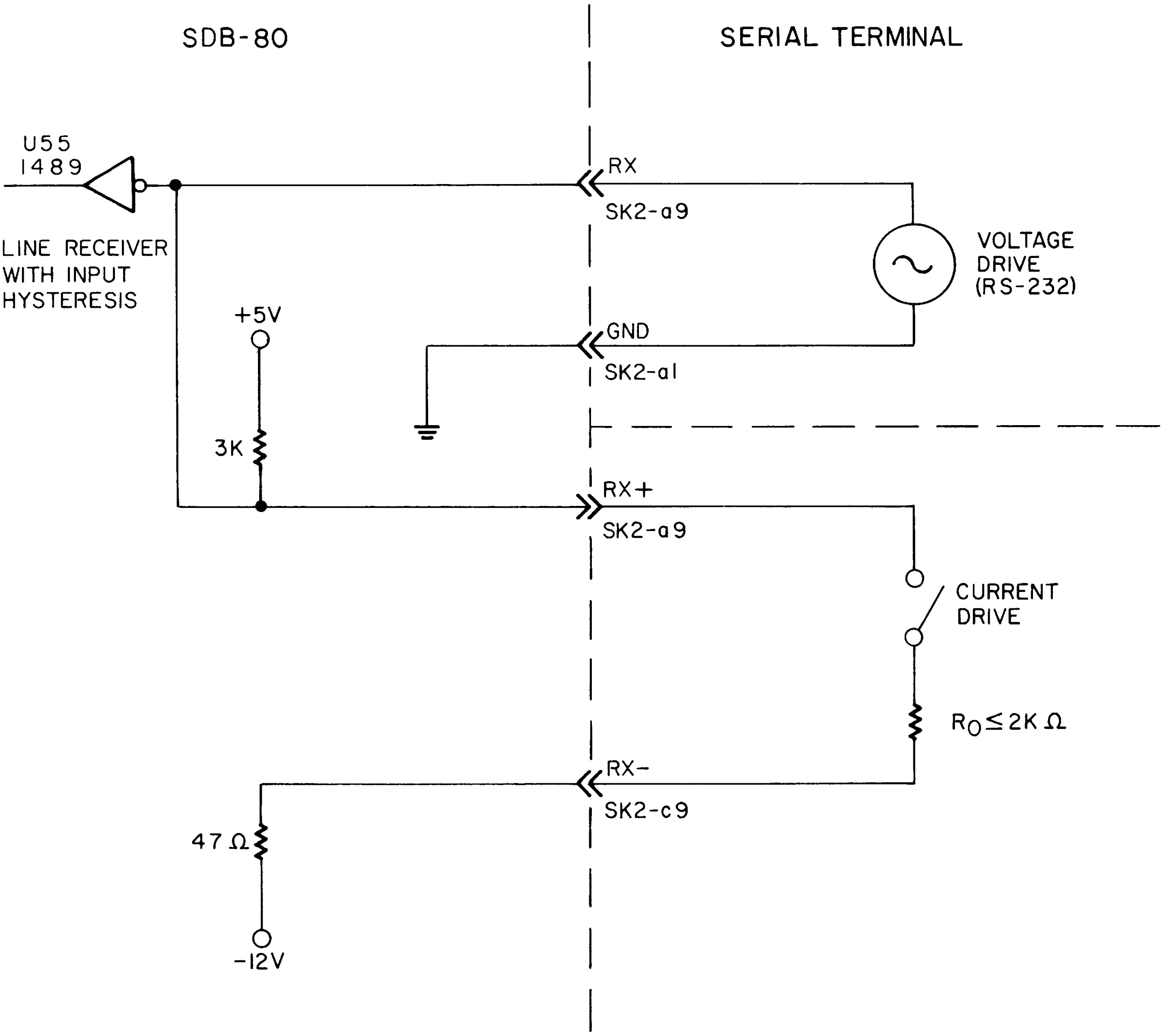


Figure 5

2) Current Drive-Teletype (TX^+, TX^-)

The output interface consists of a 12V supply, current limiting resistors, and an NPN switch to ground. This arrangement is capable of supplying more than the 20mA current required for a teletype.

RECEIVE SERIAL DATA (RX, RX^+, RX^-):

The incoming serial data communication link uses these three lines. The interface schematic of both the voltage and current drive situation is shown in Figure 5.

1) Voltage RS-232

A 1489 line receiver with input hysteresis receives the input voltage signal directly. This circuit detects the RS-232 voltage excursions' above and below ground. Typical threshold voltage levels for the part are -1 volt and +2 volt.

2) Current - Teletype

The input resistor/supply network shown in Figure 5 converts the unipolar. current drive to a bipolar voltage swing that the input line receivers (1489) can detect. Resistor and voltage values on the SDB-80 have been set to take into account the relatively high impedance closures of a teletype terminal.

HANDSHAKE LINES (DTR, RTS, DSR, CTS):

These four signals (shown in Figure 6) are general purpose control lines. DTR and RTS are clocked on to the DATA BUS D0 and D1, respectively, during a Read Port DE_H command. DSR and CTS output D0 and D1, respectively, during a Write To Port DE_H command. The D-FFs (U65) are driven by two upper 8 bit address lines which carry the same information as the DATA BUS during an 'out' instruction. This is done to reduce the capacitance on the DATA

HANDSHAKE LINES

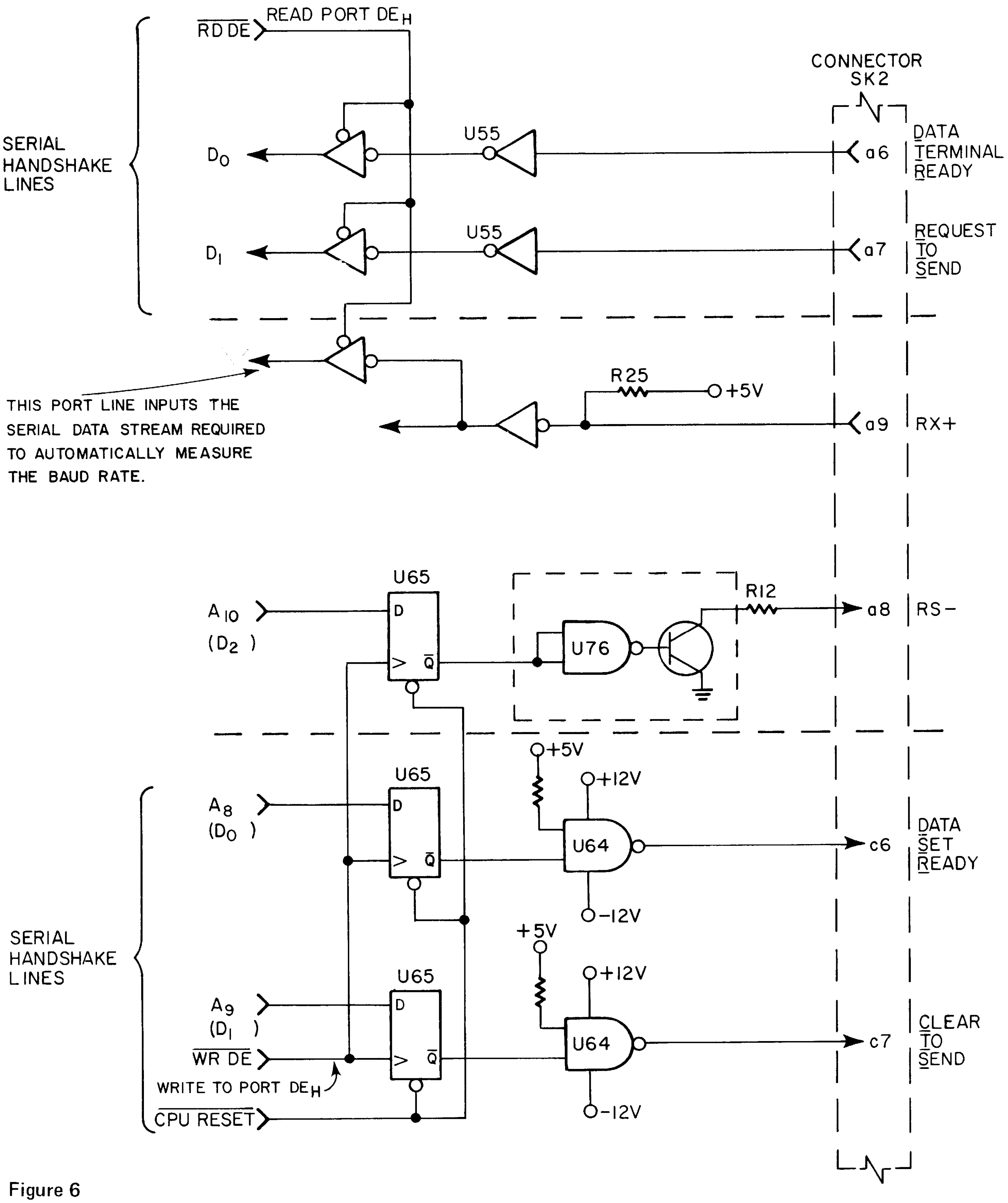


Figure 6

BUS. In OEM applications these four lines can be used as modem control lines to simplify interfacing to most any modem.

SILENT 700 CONTROL LINE (CDET)

The Carrier Detect signal (Figure 2) indicates to an RS-232 terminal that data can be sent over the interface line. CDET is wired to the ON condition (+12V) to continuously give the terminal an "OK to send" signal. This has been included specifically to allow the SDB-80 to interface directly with a Silent 700 terminal.

Two signals need additional comment.

1) SI - Serial IN

This port line inputs the serial data stream from the EIA or teletype terminal that is required by DDT-80 to measure the incoming baud rate and automatically adjust the SDB-80 baud rate correspondingly.

2) XP - $\overline{\text{XPAND}}$

This signal, found on SK2-a24, detects the presence of additional system boards. With no extra board $\overline{\text{XPAND}}$ is pulled high by a pull up resistor. This is recorded on D3 during a read of port DE_H. Inserting another board into the system (debug board for example) will force $\overline{\text{XPAND}}$ to a logic "0".

PARALLEL INTERFACE

Two Parallel I/O Controllers (MK 3881) are included on the SDB-80 Board. This gives four independent 8-bit I/O ports with two handshake (data transfer) control lines per port. All I/O lines are TTL buffered and have provision for termination resistors on board. Logically each port pair looks similar (depending upon option 1) to the on-card PIO devices. Figure 1 shows a diagram of a generalized parallel I/O port.

I. Connectors

All PIO connections are made over board connector SK2.

II. Resistor Terminations

One 14-pin socket per port is provided for resistor dual inline packages so that terminations may be placed on the data lines. A parallel termination is provided for each 8-bit port data line plus the input strobe (STB) handshake line. As shown in Figure 1, the termination resistors may be either simple pull up resistors (port A) or an impedance matching network (port B). The SDB-80 is shipped with four 1K Ω pull up terminations. In addition to the parallel termination resistors each ready (RDY) handshake output line is "terminated" with a series 47 Ω resistor on the board. This is used to help damp and reduce any reflections on this output line.

III. Handshake Line Buffers (STB,RDY)

Standard TTL Exclusive OR gates (7486) are used to buffer and isolate these lines. Jumper options are provided on board to independently control the polarity or "sense" of each handshake signal so as to ease the interfacing between the board and peripheral devices. The input

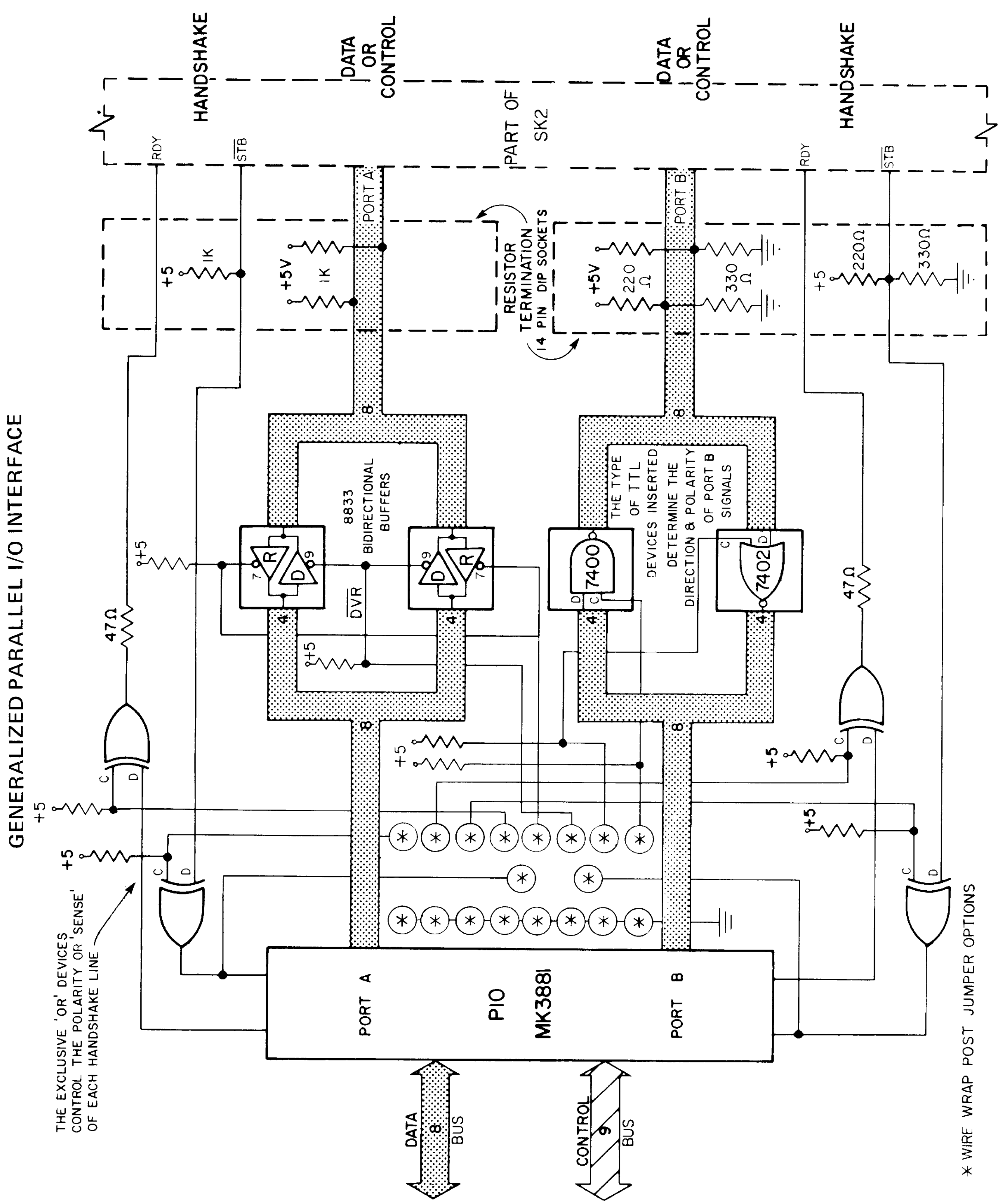

GENERALIZED PARALLEL I/O INTERFACE
HANDSHAKE
DATA OR CONTROL
PART OF SK2
DATA OR CONTROL
HANDSHAKE
RDY
STB
PORT A
PORT B
RESISTOR TERMINATION 14 PIN DIP SOCKETS
+5
1K
+5V
220 Ω
330 Ω
220Ω
330Ω
8833 BIDIRECTIONAL BUFFERS
DVR
THE TYPE OF TTL DEVICES INSERTED DETERMINE THE DIRECTION & POLARITY OF PORT B SIGNALS
7400
7402
47Ω
THE EXCLUSIVE 'OR' DEVICES CONTROL THE POLARITY OR 'SENSE' OF EACH HANDSHAKE LINE
PIO
MK3881
DATA 8 BUS
CONTROL 9 BUS
* WIRE WRAP POST JUMPER OPTIONS

Figure 1

Control and Data lines to each gate are marked C and D respectively in Figure 1. C controls the data as shown.

Control = logic "0"; Data = non-inverted

Control = logic "1"; Data = inverted

The following table indicates whether the handshake buffers are jumpered to invert or non-invert the PIO (chip) signals on the SDB-80 as shipped from the factory.

Data Line	Polarity of Buffer (as shipped from) factory
RDY (D0)	inverting
STB (D0)	inverting
RDY (D2)	inverting
STB (D2)	non-inverting
RDY (D4)	non-inverting
STB (D4)	non-inverting
RDY (D6)	non-inverting
STB (D6)	non-inverting

IV. Port A Data Buffer

Port A data bus lines are buffered using two quad party line non-inverting transceivers (DS8833). This allows true bidirectional capability. Jumper options allow for fixed IN, fixed OUT or BIDIRECTIONAL under software control. Replacing the DS8833 with a DS8835 effects a polarity change in the output bits.

The drivers and receivers (as designated by D and R respectively in Figure 1) are enabled by jumpers on sets of Wire Wrap pins. The enable lines are listed as $\overline{REC}$ for receiver enable and $\overline{DVR}$ for driver enable.

The jumper connections will be detailed later under VI Jumper information.

V. Port B Data Buffers

Port B data lines are arranged in such a fashion as to allow the user to determine the port direction (in increments of 4-bit sections). Sockets are provided for standard 14-pin 7400 series TTL packages. Depending upon the package type inserted, the port may be dedicated IN or OUT. In the output mode ports may be selected to provide standard; or buffered drive, active pull up or open collector, low or high voltage, etc.

A table showing the different types of devices that may be used to buffer port B is shown below:

IN	OUT
7402 STD drive, inverting (NOR)	7400 STD drive, inverting (NAND)
	7403 Open collector, inverting (NAND)
	7408 STD drive, non-inverting (AND)
	7409 Open collector, non-inverting (AND)
	7426 Open collector, high voltage, inverting (NAND)
	7432 STD drive, non-inverting (OR
	7437 Buffer, inverting (NAND)
	7438 Open collector, buffer, inverting (NAND)
	7486 STD drive, invert/non-inverting (EX-OR)

VI. Jumper Information

Each I/O makes use of a group of eighteen wire wrap jumper posts. For I.O #1 these are located between U44 and U45 and are numbered E66 through E83. The jumper post group that serves I/O #2 is located below the CTC and identified E84 through E101.

These allow the following jumper option functions:

1) Determine polarity of handshake lines by strapping the control line of the exclusive OR buffers U45 and U53.

2) Strap the control line on the buffers of port B for proper AND, NOR, or EX-OR operation.

3) Enable the receiver or driver portions of the port A buffers.

4) Control the bidirectional capability of port A.

Figure 2 shows the Wire Wrap pins jumpered so that each control line is strapped to a logical "0".

The following table summarizes all jumper options for the two I/O interfaces. Refer to Figures 3 and 4 which show the electrical configuration of each interface as shipped from the factory for an SDB-80.

1) Handshake lines

Designator	Wire Wrap Jumper Pins		
RDY (D0)			
INVERTED	E78	OPEN	# 1
NON-INVERTED	E78	STRAPPED	
$\overline{\text{STB (D0)}}$			
INVERTED	E74	OPEN	
NON-INVERTED	E74	STRAPPED	
RDY (D2)			
INVERTED	E76	OPEN	
NON-INVERTED	E76	STRAPPED	
$\overline{\text{STB (D2)}}$			
INVERTED	E72	OPEN	
NON-INVERTED	E72	STRAPPED	
RDY (D4)			
INVERTED	E85	OPEN	# 2
NON-INVERTED	E85	STRAPPED	
$\overline{\text{STB (D4)}}$			
INVERTED	E91	OPEN	
NON-INVERTED	E91	STRAPPED	

RDY (D6)			# 2 CON'T
INVERTED	E87	OPEN	
NON-INVERTED	E87	STRAPPED	
STB (D6)			
INVERTED	E89	OPEN	
NON-INVERTED	E89	STRAPPED	

2) Control lines - Port B ($D2_H$ or $D6_H$)

Buffer & Socket		Wire Wrap Pins	
7400, 7403, 7408, 7409 7426, 7437, 7438			
U61		E80	OPEN
U50		E82	OPEN
U63		E95	OPEN
U52		E93	OPEN
7402,7432			
U61		E80	STRAPPED
U50		E82	STRAPPED
U63		E95	STRAPPED
U52		E93	STRAPPED
7486			
U61	(INVERTING)	E80	OPEN
U61	(NON-INVERTING)	E80	STRAPPED
U50	(INVERTING)	E82	OPEN
U50	(NON-INVERTING)	E82	STRAPPED
U63	(INVERTING)	E95	OPEN
U63	(NON-INVERTING)	E95	STRAPPED
U52	(INVERTING)	E93	OPEN
U52	(NON-INVERTING)	E93	STRAPPED

3) Control lines - Port A ($D0_H$ or $D4_H$)

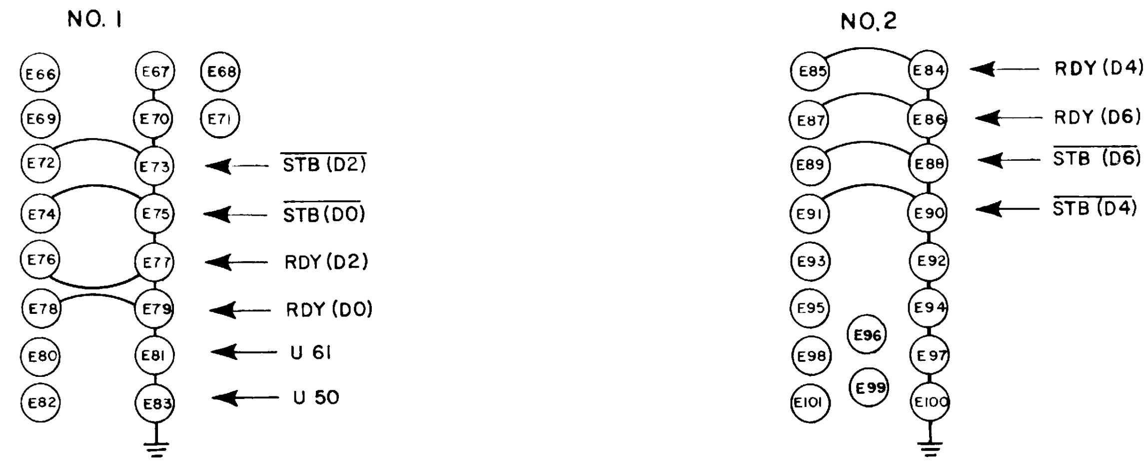
Direction: Port is Strapped "IN"

Socket	Wire Wrap Pins		Wire Wrap Pins	
U60	E69	STRAPPED	E66-68	OPEN
U49	E66	OPEN	E69-71	OPEN
U62	E98	STRAPPED	E99-101	OPEN
U51	E101	OPEN	E96-98	OPEN

Port is strapped "OUT"

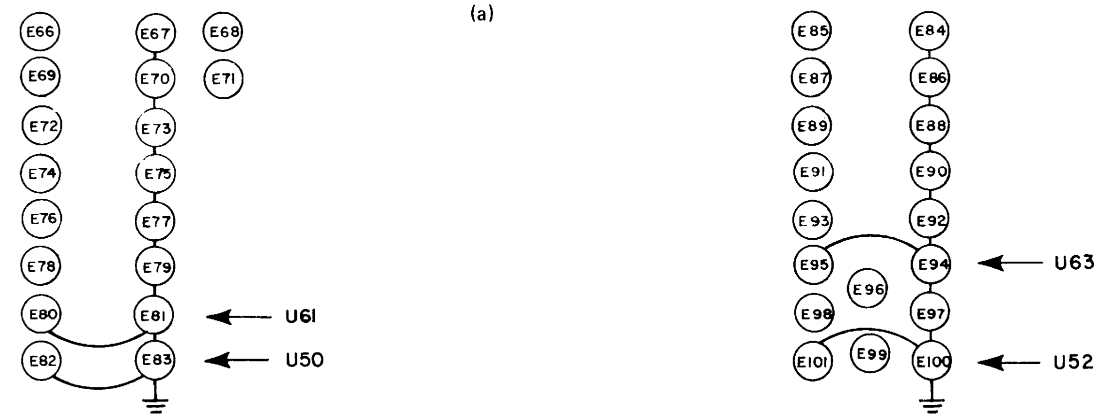
U60	E69	OPEN	E66-68	OPEN
U49	E66	STRAPPED	E69-71	OPEN
U62	E98	OPEN	E99-101	OPEN
U51	E101	STRAPPED	E96-98	OPEN

STRAPS TO CONTROL PARALLEL PORTS



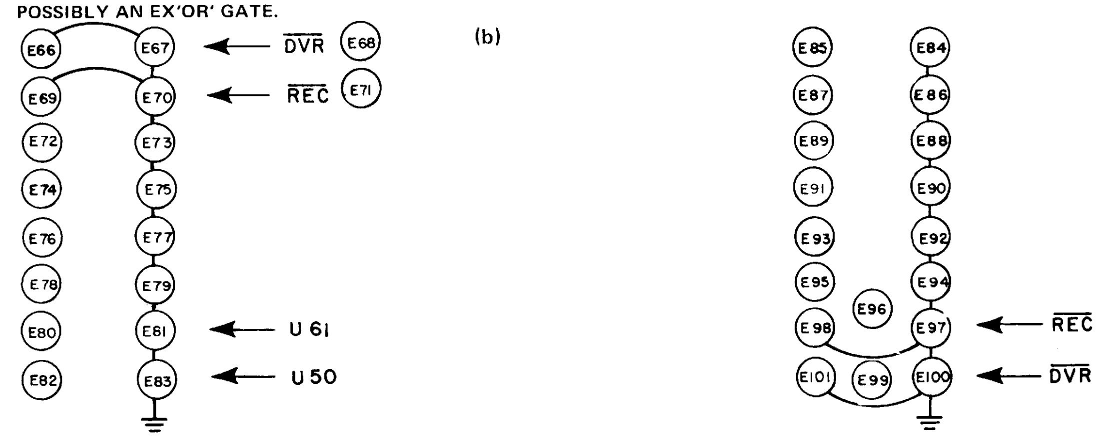
CONTROL LINE FOR EACH HANDSHAKE BUFFER IS STRAPPED TO A LOGIC "O" SO HANDSHAKE DATA IS NON-INVERTED. ABSENCE OF A STRAP INVERTS THE DATA.

(a)



CONTROL LINE FOR EACH BUFFER IS STRAPPED TO A LOGIC "O" SO THAT THE 'OR' OR 'NOR' BUFFER IS ENABLED. ABSENCE OF A STRAP APPLIES TO A 'AND','NAND', OR POSSIBLY AN EX'OR' GATE.

(b)



BOTH THE RECEIVER AND DRIVER CONTROL LINES ARE STRAPPED TO "O" CAUSING THE RECEIVERS AND DRIVERS TO BE ENABLED. IN OPERATION EITHER ONE MAY BE STRAPPED BUT NOT BOTH. IN THE BIDIRECTIONAL MODE NEITHER STRAP IS USED.

(c)

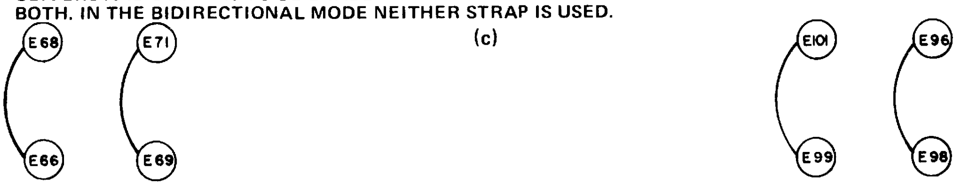


Figure 2

WIRE WRAP PINS ARE WIRED FOR BIDIRECTIONAL MODE.

PARALLEL I/O INTERFACE #1 AS SHIPPED FROM THE FACTORY

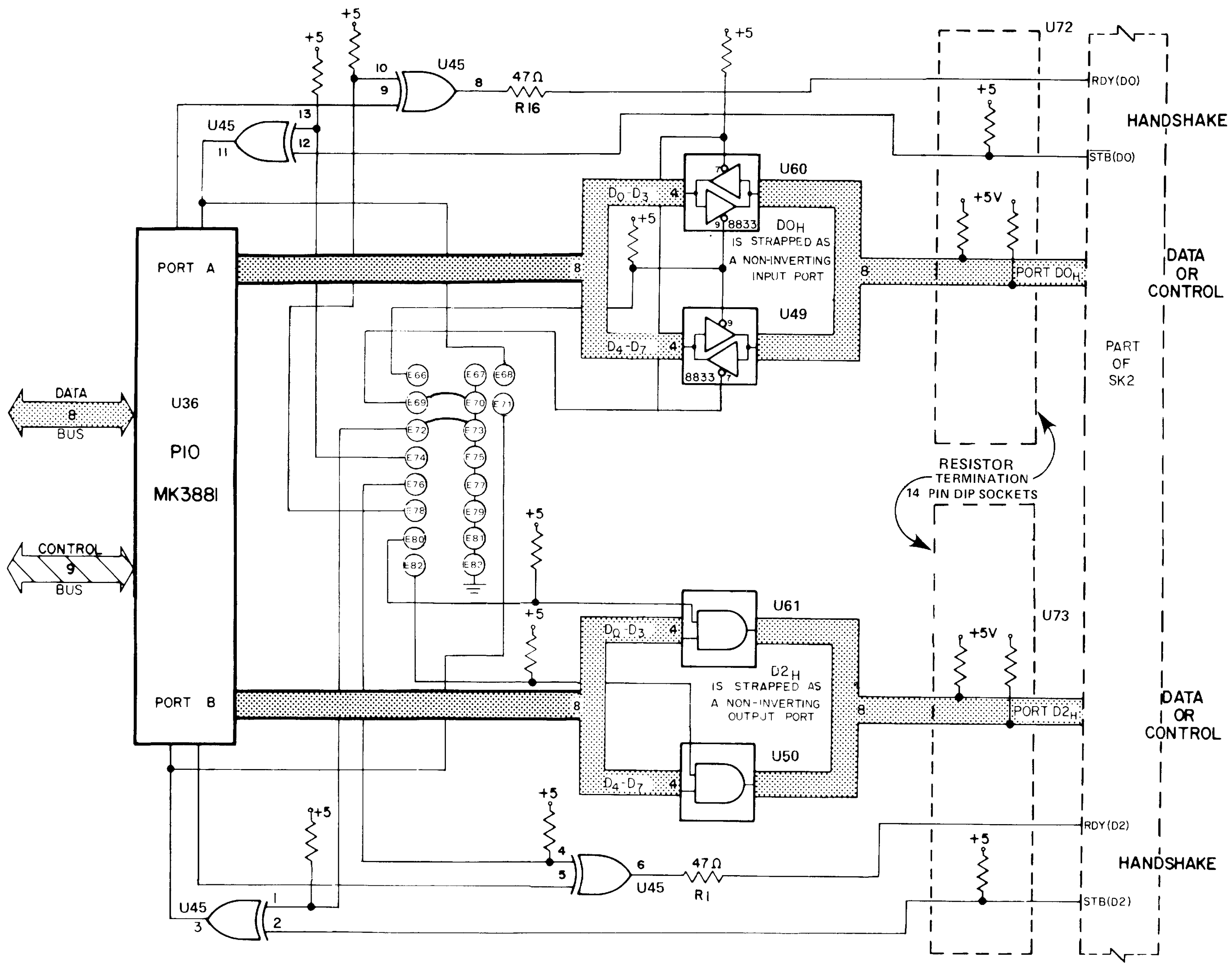


Figure 3

PARALLEL I/O INTERFACE #2 AS SHIPPED FROM THE FACTORY

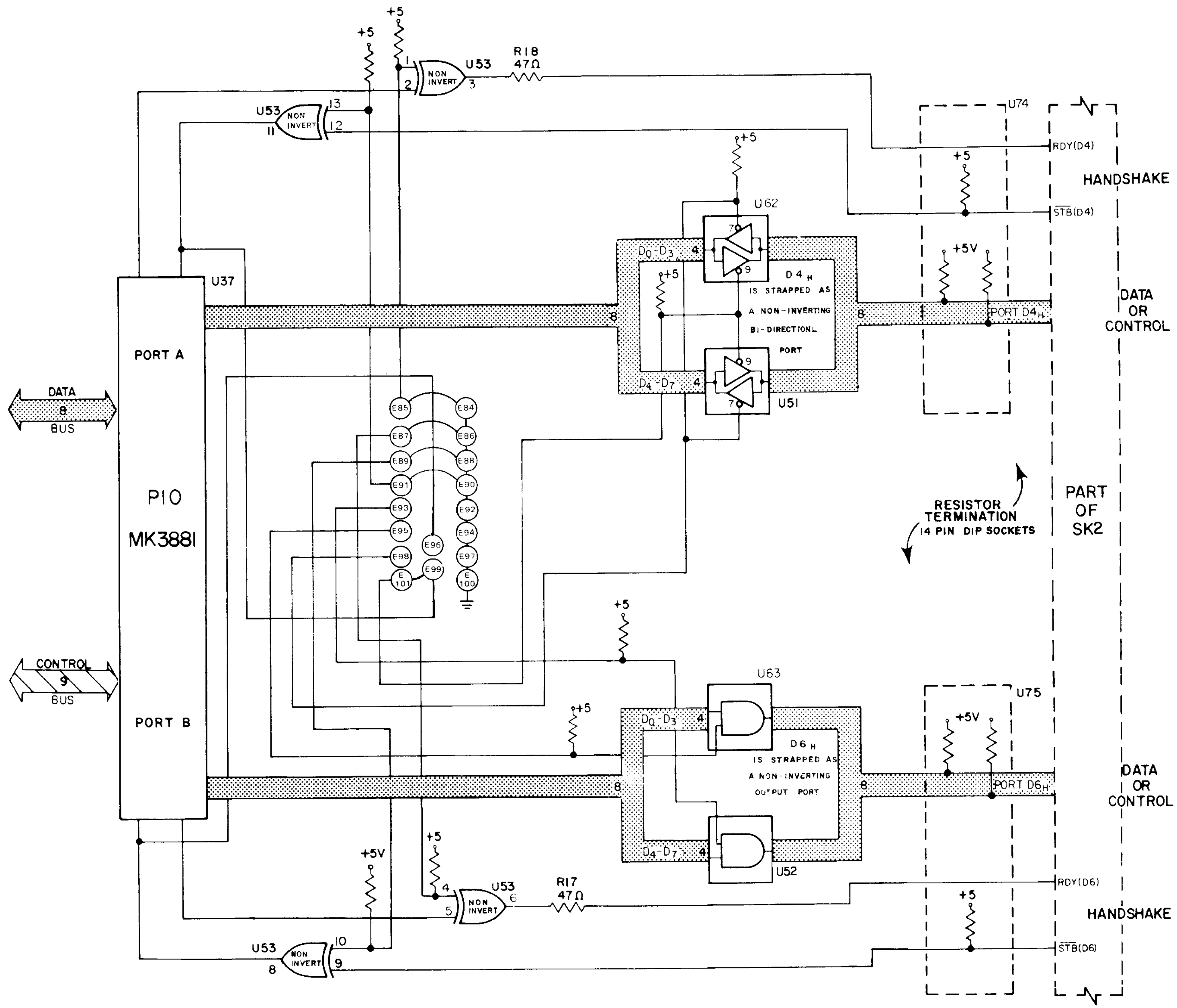


Figure 4

PARALLEL I/O #1 FOR CUSTOMER USE

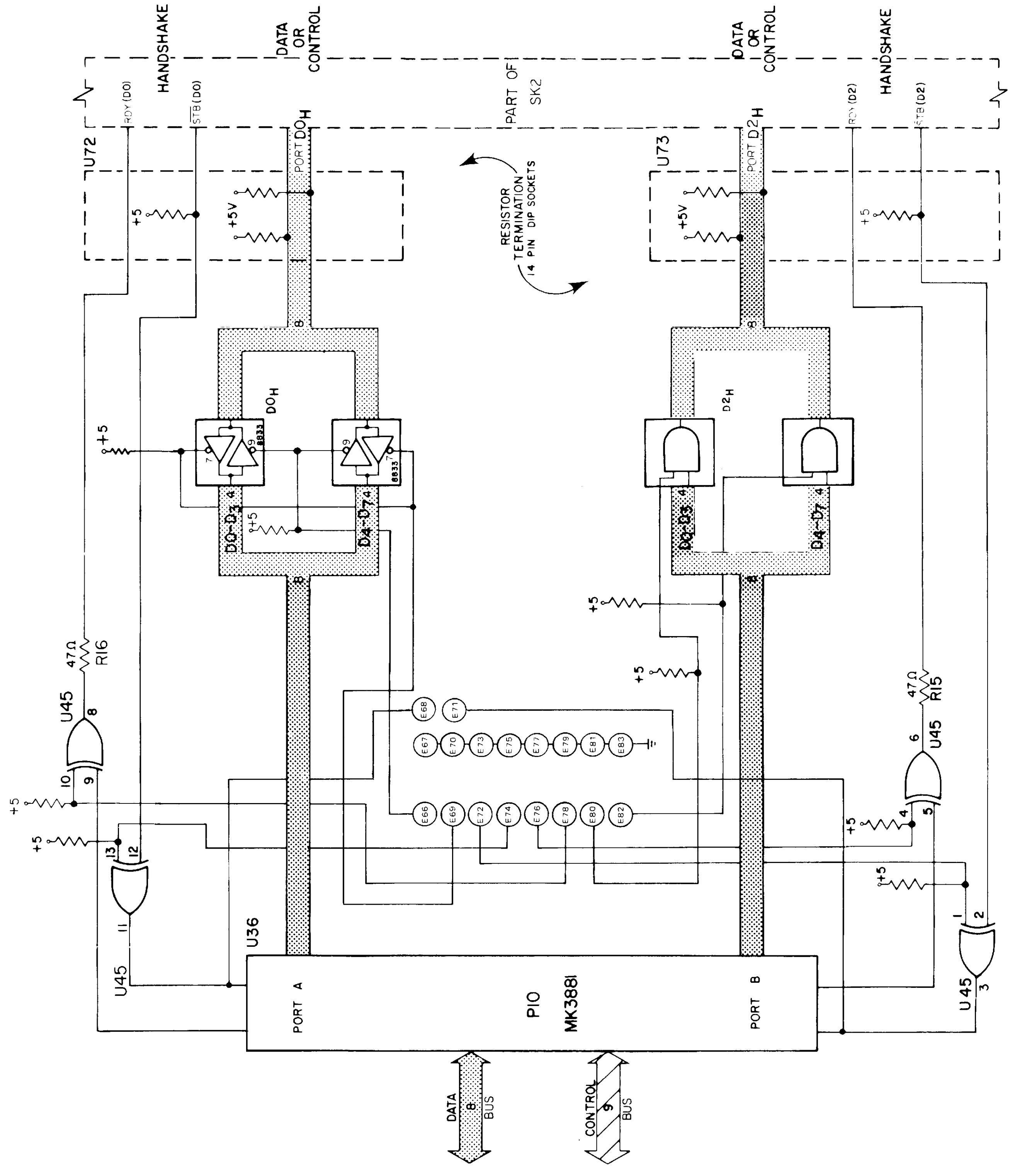


Figure 5

PARALLEL I/O #2 FOR CUSTOMER USE

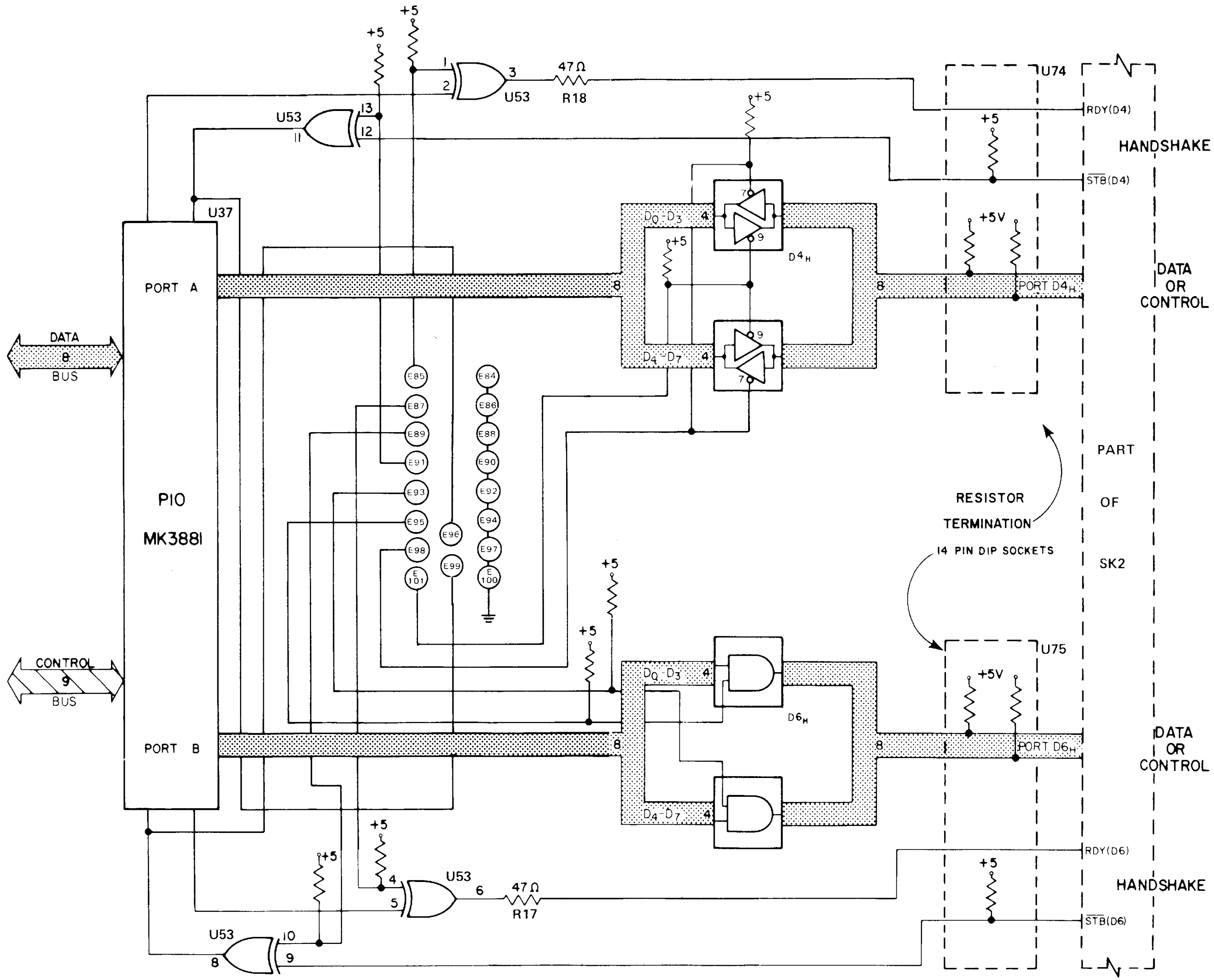


Figure 6

:Port is strapped "BIDIRECTIONAL"

U60	E69	OPEN	E66-68	CONNECTED
U49	E66	OPEN	E69-71	CONNECTED
U62	E98	OPEN	E99-101	CONNECTED
U51	E101	OPEN	E96-98	CONNECTED

Two blank schematics of interfaces #1 and #2 are included (Figures 5 and 6) to help the user effect any changes in the port options.

VII. Port Addresses

Each port in a PIO chip has two addresses; one for CONTROL and one for DATA. The port addresses are derived from the lowest 8-address lines (A0 - A7) A0 and A1 are fed directly to the PIO to select either control or data (A0) and port a or port B (A1). The rest of the addresses, A2 - A7 are decoded in the port section which provides a chip enable for each PIO. This $\overline{CE}$ function, along with A0 and A1, create the proper address for each port.

Port addresses for the SDB-80 are summarized below:

	PIO # 1		PIO # 2	
	Port A	Port B	Port A	Port B
DATA:	$D0_H$	$D2_H$	$D4_H$	$D6_H$
CONTROL:	$D1_H$	$D3_H$	$D5_H$	$D7_H$

MEMORY: MAPPING AND DECODING

A. MAPPING

Mapping versatility along with the large quantity of memory that is available on the SDB-80 are two of the board's key features. Figure 1 shows a block diagram that outlines the entire memory section. This board was designed with two uses in mind; software development and OEM applications. In a development system, support firmware (O. S., text editor, assembler, etc.) has been placed toward the top of the memory map while user memory has been allocated the space between addresses 0000_H and $9FFF_H$. Figure 2 is a generalized memory map showing this placement of memory for the SDB-80 software development system.

Sockets are provided for the following memory parts:

1) 4K bytes or 16K bytes of dynamic RAM-USER RAM
2) 256 x 8 Static RAM (¼ K) - O. S. RAM or Scratchpad RAM
3) 4 ROM/PROM Sockets - USER ROM/PROM, Sockets #1 - #4
4) 1 ROM/PROM Socket - O. S. ROM/PROM, Socket #5

The five ROM/PROM Sockets are capable of handling the following parts:

1K bytes MOS PROM MK 2708

1K bytes Bipolar PROM 82S2708 (Signetics)

1K bytes MOS ROM MK 30000

2K bytes MOS ROM MK 34000

4K bytes MOS ROM MK 32000

BLOCK DIAGRAM OF MEMORY SECTION

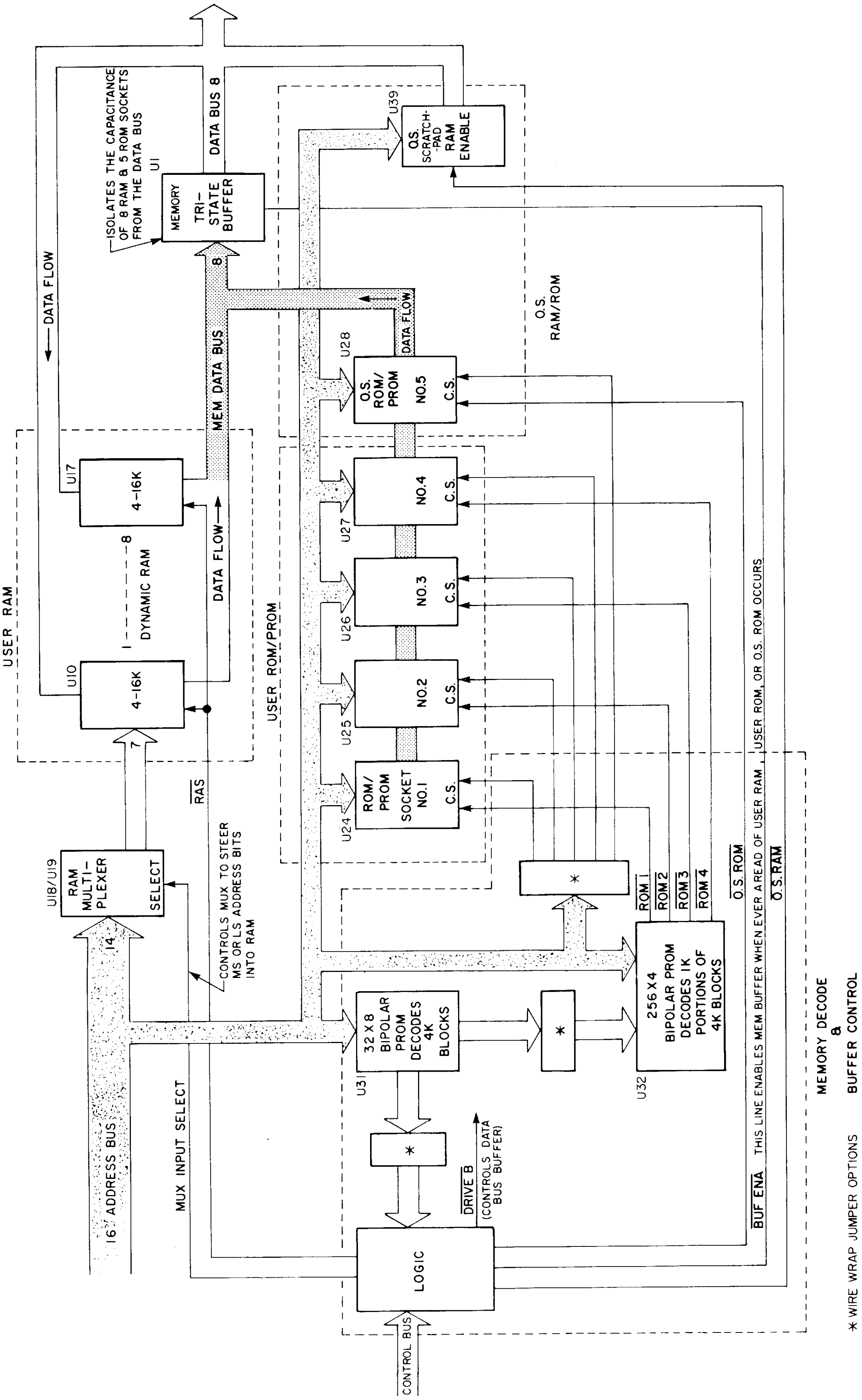


Figure 1

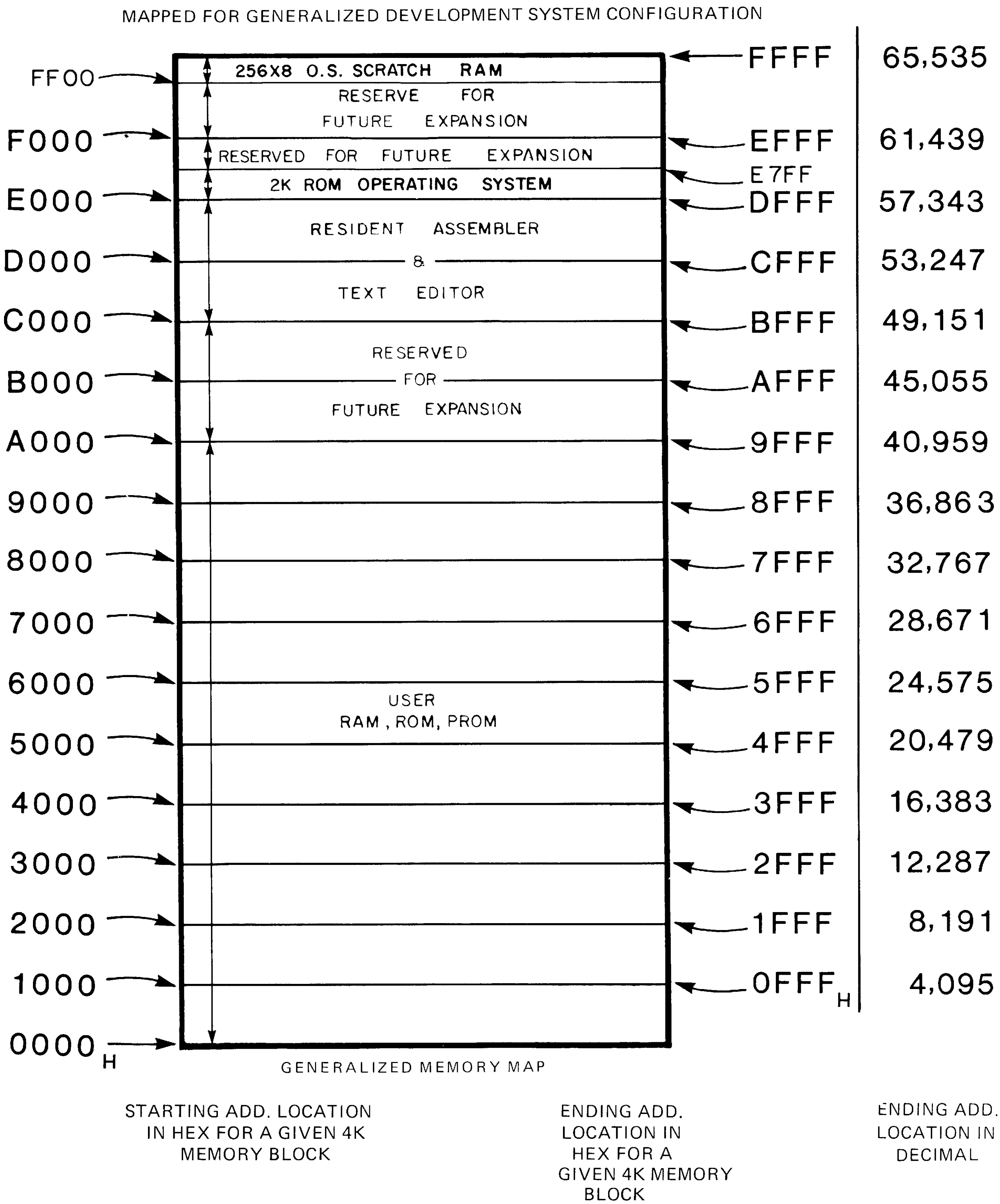

MAPPED FOR GENERALIZED DEVELOPMENT SYSTEM CONFIGURATION
FF00
F000
E000
D000
C000
B000
A000
9000
8000
7000
6000
5000
4000
3000
2000
1000
0000H
256X8 O.S. SCRATCH RAM
RESERVE FOR FUTURE EXPANSION
RESERVED FOR FUTURE EXPANSION
2K ROM OPERATING SYSTEM
RESIDENT ASSEMBLER & TEXT EDITOR
RESERVED FOR FUTURE EXPANSION
USER RAM , ROM, PROM
FFFF 65,535
EFFF 61,439
E7FF
DFFF 57,343
CFFF 53,247
BFFF 49,151
AFFF 45,055
9FFF 40,959
8FFF 36,863
7FFF 32,767
6FFF 28,671
5FFF 24,575
4FFF 20,479
3FFF 16,383
2FFF 12,287
1FFF 8,191
0FFFH 4,095
GENERALIZED MEMORY MAP
STARTING ADD. LOCATION IN HEX FOR A GIVEN 4K MEMORY BLOCK
ENDING ADD. LOCATION IN HEX FOR A GIVEN 4K MEMORY BLOCK
ENDING ADD. LOCATION IN DECIMAL

Figure 2

This gives a total on board memory capacity of:

RAM: 16K bytes + 1/4 K bytes

ROM: 20K bytes

An SDB-80 as shipped from the factory would reflect the memory map as shown in Figure 3. Note: The two possible options with respect to the USER RAM section.

In an OEM situation the support firmware is generally not needed. Therefore, MOSTEK offers the SDB-80 board stripped of its ROMs. The memory remaining is the 256 X 8 scratchpad RAM and the 4K/16K dynamic RAM option. On an OEM board (OEM-80) this memory can be mapped to the users desired locations. Figure 4 shows the memory placement on an OEM-80 board as shipped from the factory.

Until now computer boards have not offered the quantity of memory as is available on the SDB-80. There are two basic reasons for this:
1) Only recently have high density, pin compatable memories suitable for use with microprocessors become available. MOSTEK has pioneered the concept whereby various size memories have the same pin out and all will operate in the same sockets.
2) Until now, high density dynamic RAMs have not been used extensively on microprocessor boards because of the overhead logic required for the refresh operation. Now, with the internal refresh capability of the Z-80 CPU, transparent refresh is much simpler and more practical to implement.

SDB-80 MEMORY MAP

MAPPED FOR DEVELOPMENT SYSTEM CONFIGURATION
AS SHIPPED FROM FACTORY

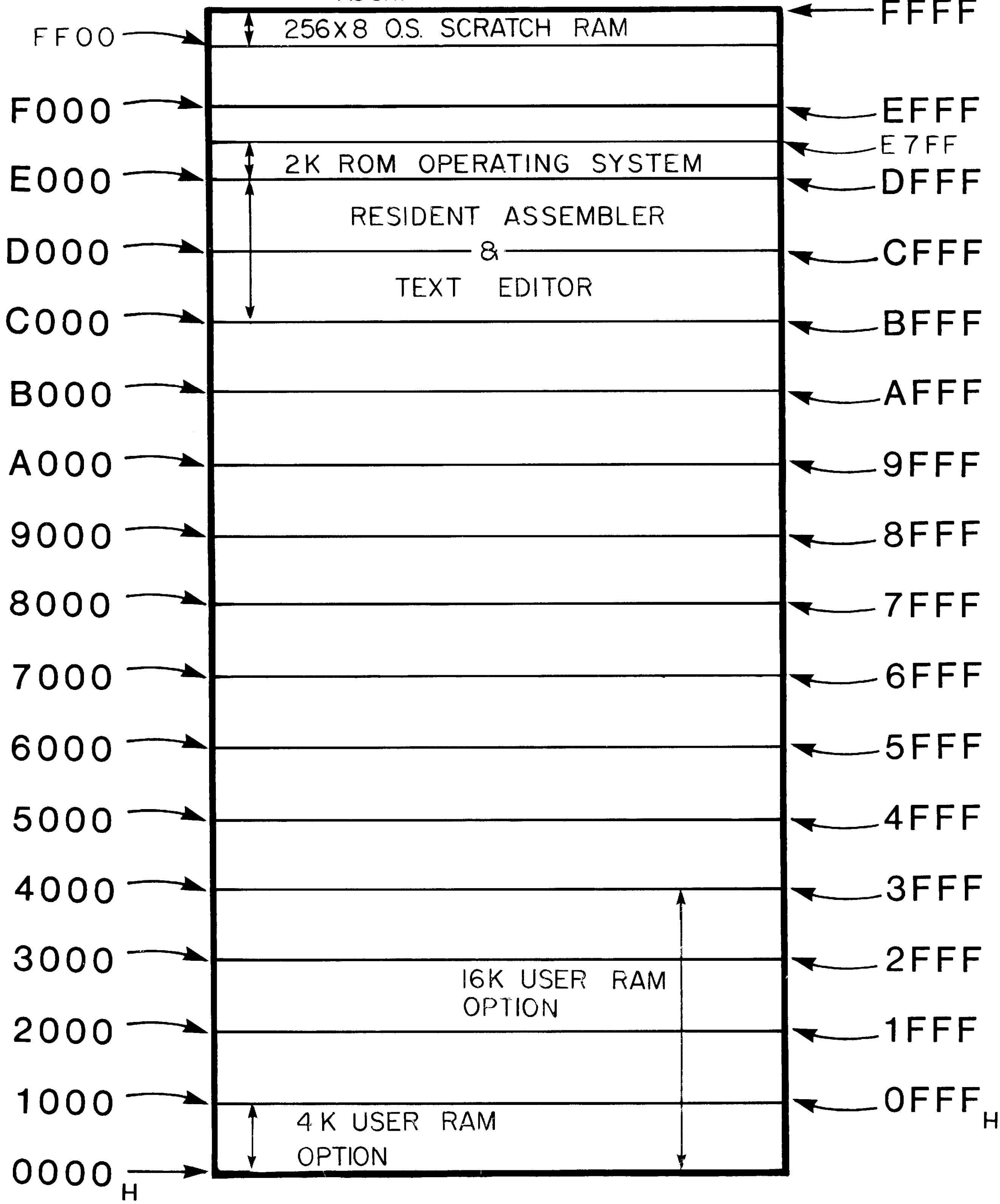


Figure 3

MAPPED FOR OEM SYSTEM CONFIGURATION
AS SHIPPED FROM FACTORY

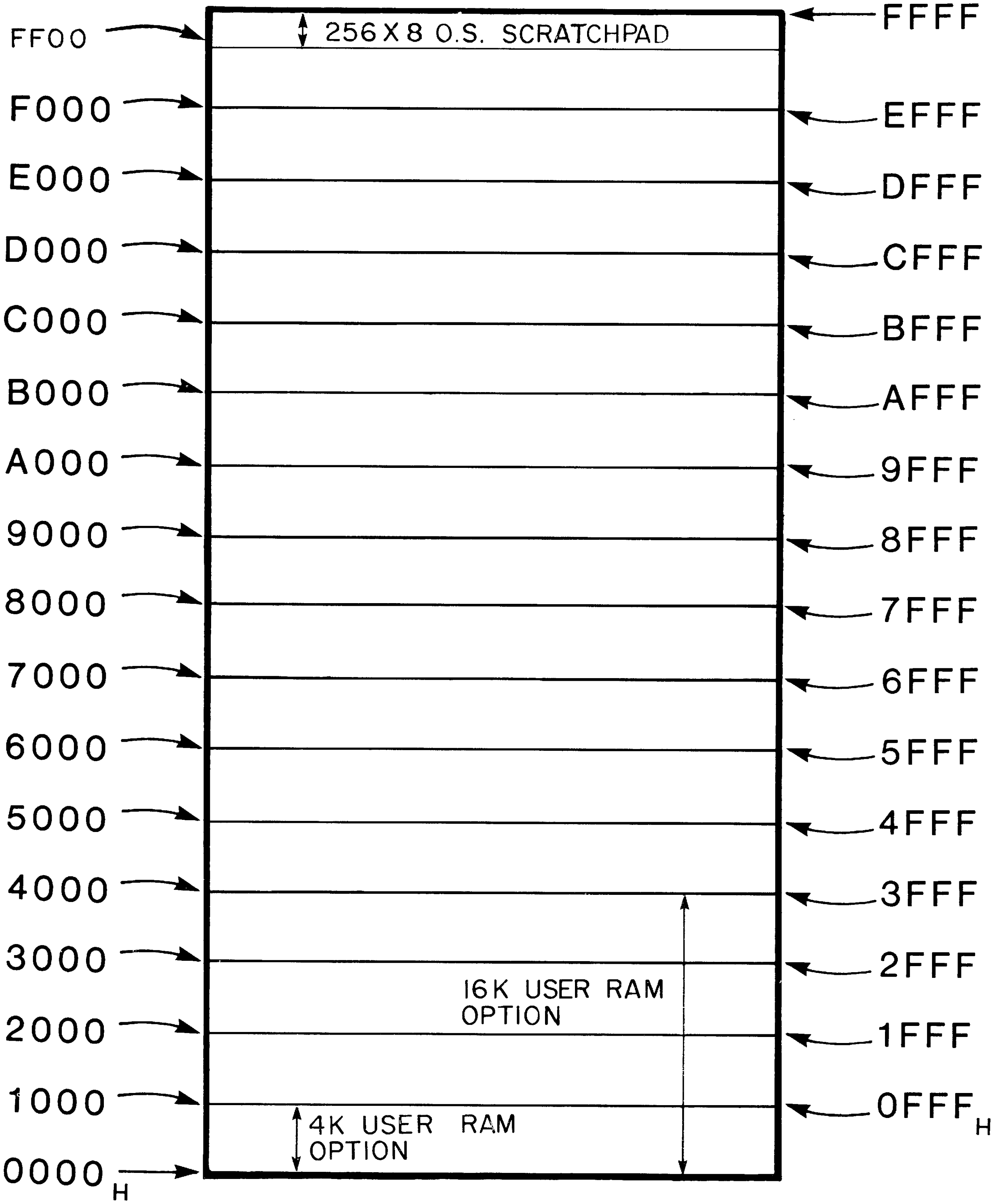


Figure 4

The memory decoding logic is arranged such that this memory can be placed anywhere in the 65K memory map (with few exceptions). Selection of memory locations is accomplished by using jumpers and bipolar fusable link PROMs.

In describing the memory section it will be convenient to follow the order described in 1 through 4 above. Figure 5 is a photograph of the SDB-80 showing, in detail, the component areas comprising the memory section.

I. USER RAM

Eight sockets are provided for either 4K (MK 4027) or 16K (MK 4116) dynamic RAMs. These 16 pin parts are fully interchangeable, requiring only three jumper option changes on board. Using signals from the Z-80 CPU, refresh occurs during an unused portion of the instruction cycle making the refresh "transparent" to the user.

II. OPERATING SYSTEM RAM OR SCRATCHPAD RAM

One 22 pin socket is provided for a 256x 8 static RAM. In a developement system, this RAM would be used by the system software as temporary storage, thus leaving the 4 or 16K user RAM free for the user. Also, it allows the 4 or 16K RAM to be mapped anywhere in memory without affecting the debug software. In an OEM system, this RAM may or may not be required.

III. USER ROM/PROM, O. S. ROM

Five 24 pin ROM/PROM Sockets are provided on the SDB-80. Four are for general usage while the fifth is reserved for a 2K O. S. ROM in a development system. All are separately decoded and can be placed contiguously in the memory map for programs requiring up to 20K bytes of ROM.

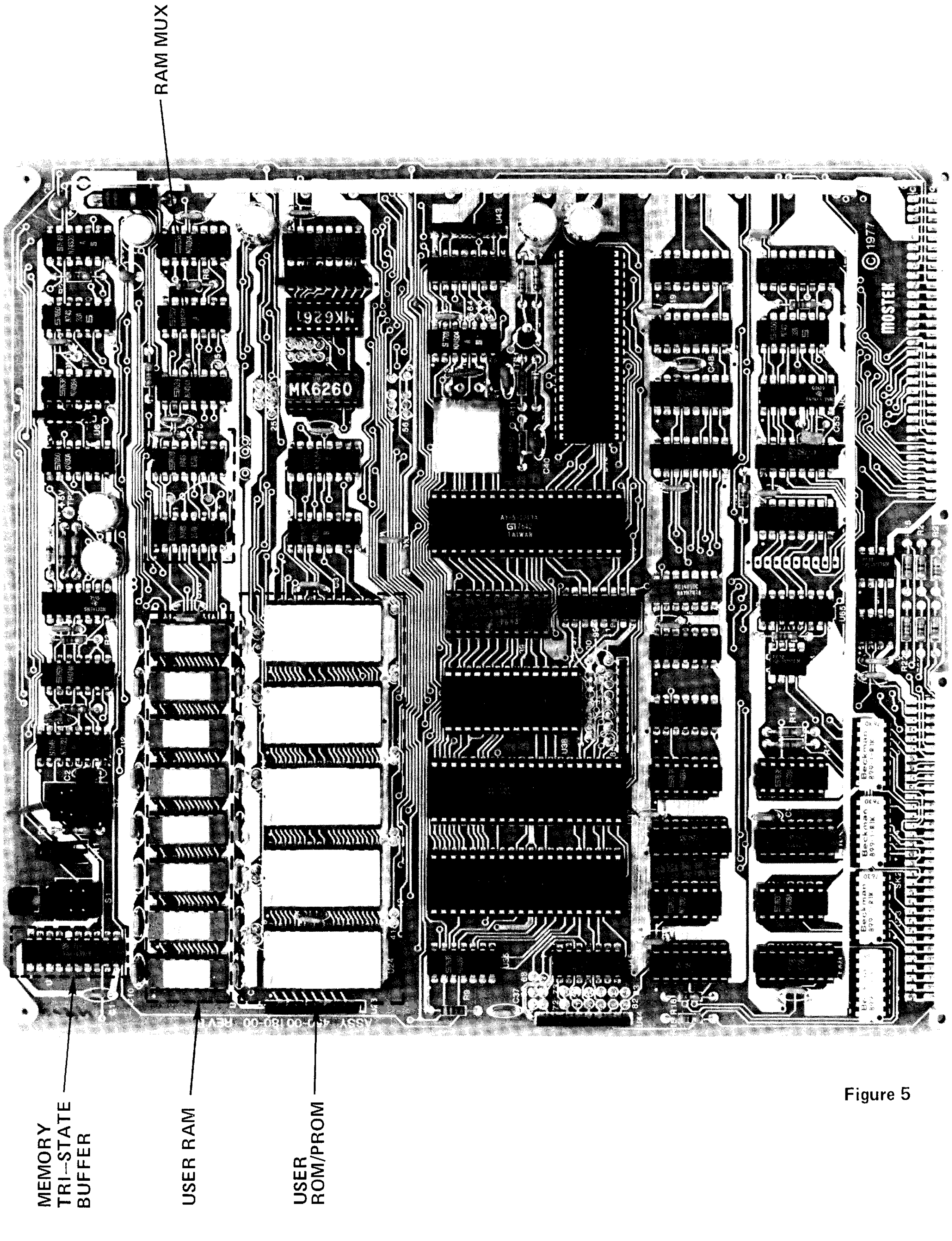

RAM MUX
MEMORY TRI–STATE BUFFER
USER RAM
USER ROM/PROM
MK6260
MOSTEK
© 1977

Figure 5

B. DECODING

The circuitry required for handling the memory on board is illustrated in the block diagram of Figure 1. Information for the various combinations of memory placement and size is stored in the two bipolar PROMs listed as U32 and U31. These parts are preprogrammed at the factory with the most useful combinations of memory mapping. However, because these are finite size PROMs, all combinations cannot be included. Therefore, these parts are socketed and may be replaced with differently patterned PROMs if desired.

Memory mapping and decoding on the SDB-80 is specifically accomplished through the use of:

1) Data stored in two bipolar PROMs
2) Jumper options on the board
3) TTL logic gates

(refer to the schematic of Figure 6)

I) 32 x 8 BIPOLAR PROM (U31)

The four high order address lines (A_{12} - A_{15}) enable this PROM to create the basic 4K boundaries shown in Figure 2. Its eight outputs determine the placement (within a given 4K boundary) of all memory on the board. Figure 7 shows the programmed 4K (or multiple of 4K) locations stored in the PROM as shipped from the factory.

The jumper option, E5, enables the user to interchange RAM and ROM between the lower and upper portions of the memory map. (As might be done when switching from a development situation to an OEM situation).

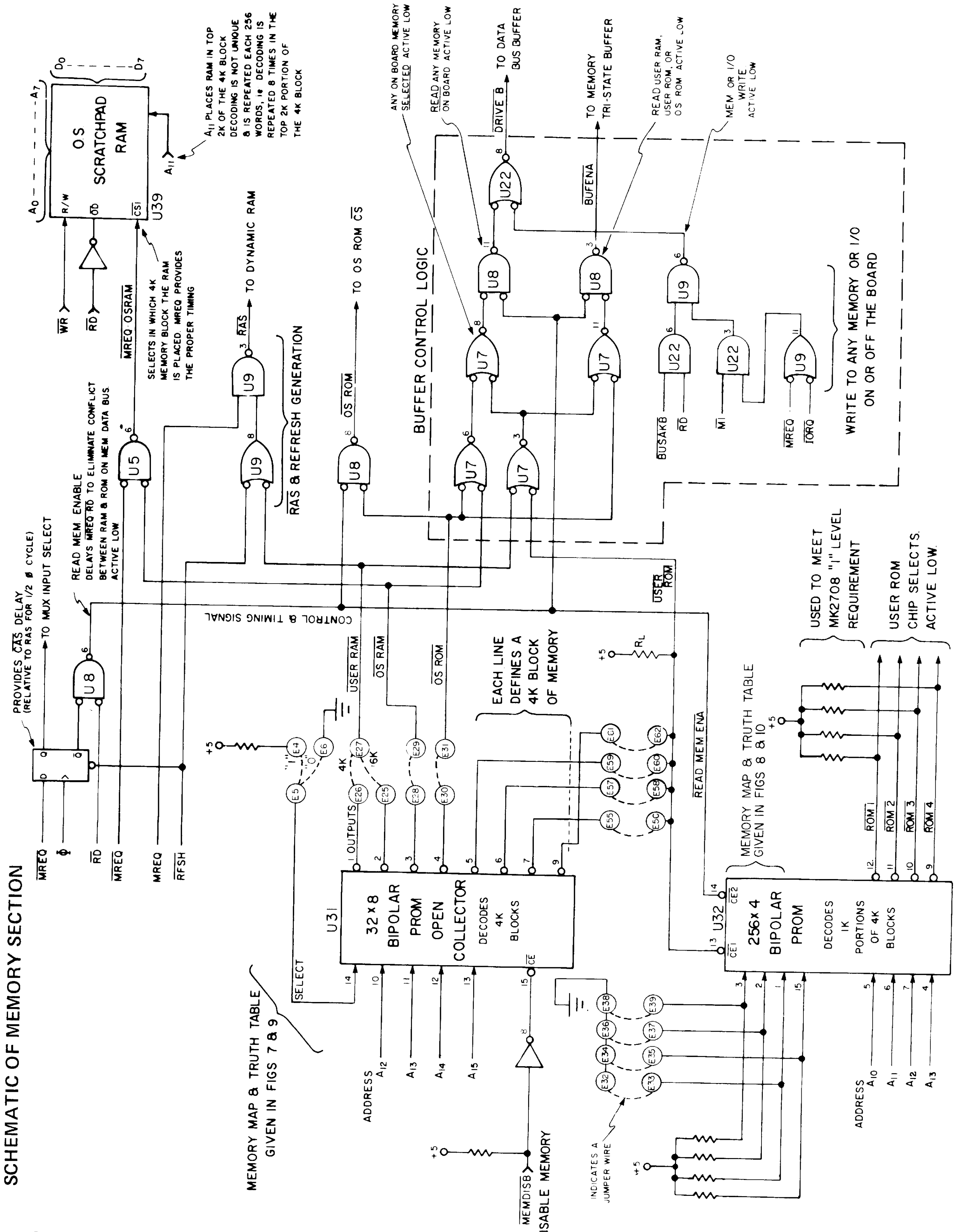

SCHEMATIC OF MEMORY SECTION
OS SCRATCHPAD RAM
U39
A11 PLACES RAM IN TOP 2K OF THE 4K BLOCK DECODING IS NOT UNIQUE & IS REPEATED EACH 256 WORDS, ie DECODING IS REPEATED 8 TIMES IN THE TOP 2K PORTION OF THE 4K BLOCK
SELECTS IN WHICH 4K MEMORY BLOCK THE RAM IS PLACED. MREQ PROVIDES THE PROPER TIMING
MREQ·OSRAM
TO DYNAMIC RAM
RAS & REFRESH GENERATION
TO OS ROM CS
OS ROM
BUFFER CONTROL LOGIC
ANY ON BOARD MEMORY SELECTED ACTIVE LOW
READ ANY MEMORY ON BOARD ACTIVE LOW
DRIVE B
TO DATA BUS BUFFER
TO MEMORY TRI-STATE BUFFER
BUFENA
READ USER RAM, USER ROM, OR OS ROM ACTIVE LOW
MEM OR I/O WRITE ACTIVE LOW
WRITE TO ANY MEMORY OR I/O ON OR OFF THE BOARD
BUSAKB
RD
M1
MREQ
IORQ
PROVIDES CAS DELAY (RELATIVE TO RAS FOR 1/2 Ø CYCLE)
TO MUX INPUT SELECT
READ MEM ENABLE DELAYS MREQ·RD TO ELIMINATE CONFLICT BETWEEN RAM & ROM ON MEM DATA BUS. ACTIVE LOW
CONTROL & TIMING SIGNAL
USER RAM
OS RAM
OS ROM
USER ROM
EACH LINE DEFINES A 4K BLOCK OF MEMORY
READ MEM ENA
U31
32 x 8 BIPOLAR PROM OPEN COLLECTOR
DECODES 4K BLOCKS
OUTPUTS
SELECT
MEMORY MAP & TRUTH TABLE GIVEN IN FIGS 7 & 9
ADDRESS
MEMDISB
DISABLE MEMORY
INDICATES A JUMPER WIRE
U32
256x4 BIPOLAR PROM
DECODES 1K PORTIONS OF 4K BLOCKS
MEMORY MAP & TRUTH TABLE GIVEN IN FIGS 8 & 10
USED TO MEET MK2708 "1" LEVEL REQUIREMENT
USER ROM CHIP SELECTS. ACTIVE LOW.
ROM 1
ROM 2
ROM 3
ROM 4

Figure 6

Outputs 1-4 are each associated with a particular memory. Outputs 5-9 are used only to form a CE function for the second PROM, U32. This CE determines in which 4K block the second PROM can be active. Because the outputs of U31 are open collector, these four lines may be wire-ORed so that U32 can be enabled over a range of from 4K to 16K bytes of memory.

$\overline{\text{MEMDISB}}$ (memory disable) is an input to the card through connector SK1. Its function is to disable this PROM and thus <u>all</u> memory on board. This line can be used if there is off-board memory mapped into the same memory location as the on-board memory. $\overline{\text{MEMDISB}}$ disables the on-board memory to resolve the conflict.

II. 256 x 4 BIPOLAR PROM (U32)

This PROM is used to further subdivide the 4K blocks into 1K segments and to provide the four user ROM/PROM Sockets with chip select lines. U32 is selected whenever memory is read ($\overline{\text{READ MEM ENA}}$, $\overline{CE_2}$) and when U31 selects a USER ROM ($\overline{\text{USER ROM}}$, $\overline{CE_1}$).

Of this port's eight input address lines the least significant four are connected to the CPU address lines A_{10} - A_{13}. This allows the PROM to decode from 1K up to 16K memory segments in 1K increments (depending, of course, on the stored data). Pins 1, 2, 3 and 15 are the divices four MSBs and are jumper options E32 through E39. This option enables the user to select 1 of 16 possible combinations involving the USER ROM/PROM Sockets.

$\overline{\text{ROM}}$ 1 and $\overline{\text{ROM}}$ 4 are the four enable lines to the four USER ROM Sockets. Any of these lines go active LOW depending upon the condition of the 1) address, 2) jumper option, 3) chip enables lines, and 4) stored program. The outputs are not open collector, the pullup resistors are used to meet the "1" level requirements for any 2708 MOS PROMS used.

III) OPERATION OF THE PROM DECODING

Figure 7 is a map of the information stored in U31. Each arrow indicated a 4K, or multiple, block that is decoded to enable specific memory parts on board. The memory map can best be described by considering each output separately.

1) Output 1 - 4K RAM

Connecting output 1 (E26) to "$\overline{\text{USER RAM}}$" (E27) enables a 4K dynamic RAM starting a either 0000_H or at 4000_H depending upon whether Jumper E5 is a "0" or "1" respectively.

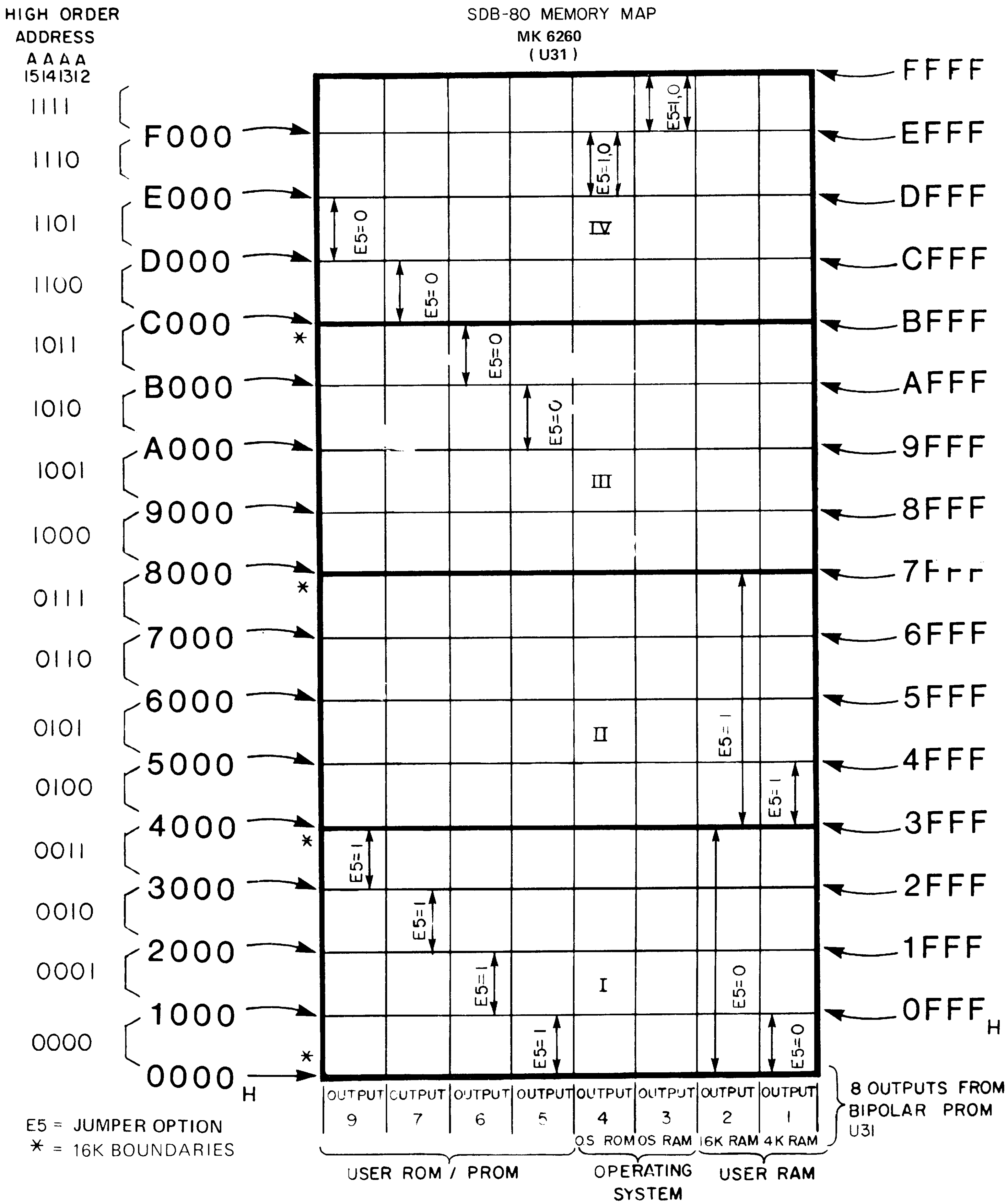
2) Output 2 - 16K RAM

Connecting output 2 (E25) to "$\overline{\text{USER RAM}}$" (E27) will upgrade the board to 16 bytes of RAM starting at either 0000_H or 40000_H depending upon the logic state of E5.

3) Output 3 - O. S. RAM

4) Output 4 - O. S. ROM

These parts are mapped at the top of memory as shown for E5 equal to either a "1" or a "0". This is because in the software development system, the O. S. RAM and ROM must be fixed at a specific location regardless of user memory placement. To place the O. S. RAM at the top of the map requires output 3 (E28) be connected to "$\overline{\text{OS RAM}}$". Only 8 of the 12 bits are decoded, bits 8 - 11 are 'don't care'.



MEMORY MAP SHOWING DECODED MEMORY LOCATIONS ON 4K BOUNDARIES

Figure 7

3) Outputs 5, 6, 7, 9

These four lines tell the second PROM (U32) in what 4K blocks it can enable the USER ROM Sockets. These outputs are open collector such that they may be tied together in any combination to define the amount of USER ROM decoded (from 4K to 16K). As with the other parts, the jumper E5 determines the placement (low or high) in the map.

The rationale for the various placements of memory as defined by Figure 7 is as follows:

1) The operating system has been placed "out of the way" at the top of the map.

2) Either RAM or ROM/PROM can be started at address 0000_H depending upon the application. Usually in a development situation it is desirable to start RAM (and thus start building the software program) at location 0_H. However, once the program is written, debugged, and placed into PROM for evaluation it is desirable to be able to start the PROMs at 0_H (because this is where the program was written).

In an OEM application the decision where to place memory is usually application dependant. This is why MOSTEK has given the user such flexability in placing his memory. If the pattern shown in Figure 7 is not suitable, the user can customize his own PROM. Programing information for this PROM will be given later.

3) For its development station MOSTEK has defined the memory space from $A000_H$ to $E000_H$ for its firmware support package (see Figure 2). This is defined by outputs 5, 6, 7, 9 and E5 = 0. In an OEM situation these memory spaces are available to the user. Note that in this upper position these four 4K blocks are adjacent to the O. S. block giving the user five 4K contiguous memory blocks (for 20K of contiguous ROM) if desired.

If all ROM/PROM were in 4K blocks then the only decode necessary would be U31 (and the map in Figure 7). However, because of 1K and 2K ROM/PROMS it is necessary to subdivide the 4K blocks of Figure 7. This is the function of the bipolar PROM U32; to decode 1K memory segments. U31 decodes specific 4K blocks, then enables U32, which in turn divides each block into four 1K segments. Specific USER ROM memories are selected by the lines $\overline{\text{ROM 1}}$ - $\overline{\text{ROM 4}}$ (See Figure 6). The map of U32 is shown in Figure 8.

This map is a 16K subset of the more general map shown in Figure 7. Note that Figure 7 is divided into four 16K areas. Figure 8 represents an expanded view of any one of these areas. It must start on a 16K boundary line. This is true basically because two of the address lines to U32 are common with U31 (A_{12} & A_{13}).

The left hand side of the map represents the 16 memory locations as defined by the four input address lines (shown both in decimal and hex). The bottom of the map represents the 16 possible memory configurations assigned to the four USER RAM Sockets. Selection is made by the input jumper lines E55 through E62 (shown in both decimal and hex). As previously stated with PROM, U31 this part is socketed so that changes in the pattern can be effected. Programming information will be given later.

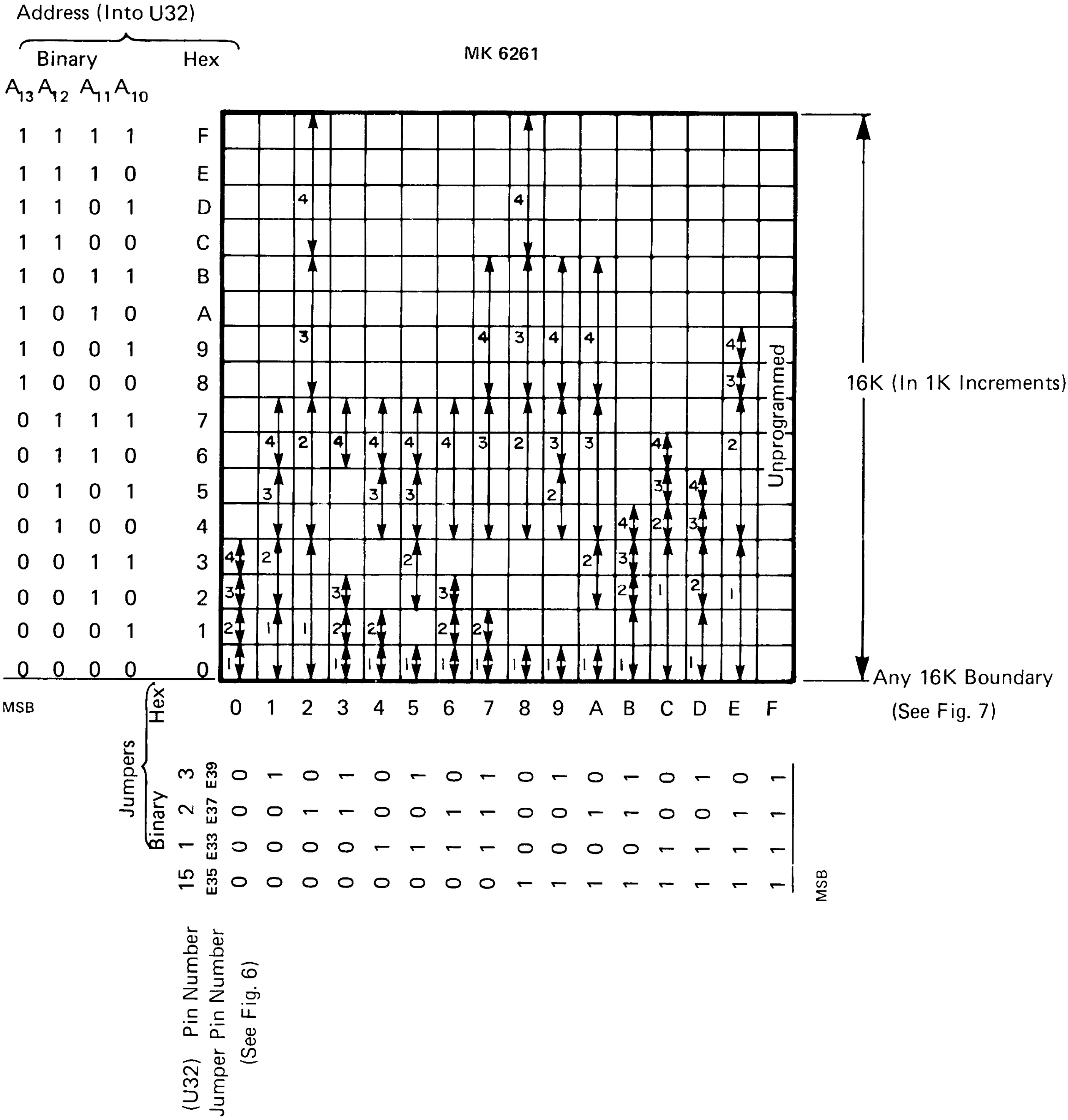

Address (Into U32)
Binary
Hex
A_{13} A_{12} A_{11} A_{10}
MK 6261
Unprogrammed
16K (In 1K Increments)
Any 16K Boundary
(See Fig. 7)
MSB
Jumpers
Hex
Binary
(U32) Pin Number
Jumper Pin Number
(See Fig. 6)
15 1 2 3
E35 E33 E37 E39
MSB

Figure 8

The select line for each socket is defined in the following way:

1) $\overline{\text{ROM 1}}$ - Socket 1 (U24)

2) $\overline{\text{ROM 2}}$ - Socket 2 (U25)

3) $\overline{\text{ROM 3}}$ - Socket 3 (U26)

4) $\overline{\text{ROM 4}}$ - Socket 4 (U27)

5) (The OS ROM will be referred to as socket 5 - (U28)

The numbers associated with each arrow in the map refer to USER ROM Socket numbers.

Several examples of memory placement will illustrate the use of the maps of Figures 7 * 8.

1) Assume the SDB-80 is to have a 4K RAM starting at 0_H with four 1K PROMS (2708s) elsewhere in the map. The 4K RAM would be placed by connecting "output 1" to "$\overline{\text{USER RAM}}$" and E5 = 0. The PROMs would be placed by connecting "output 7" to "$\overline{\text{USER ROM}}$" and selecting jumper combination 0_H. The PROMs have been located in the 4K space between $C000_H$ and $CFFF_H$. Referring to Figure 8, column 0_H shows four contiguous sockets (1, 2, 3, and 4) starting at the 16K boundary. The low address PROM would be placed in socket 1, the next higher address PROM in socket 2, etc.

2) Assume it is desired to place 8K of ROM (4 MK 34000) at 0_H address and have 16K RAM on board. The 16K RAM would be placed starting at 4000_H by wire-oring "outputs 5 & 6" together (to create an 8K space) and selecting jumper combination 1_H. As in the previous case the low address ROM would be placed in socket 1 while the high order ROM would be placed in socket 4.

3) Assume it is desired to have 16K of RAM starting at address 0_H while also supporting 16K of ROM (for example, four 4K parts containing development system firmware). The RAM would be placed by "output 2" connected to "$\overline{\text{USER RAM}}$" and E5 = 0. The ROM would be placed by Wire-Oring "outputs 5, 6, 7, 9" together, connecting to "$\overline{\text{USER ROM}}$" (to create a 16K space between $A000_H$ & $E000_H$ on Figure 7), and selecting jumper combination 2_H (Figure 8). Jumper combination 2_H is the third column from the left in Figure 8 showing the full 16K decoded space.

The low address ROM is <u>not</u> placed in socket 1 as in the previous examples. The problem comes about in the crossing of 16K boundaries. In decoding the four 4K blocks ($A000_H$ to $E000_H$) the map of Figure 8 must alternate between areas III & IV (depending upon the ROM selected) shown in Figure 7.

When Figure 8 represents area III only the top half of Figure 8 is decoded while for area IV only the bottom of Figure 8 is decoded. Therefore starting at the low address in III the socket that is decoded is socket 3. The next higher address corresponds to socket 4. Going across the 16K boundary the low address memory space ($C000_H$ to $D000_H$) corresponds to the bottom of Figure 8 and to socket 1 while the next higher address ($D000_H$ to $E000_H$) will be decoded in socket 2.

Thus the order for inserting the ROMs into sockets would be:

high address	$D000_H$ through $DFFF_H$	Socket 2
	$C000_H$ through $CFFF_H$	Socket 1
	$B000_H$ through $BFFF_H$	Socket 4
low address	$A000_H$ through $AFFF_H$	Socket 3

This condition of the low order address ROM not belonging in Socket 1 can occur in one of two situations:

i) When the memory space being decoded crosses a 16K boundary (as in this example).

ii) If the PROM U32 has been coded such that Socket 1 is not always the lowest address and Socket 4 is not always the highest address as shown in Figure 8.

4) One last example should be sufficient to illustrate this boards mapping technique. Assume one has an OEM application needing only 256 bytes of RAM, three 2K ROMs and two 1K PROMs. The scratch RAM would be sufficient for this application so, "output 3" would be connected to "OS RAM" (jumper E28 to E29). A 2K ROM would be located in Socket 5 (the O. S. ROM Socket). To enable this socket "output 4" would be connected to " O. S. ROM" (jumper E30 to E31). To place the other ROM/PROM jumper point E5 is strapped to E4. Outputs "5" and "6" are connected to "USER ROM: by jumpering E59 to E60 and E57 to E58. This selects the 8K block of memory 000_H to 2000_H that starts at 000_H. To further divide the 8K space into smaller increments one refers to the detailed map of Figure 8. With the jumpers set to D_H the four

USER ROM/PROM Sockets will be selected. The low address ROMs will be placed in sockets 1 and 2. The high address PROMs will go in Socket 3 and 4 while the final 2K ROM, highest address of all, will reside in Socket 5.

IV. PROGRAMMING OF THE BIPOLAR PROMS

If the situation occurs where it is necessary to place memory in areas not defined by Figures 7 & 8 then either one or both of the bipolar PROMs may need to be reprogrammed. One section of U32 (column F_H) has been left unprogrammed and is available to the user for specific ROM/PROM placement. These two parts have been socketed to make any changes as simple as possible.

1) U31

Figure 9 is the truth table representing the information stored in the 32 x 8 bipolar PROM as shipped from the factory. Addresses and outputs are arranged from MSB to LSB as defined in the device data sheet. Because the outputs are active low and because there are fewer active states of interest. The truth table consists, basically, of two of the memory maps of Figure 7; one stacked on top of the other as shown. The lower truth table is defined for E5 - 0, while the upper is for E5 = 1. Each grid line in the truth table represents a 4K memory block, the same as shown in Figure 7.

Because of the one-to-one correspondence between Figure 7 and 9 the truth table is extremely easy to program. A logic "0" in a specific 4K location in the truth table corresponds to decoding that same 4K space in the memory map. For Example:

TRUTH TABLE OF

MK 6260
(U31)

NOTES:

1. Outputs are active low
2. A logic "1" is implied wherever there is no logic "0".
3. Each grid line represents 4K of Z80 memory space.

J	A_{15}	A_{14}	A_{13}	A_{12}	Address (Hex)		Output (Hex)		9	7	6	5	4	3	2	1	
1	1	1	1	1	1	F	F	B	[2]1	1	1	1	1	0	1	1	65K with E5=1
1	1	1	1	0	1	E	E	7					0	1			
1	1	1	0	1	1	D	F	F									
1	1	1	0	0	1	C	F	F									
1	1	0	1	1	1	B	F	F									
1	1	0	1	0	1	A	F	F									
1	1	0	0	1	1	9	F	F									
1	1	0	0	0	1	8	F	F									
1	0	1	1	1	1	7	F	D							0		
1	0	1	1	0	1	6	F	D							0		
1	0	1	0	1	1	5	F	D							0		
1	0	1	0	0	1	4	F	C							0	0	
1	0	0	1	1	1	3	7	F	0								
1	0	0	1	0	1	2	B	F		0							
1	0	0	0	1	1	1	D	F			0						
1	0	0	0	0	1	0	E	F				0					
0	1	1	1	1	0	F	F	B						0			65K with E5=0
0	1	1	1	0	0	E	F	7					0				
0	1	1	0	1	0	D	7	F	0								
0	1	1	0	0	0	C	B	F		0							
0	1	0	1	1	0	B	D	F			0						
0	1	0	1	0	0	A	F	E				0					
0	1	0	0	1	0	9	F	F									
0	1	0	0	0	0	8	F	F									
0	0	1	1	1	0	7	F	F									
0	0	1	1	0	0	6	F	F									
0	0	1	0	1	0	5	F	F									
0	0	1	0	0	0	4	F	F									
0	0	0	1	1	0	3	F	D							0		
0	0	0	1	0	0	2	F	D							0		
0	0	0	0	1	0	1	F	D							0		
0	0	0	0	0	0	0	F	C							0	0	
									9	7	6	5	4	3	2	1	
													O.S. ROM	O.S. RAM	16K RAM	4K RAM	
									User ROM/PROM				O.S.		User RAM		

4

Figure 9

i) A "0" placed in the lower right hand corner of the truth table (address = 0000, E5 = 0, output 1) decodes the 4K space indicated by the arrow in the lower right hand corner of Figure 7.

ii) To decode the memory space between 3000_H and 4000_H on output 9, and for E5 = 1, a "0" is placed in the top truth table (E5 = 1), in the column labeled "output 9", and at location 0011 (A_{15}, A_{14}, A_{13}, A_{12}). (Remember that the first bit of the address shown represents the state of E5).

Multiple decoding as shown in the two top 4K blocks of Figure 7, is achieved by placing a "0" in the same location in the top and bottom truth tables of Figure 9.

2) U32

Figure 10 is the truth table representing the information stored in the 256 x 4 bipolar PROM as shipped from the factory. There are eight input (address) lines to this PROM: four jumper lines (MS byte) and four bus address lines (LS byte). The four output lines are (MSB listed 1st) $\overline{\text{ROM 4}}$ - $\overline{\text{ROM 1}}$. This input/output information is shown in the truth table.

Because the outputs are active low and because there are fewer active states the truth table is programmed with logic "0s" since these are the states of interest. Figure 10 is basically laid out as a representation of the 16 columns, listed as 0_H through F_H (the Hex jumper address) shown in Figure 8. The first three column numbers are listed in Figure 10 for reference. Each column number, of course, is the same as the jumper address (in Hex). Each grid line in the truth table represents a 1K memory segment, the same as in Figure 8.

TRUTH TABLE FOR 256 x 4 BIPOLAR PROM (U32) 1K SEGMENT DECODE

Jumpers (HEX)	Address (HEX)	Output (HEX)	$\overline{ROM\ 4}$	$\overline{ROM\ 3}$	$\overline{ROM\ 2}$	$\overline{ROM\ 1}$
1	F	F	1	1	1	1
1	E	F				
1	D	F				
1	C	F	Column "1"			
1	B	F				
1	A	F				
	9	F				
1	8	F				
1	7	7	0			
1	6	7	0			
1	5	B		0		
1	4	B		0		
1	3	D			0	
1	2	D			0	
1	1	E				0
1	0	E				0
0	F	F				
0	E	F				
0	D	F				
0	C	F				
0	B	F				
0	A	F				
0	9	F				
0	8	F	Column "0"			
0	7	F				
0	6	F				
0	5	F				
0	4	F				
0	3	7	0			
0	2	B		0		
0	1	D			0	
0	0	E				0

Jumpers (HEX)	Address (HEX)	Output (HEX)	$\overline{ROM\ 4}$	$\overline{ROM\ 3}$	$\overline{ROM\ 2}$	$\overline{ROM\ 1}$
3	F	F				
3	E	F				
3	D	F				
3	C	F				
3	B	F				
3	A	F				
3	9	F				
3	8	F				
3	7	7	0			
3	6	7	0			
3	5	F				
3	4	F				
3	3	F				
3	2	B		0		
3	1	D			0	
3	0	E				0
2	C	7	0			
2	E	7	0			
2	D	7	0			
2	C	7	0			
2	B	B		0		
2	A	B		0		
2	9	B		0		
2	8	B		0		
2	7	D			0	
2	6	D			0	
2	5	D			0	
2	4	D			0	
2	3	E				0
2	2	E	Column "2"			0
2	1	E				0
2	0	E				0

Jumpers (HEX)	Address (HEX)	Output (HEX)	$\overline{ROM\ 4}$	$\overline{ROM\ 3}$	$\overline{ROM\ 2}$	$\overline{ROM\ 1}$
5	F	F				
5	E	F				
5	D	F				
5	C	F				
5	B	F				
5	A	F				
5	9	F				
5	8	F				
5	7	7	0			
5	6	7	0			
5	5	B		0		
5	4	B		0		
5	3	D			0	
5	2	D			0	
5	1	F				
5	0	E				0
4	F	F				
4	E	F				
4	D	F				
4	C	F				
4	B	F				
4	A	F				
4	9	F				
4	8	F				
4	7	7	0			
4	6	7	0			
4	5	B		0		
4	4	B		0		
4	3	F				
4	2	F				
4	1	D			0	
4	0	E				0

Jumpers (HEX)	Address (HEX)	Output (HEX)	$\overline{ROM\ 4}$	$\overline{ROM\ 3}$	$\overline{ROM\ 2}$	$\overline{ROM\ 1}$
7	F	F				
7	E	F				
7	D	F				
7	C	F				
7	B	7	0			
7	A	7	0			
7	9	7	0			
7	8	7	0			
7	7	B		0		
7	6	B		0		
7	5	B		0		
7	4	B		0		
7	3	F				
7	2	F				
7	1	D			0	
7	0	E				0
6	F	F				
6	E	F				
6	D	F				
6	C	F				
6	B	F				
6	A	F				
6	9	F				
6	8	F				
6	7	7	0			
6	6	7	0			
6	5	7	0			
6	4	7	0			
6	3	F				
6	2	B		0		
6	1	D			0	
6	0	E				0

NOTES:

1. Outputs are active Low
2. A logic "1" is implied whenever there is no logic "0"
3. Each grid line represents 1K of Z80 memory space

4

Figure 10

Jumpers (HEX)	Address (HEX)	Output (HEX)		$\overline{\text{ROM 4}}$	$\overline{\text{ROM 3}}$	$\overline{\text{ROM 2}}$	$\overline{\text{ROM 1}}$	Jumpers (HEX)	Address (HEX)	Output (HEX)		$\overline{\text{ROM 4}}$	$\overline{\text{ROM 3}}$	$\overline{\text{ROM 2}}$	$\overline{\text{ROM 1}}$	Jumpers (HEX)	Address (HEX)	Output (HEX)		$\overline{\text{ROM 4}}$	$\overline{\text{ROM 3}}$	$\overline{\text{ROM 2}}$	$\overline{\text{ROM 1}}$	Jumpers (HEX)	Address (HEX)	Output (HEX)		$\overline{\text{ROM 4}}$	$\overline{\text{ROM 3}}$	$\overline{\text{ROM 2}}$	$\overline{\text{ROM 1}}$
9	F	F						B	F	F						D	F	F						F	F						
9	E	F						B	E	F						D	E	F						F	E						
9	D	F						B	D	F						D	D	F						F	D						
9	C	F						B	C	F						D	C	F						F	C						
9	B	7		0				B	B	F						D	B	F						F	B						
9	A	7		0				B	A	F						D	A	F						F	A						
9	9	7		0				B	9	F						D	9	F						F	9						
9	8	7		0				B	8	F						D	8	F						F	8			Unprogrammed Section			
9	7	B			0			B	7	F						D	7	F						F	7						
9	6	B			0			B	6	F						D	6	F						F	6						
9	5	D				0		B	5	F						D	5	7		0				F	5						
9	4	D				0		B	4	7		0				D	4	B			0			F	4						
9	3	F						B	3	B			0			D	3	D				0		F	3						
9	2	F						B	2	D				0		D	2	D				0		F	2						
9	1	F						B	1	E					0	D	1	E					0	F	1						
9	0						0	B	0	E					0	D	0	E					0	F	0						
8	F	7		0				A	F	F						C	F	F						E	F	F					
8	E	7		0				A	E	F						C	E	F						E	E	F					
8	D	7		0				A	D	F						C	D	F						E	D	F					
8	C	7		0				A	C	F						C	C	F						E	C	F					
8	B	B			0			A	B	7		0				C	B	F						E	8	F					
8	A	B			0			A	A	7		0				C	A	F						E	A	F					
8	9	B			0			A	9	7		0				C	9	F						E	9	7		0			
8	8	B			0			A	8	7		0				C	8	F						E	8	B			0		
8	7	D				0		A	7	B			0			C	7	F						E	7	D				0	
8	6	D				0		A	6	B			0			C	6	7		0				E	6	D				0	
8	5	D				0		A	5	B			0			C	5	B			0			E	5	D				0	
8	4	D				0		A	4	B			0			C	4	D				0		E	4	D				0	
8	3	F						A	3	D				0		C	3	E					0	E	3	E					0
8	2	F						A	2	D				0		C	2	E					0	E	2	E					0
8	1	F						A	1	F						C	1	E					0	E	1	E					0
8	0	E					0	A	0	E					0	C	0	E					0	E	0	E					0

Figure 10 (Con't)

Because of the close correspondence between Figures 8 and 10, the truth table is extremely easy to program. For Example:

1) A "0" placed in the lower right hand corner of column 6 decodes the 1K space indicated by the arrow and "1" (for Socket 1) at the bottom of the column 6 of Figure 8.

2) To decode four contiguous 1K spaces (1K per socket) starting at the lowest address the following "0s" are placed in column 0 of the truth table:

1) 0 in location 00_H under $\overline{\text{ROM 1}}$ - decodes socket 1
2) 0 in location 01_H under $\overline{\text{ROM 2}}$ - decodes socket 2
3) 0 in location 02_H under $\overline{\text{ROM 3}}$ - decodes socket 3
4) 0 in location 03_H under $\overline{\text{ROM 4}}$ - decodes socket 4

For programming purposes it is usually necessary to supply address and output data in hexidecimal format. Both Figures 9 and 10 have columns for this data. The two digit hex address always appears to the left of the 1-2 digit hex data column. In the truth tables once "0s" are filled in to define each decoded block then "1s" may be assumed to fill in all the rest of the truth table (as demonstrated in the top most row). The information is then extracted from each row (binary word) converted to hex and placed in the column labeled output. This hex information is then used to program a "blank" PROM or is supplied to a distributor who has programming capability.

V) REDUNDANT DECODING OF O. S. RAM

To reduce the number of logic gates and decoding on board the scratchpad RAM has not been uniquely decoded. This 256 x 8 RAM is repetitively decoded eight consecutive times in the last (top) 2K of the memory map. Addressing any 256 address sector in this space will address and access the RAM. The mapping of the top 8K of the memory map is done in Figure 11 showing the redundantly decoded portion. See also Figure 6.

No other part on the board may use this portion of memory space. Otherwise there would be a conflict. This space is, however, available to memory not located on the board. A peripheral memory may be decoded into this space if $\overline{\text{MEMDISB}}$ is used. This function disables the on board parts (along with the redundant decoded space) during access such that there is no memory conflict.

REDUNDANT DECODING OF SCRATCHPAD RAM

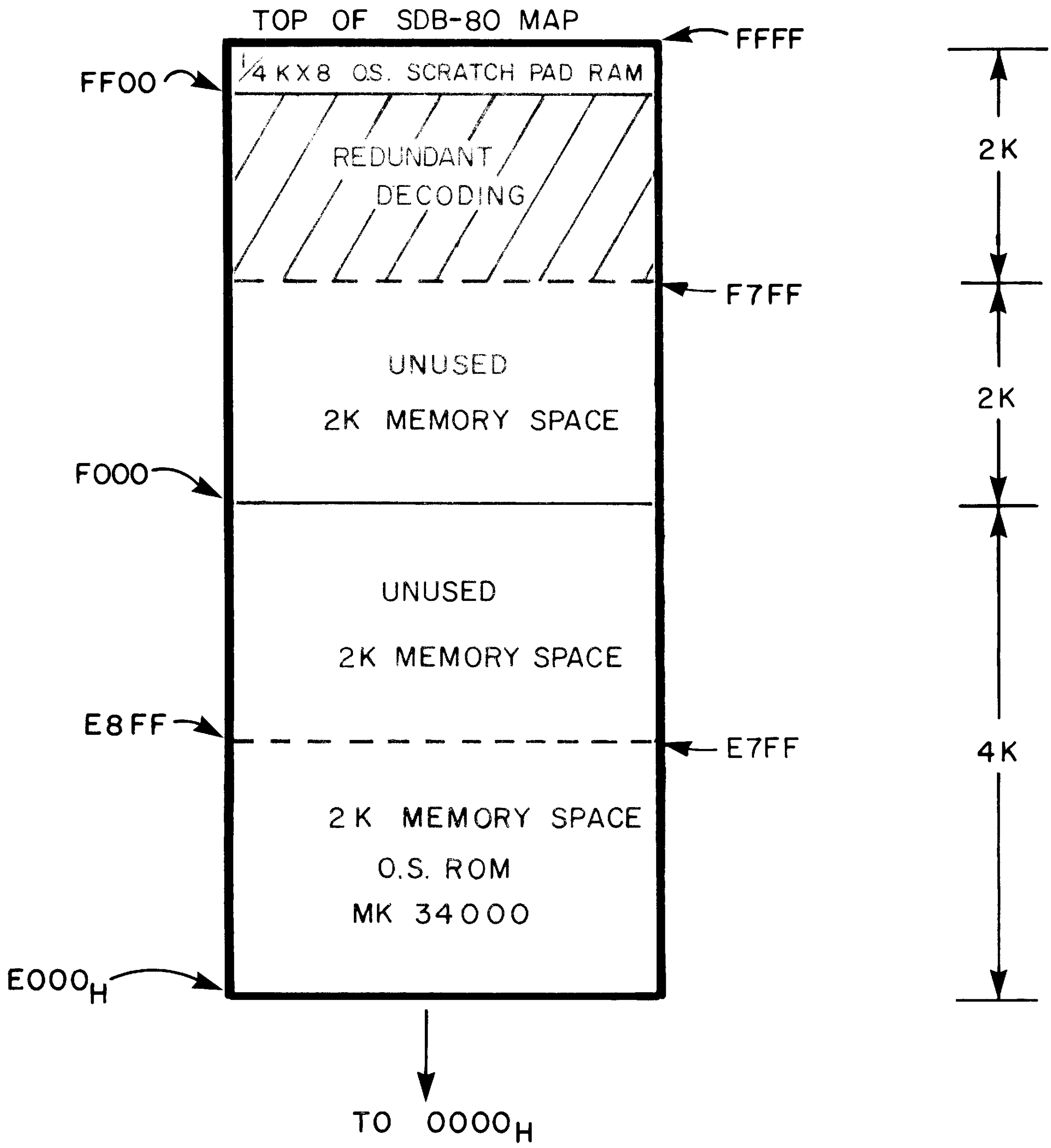


Figure 11

VI. JUMPER OPTIONS

There are four groups of wire wrap pins that control the memory options discussed in this section. The location of these jumpers on the SDB-80 is shown in Figure 12.

Briefly, the jumpers control the following:

Group 1: E1, E2, E3 & E22, E23, E24

Modify the address lines for either 4K or 16K dynamic RAM.

Group 2: E7 through E20 & E40 through E54

Adjust the pin out of the 5 ROM/PROM Sockets so they can accept any one of five different memory parts.

Group 3: E4, E5, E6, & E25 through E31

Address selection and chip enable function for O. S. RAM & ROM and the USER RAM.

Group 4: E32, through E39 & E55 through E62 memory size and location of each USER ROM Socket.

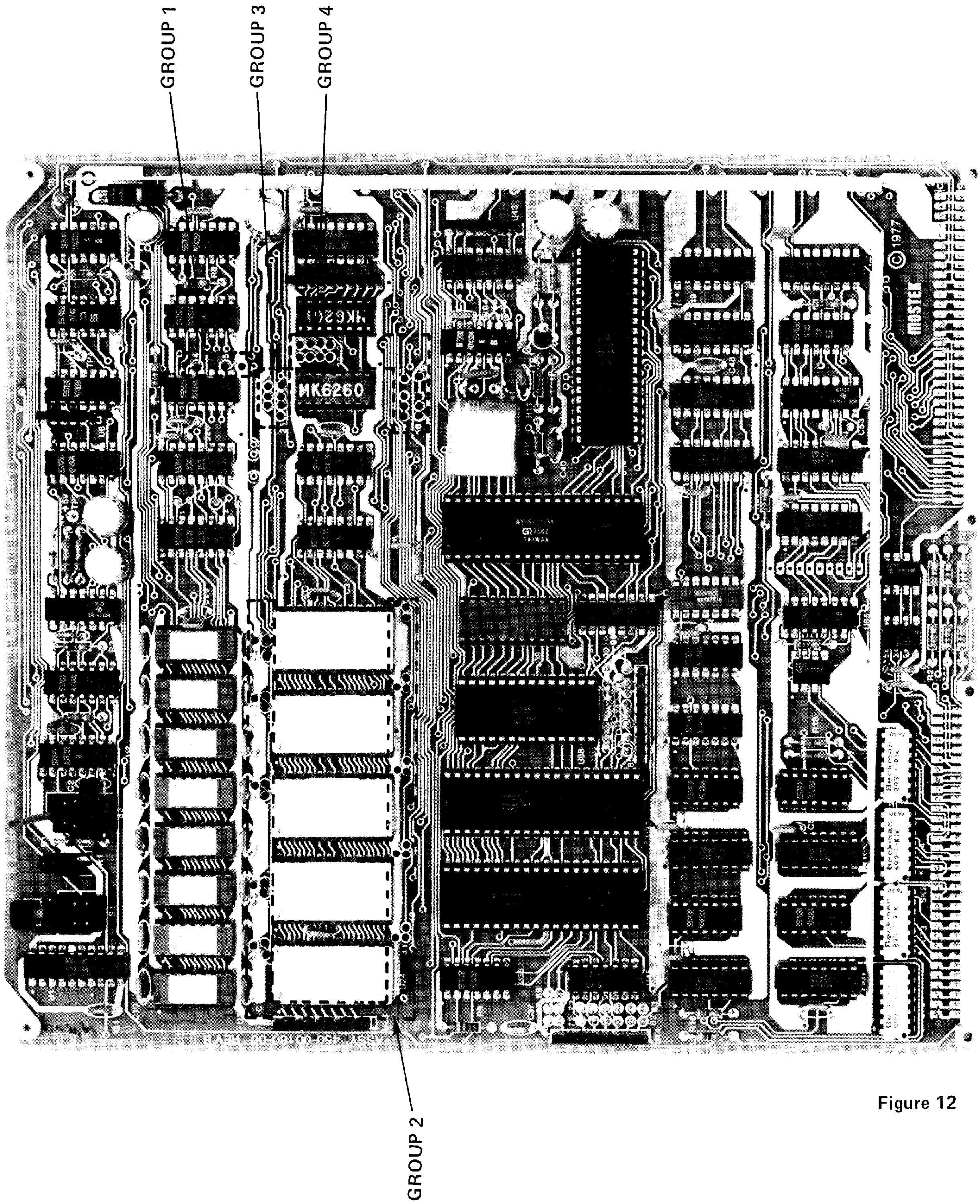


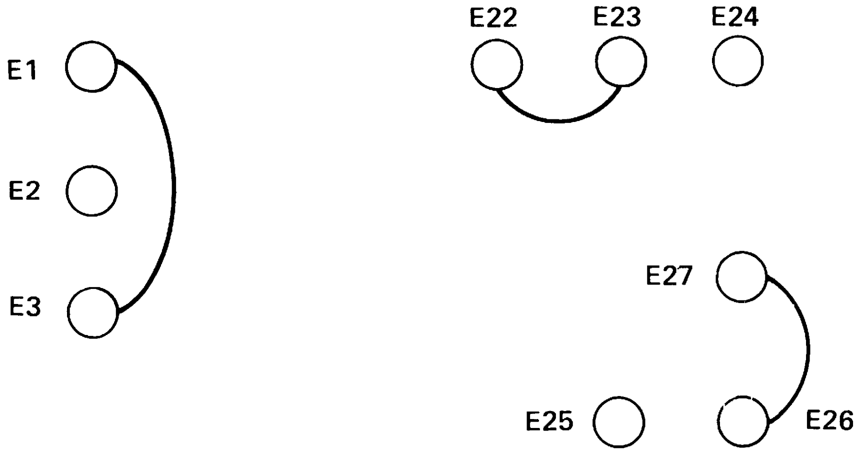
Figure 12

In more detail, Group 1 jumpers are used to rearrange the address lines to the eight 16 pin RAM Sockets. This allows either 4K or 16K parts to be placed in these sockets. E1, E2, E3 either grounds $\overline{CS}$ on the 4K RAM (part is always selected) or connects the address line from the multiplexer U19 to the appropriate address pin on 16K RAM. The proper address (A_6 or A_{12}) into the multiplexer for either 4K or 16K RAM is determined by E22, E23, E24.

Figure 13 shows the two wiring options for this set of pins. These will be wired at the factory in correspondance with the USER RAM memory size.

4K/16K WIRE WRAP PIN OPTIONS

4K DYNAMIC RAM OPTION



16K DYNAMIC RAM OPTION

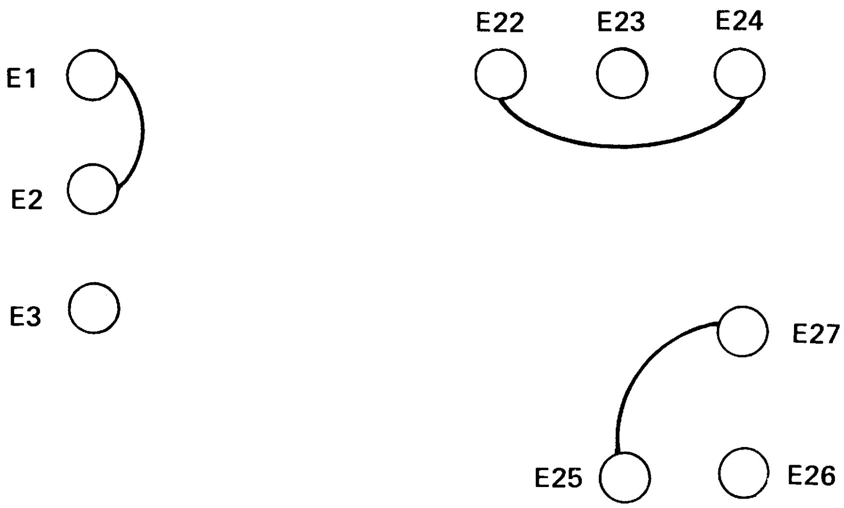


Figure 13

2) To make the 24 pin ROM/PROM Sockets universal to the various memory parts that can be used on this board, two address lines (A_{10} - A_{11}) and two supplies (GND and + 12V) must be jumpered. Figure 14 shows the various jumper connections required to configure each socket.

Socket 1 - U24
2 - U25
3 - U26
4 - U27
5 - U28

Figure 14 is divided into 2 columns; the left represents jumpers for all 1K parts, the right shows 2K and 4K options. Figure 15 details the MOS memory chip select options used. The option for each socket (1K or 2-4K) is shown independantly because a mix of memory parts may be required throughout the five sockets.

Mostek ROMs have options on their chip select inputs that must be comprehended in conjunction with the jumpers. As an example the MK 34000 has three chip select lines: Pins 18, 20, and 21. These lines must be programmed (at the time the pattern is stored) to be either active high (CS), active low ($\overline{CS}$), or open circuited (NC). ROM Sockets 1-4 (U24-U27) can be enabled in increments of less than 4K bytes, i.e., 1K or 2K. Socket # 5 (U28) however, is enabled only in 4K blocks. Because of this, the MOS chip select programming requirement for these two sets of sockets is different. If 1K ROMs are used in Sockets 1-4 and each socket is enabled for only 1K bytes of memory there will be no redundant decoding no matter how the extra chip selects have been programmed. If however, a 1K ROM is placed

SDB-80 ROM / PROM SOCKETS #1 — #5 JUMPER CONNECTIONS.

THIS ILLUSTRATES THE WIRE WRAP JUMPER CONNECTIONS FOR THE FOLLOWING:

ALL 1K PARTS	2K AND 4K OPTIONS
MK 2708 1K MOS PROM	MK 34000 2K MOS ROM
MK 30000 1K MOS RAM	MK 32000 4K MOS ROM
8252708 1K BIPOLAR PROM	

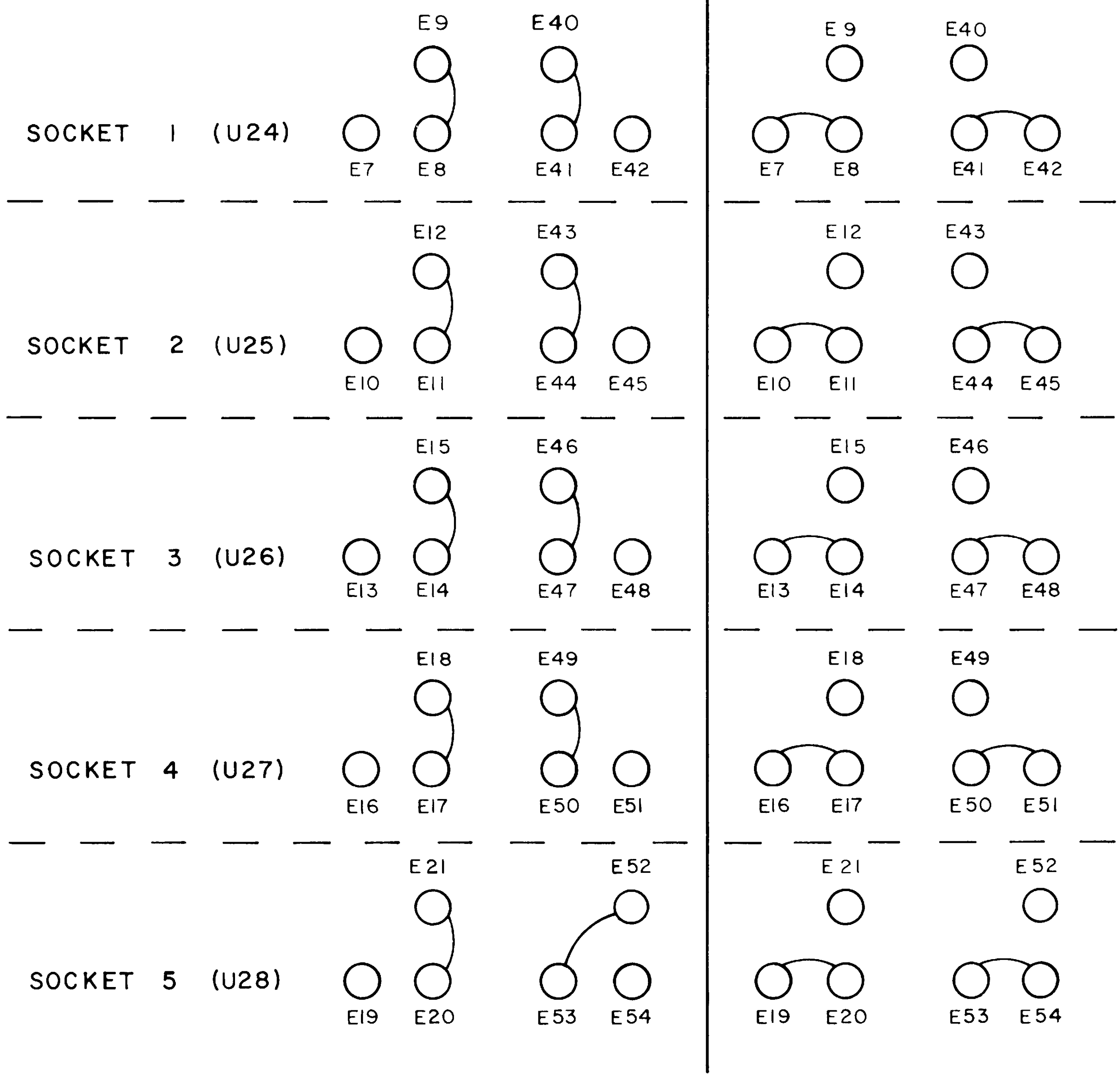


Figure 14

in Socket 5, there exists the potential for three redundantly decoded 1K spaces in addition to the desired 1K ROM space. The same is true for Sockets 1-4 if they have been enabled for more than 1K space.

If the redundant space does not conflict with any other memory space, then multiple space decoding poses no problem and the chip select functions are really a don't care situation. If the redundant memory space conflicts with some actual memory then the "extra" chip selects can be used as address inputs to reduce or eliminate the amount of redundant space.

Several examples will demonstrate how the jumpers are wired for various memory combinations.

a) Assuming the following complement of parts, what jumpers are required?

Socket 1 -	MK 34000	2K	E7-E8, E41-E42
Socket 2 -	MK 34000	2K	E10-E11, E44-E45
Socket 3 -	MK 2708	1K	E14-E15, E46-E47
Socket 4 -	MK 32000	4K	E16-E17, E50-E51
Socket 5 -	MK 30000	1K	E20-E21, E52-E53

Figure 16 (a) shows these jumper connections.

b) Assuming the following complement of parts, what jumpers are required?

Socket 1 -	MK 2708	1K	E8-E9, E40-E41
Socket 2 -	MK 2708	1K	E11-E12, E43-E44

		18	19	20	21
MK2708		–	+12V	$\overline{CS}$	–5V
82S2708		NC	NC	$\overline{CE}$1	NC*
MK30000	1K	CS2/$\overline{CS2}$	NC	CS1/$\overline{CS1}$	NC
MK34000	2K	CS2/$\overline{CS2}$/NC	A10	CS1/$\overline{CS1}$/NC	CS3/$\overline{CS3}$/NC
MK32000	4K	A11	A10	$\overline{CE}$	CS/CS/NC

Extra chip select pin. This pin is programmed to reduce redundant decoding. Programming is either active high or active low depending upon placement of the memory within the map.

+12V for PROM or A10 for ROMs.

Pin 20 is the socket enable pin driven from the decode logic. This pin must always be programmed active low, ie, $\overline{CS1}$ or $\overline{CE}$.

Pin 21 (sockets 1-5) on the SDB-80 is hard wired to –5V for the MK2708 PROM. Memory parts using those sockets must be programmed for no connection unless they require –5V on pin 21.

*NC = No connection within package

The operating system ROM has the following program options:

Pins 18–CS2, 20–$\overline{CS1}$, 21–NC

(c)

Figure 15

Socket 3 - MK 2708	1K	E14-E15, E46-E47
Socket 4 - -		
Socket 5 - MK 32000	4K	E19-E20, E53-E54

Figure 16 (b) shows these jumper connections.

(3) Figure 17 illustrates the jumper connections that determine whether 4K or 16K of RAM space is reserved for the USER RAM, and whether or not the O. S. RAM and/or ROM is enabled.

(4) Figure 18 shows the connections that determine the memory space decoded for the four User ROM/PROM Sockets and the particular allocation of this space to each socket.

Figure 18 is divided into two parts; the left hand column labeled "4K"BLOCKS DECODED: and the right hand two columns labeled "JUMPERS"

The left hand column determines how many 4K spaces, and their location within the memory map, will be decoded for the USER ROM Sockets. This is clearly shown in Figure 7. The right hand columns show the 16 possible coding combinations (JUMPERS) illustrated in Figure 8.

Several examples will demonstrate how jumpers are used for various memory conditions.

a) Assuming the following complement of parts, what jumpers are required?

Socket 1-MK 32000	4K
Socket 2-MK 32000	4K
Socket 3-MK 32000	4K
Socket 4-MK 32000	4K

Four 4K spaces require jumpers:

E59-E60
E58-E59
E55-E56
E61-E62

Four 4K sockets are shown in Figure 8 at jumper address 2_H which are pins:

E32-E33
E34-E35
E38-E39

b) Assuming the following complement of parts, what jumpers are required?

Socket 1 - MK 34000	2K
Socket 2 - MK 2708	1K
Socket 3 - MK 2708	1K
Socket 4 - MK 2708	1K

Two 4K spaces require jumpers:

E59-E60
E57-E58

One 2K socket and three 1K sockets are shown in Figure 8 at jumper address B_H which are jumpers.

E32-E33

JUMPER CONNECTION EXAMPLES

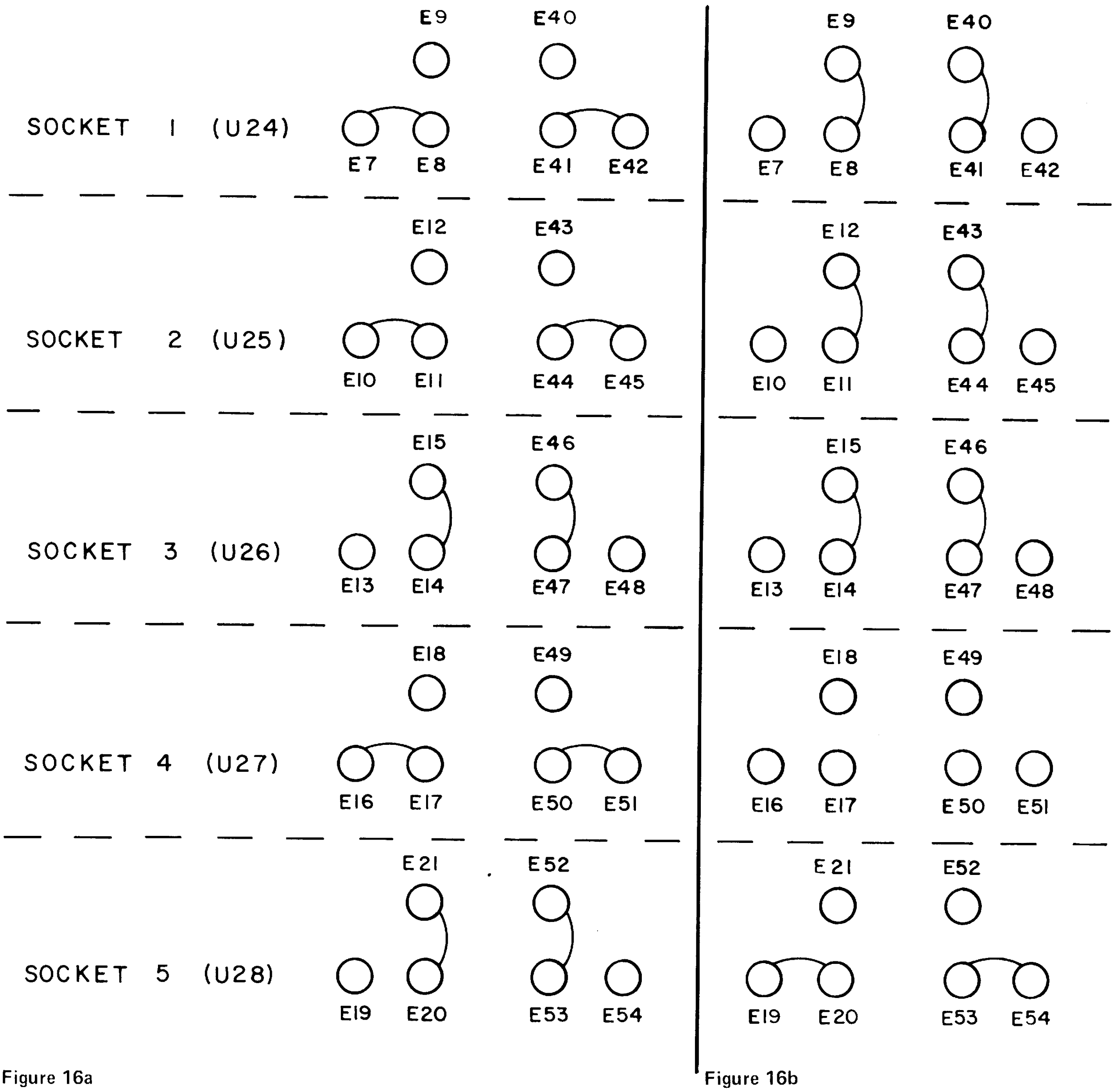


Figure 16a

Figure 16b

USER RAM, OS RAM/ROM CONNECTIONS

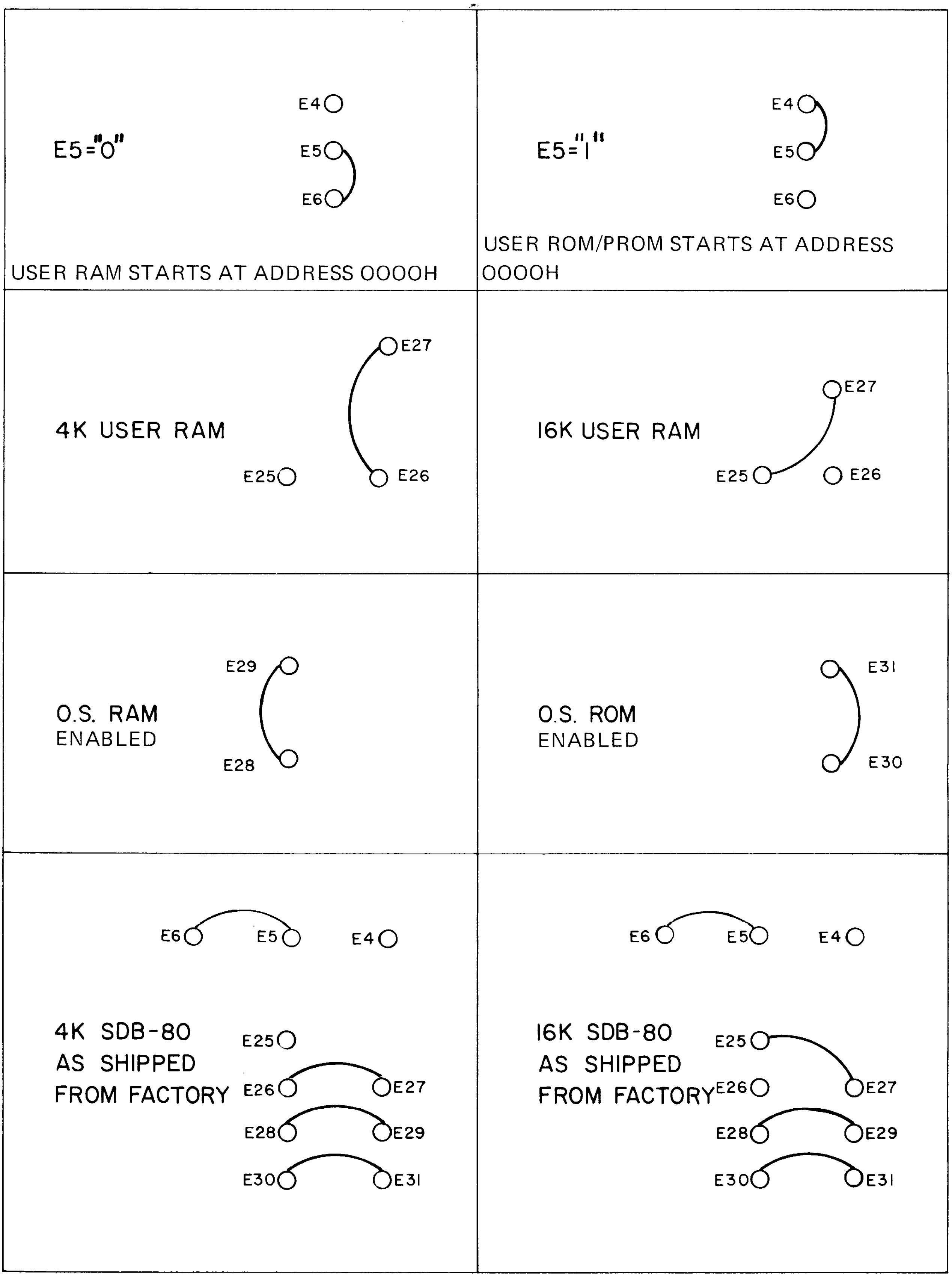
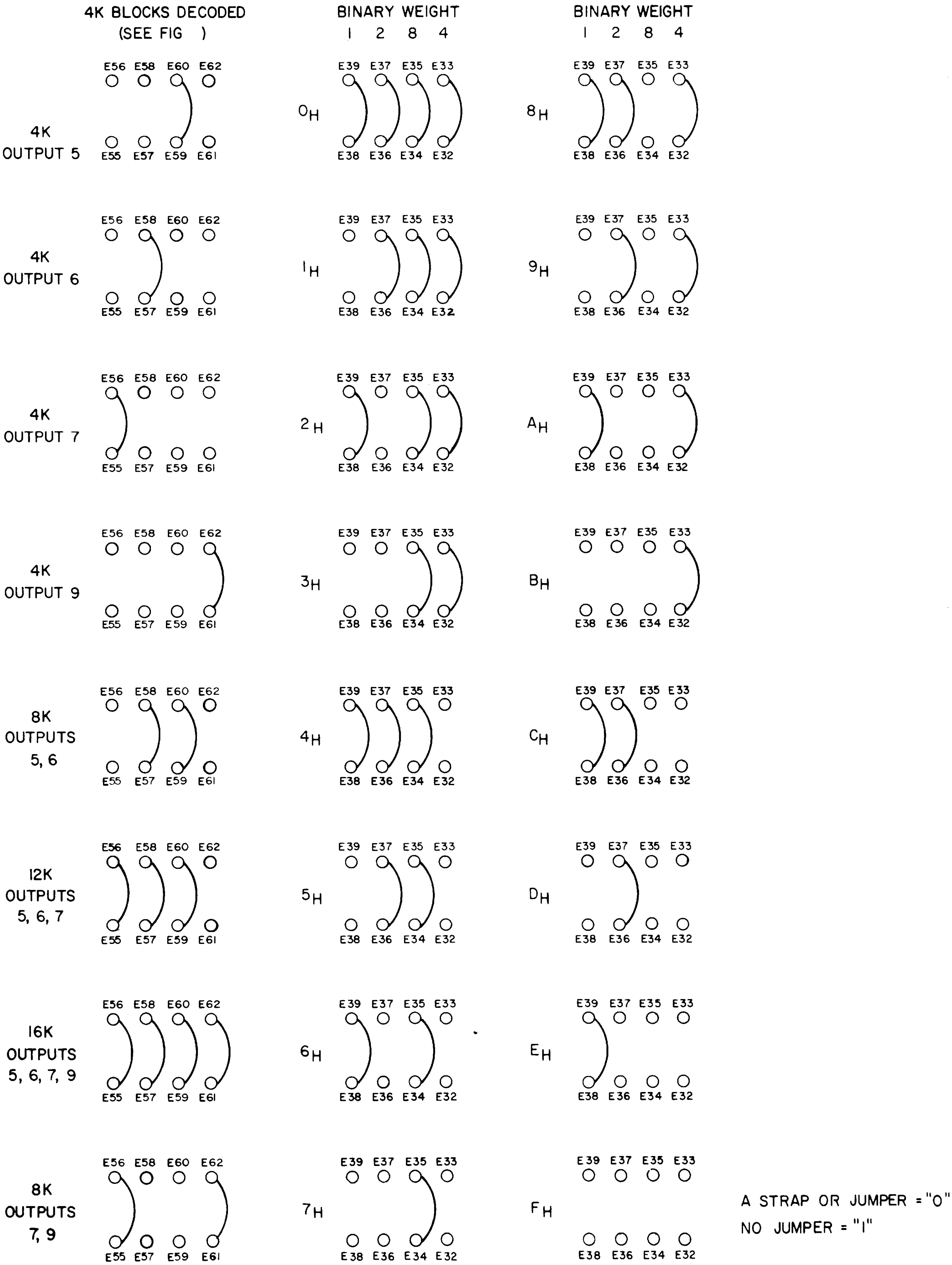


Figure 17



4K BLOCKS DECODED
(SEE FIG)
BINARY WEIGHT
1 2 8 4
BINARY WEIGHT
1 2 8 4
E56 E58 E60 E62
E55 E57 E59 E61
E39 E37 E35 E33
E38 E36 E34 E32
4K OUTPUT 5
4K OUTPUT 6
4K OUTPUT 7
4K OUTPUT 9
8K OUTPUTS 5, 6
12K OUTPUTS 5, 6, 7
16K OUTPUTS 5, 6, 7, 9
8K OUTPUTS 7, 9
0_H
1_H
2_H
3_H
4_H
5_H
6_H
7_H
8_H
9_H
A_H
B_H
C_H
D_H
E_H
F_H
A STRAP OR JUMPER = "0"
NO JUMPER = "1"

Figure 18

VII. BLANK PROGRAMMING CHARTS

To facilitate any change or modification of the memory on the SDB-80, a set of blank memory maps and truth tables have been included. The first four figures correspond to Figures 7, 8, 9 and 10 respectively. The last five figures correspond to Figures 13, 14, 16, 17 and 18 and can be used for drawing jumper options for a new memory configuration.

SDB-80 MEMORY MAP

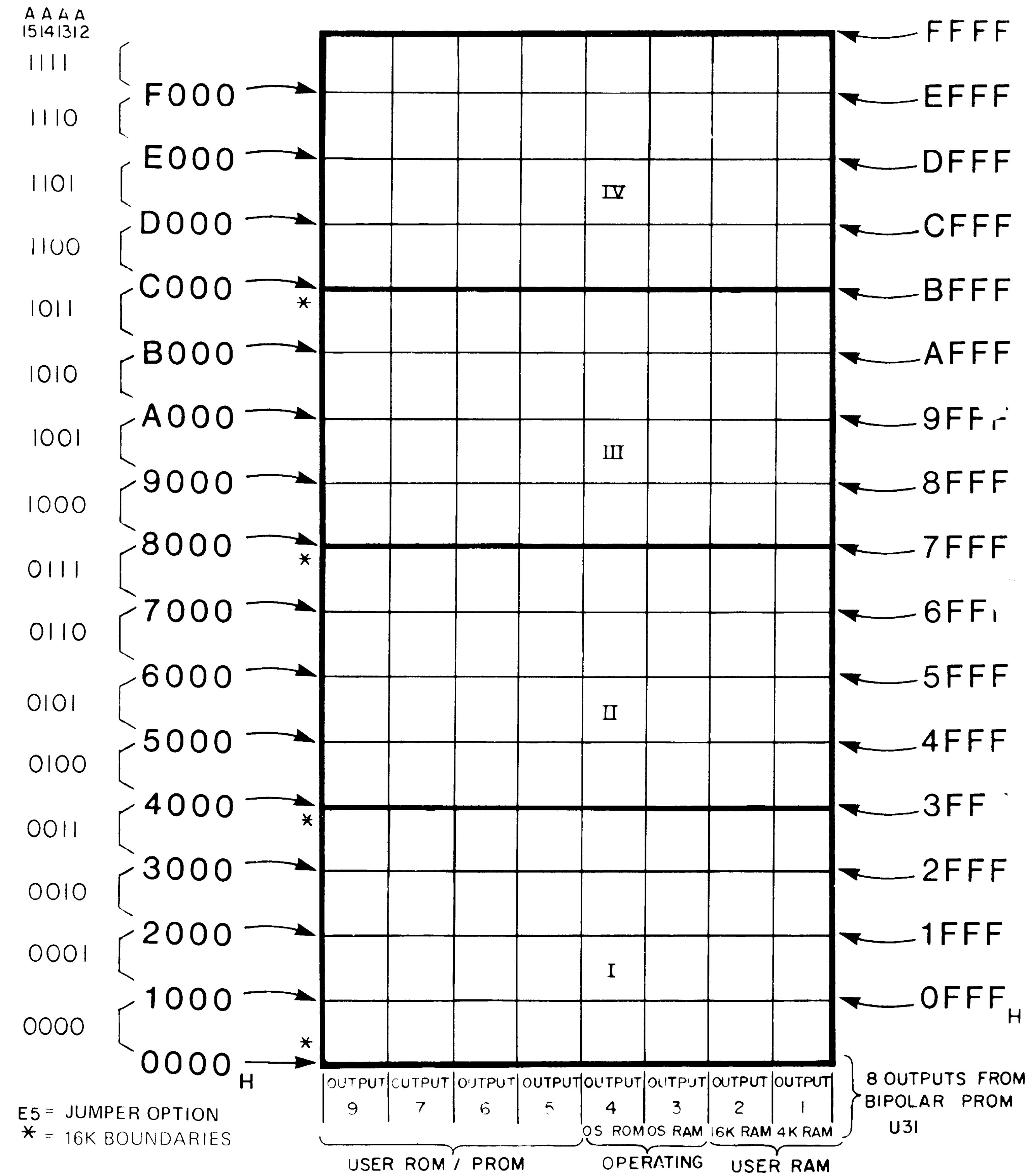


Figure 19

MEMORY MAP FOR 256 x 4 BIPOLAR PROM U32

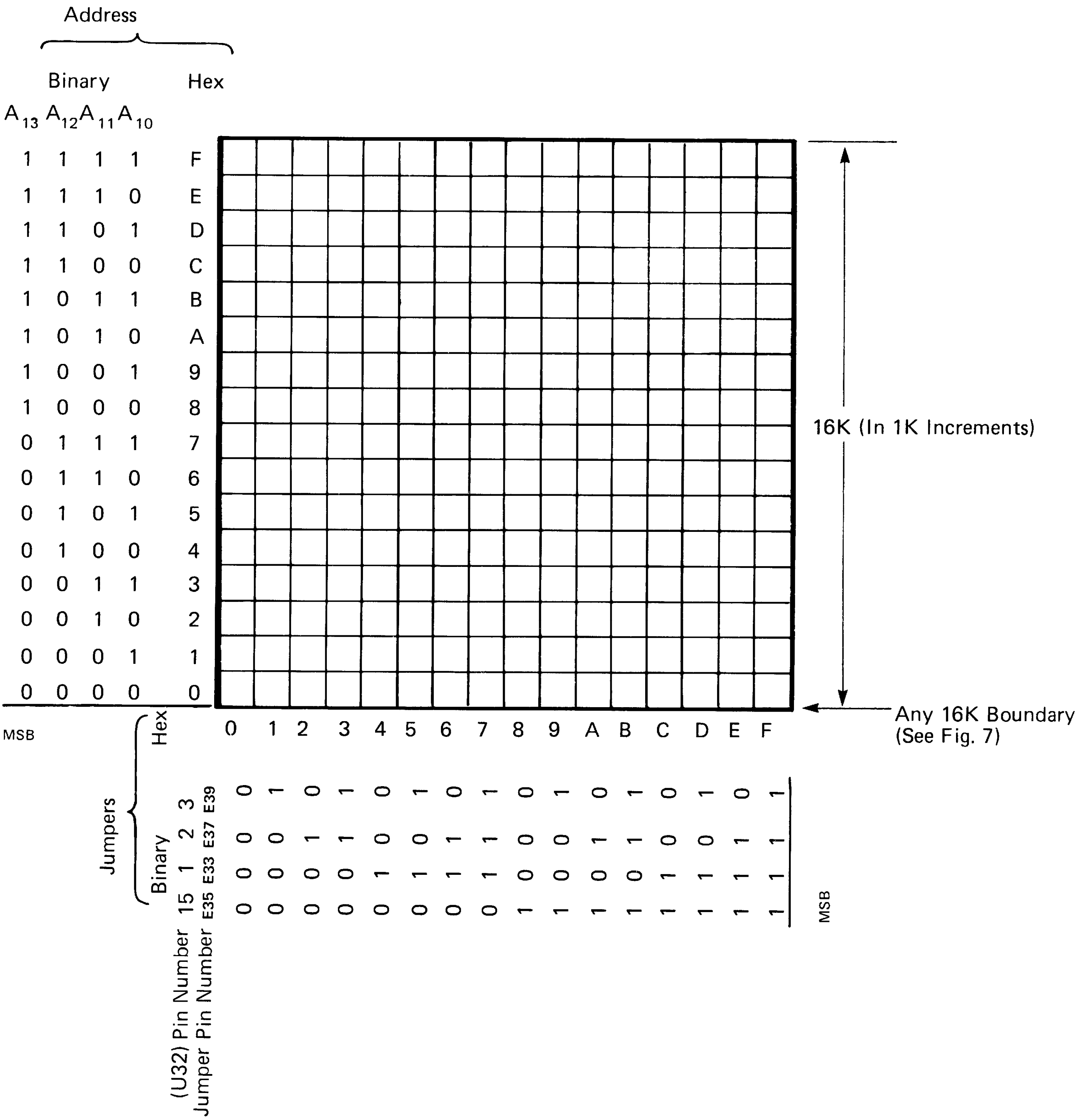


Figure 20

TRUTH TABLE FOR 32 x 8 BIPOLAR PROM U31

Address (Binary) J	A_{15}	A_{14}	A_{13}	A_{12}	Address (Hex)	Output (Hex)	Outputs (Binary) 9	7	6	5	4	3	2	1	
1	1	1	1	1	1F										
1	1	1	1	0	1E										
1	1	1	0	1	1D										
1	1	1	0	0	1C										
1	1	0	1	1	1B										
1	1	0	1	0	1A										
1	1	0	0	1	19										
1	1	0	0	0	18										65K with E5=1
1	0	1	1	1	17										
1	0	1	1	0	16										
1	0	1	0	1	15										
1	0	1	0	0	14										
1	0	0	1	1	13										
1	0	0	1	0	12										
1	0	0	0	1	11										
1	0	0	0	0	10										
0	1	1	1	1	0F										
0	1	1	1	0	0E										
0	1	1	0	1	0D										
0	1	1	0	0	0C										
0	1	0	1	1	0B										
0	1	0	1	0	0A										
0	1	0	0	1	09										
0	1	0	0	0	08										65K with E5=0
0	0	1	1	1	07										
0	0	1	1	0	06										
0	0	1	0	1	05										
0	0	1	0	0	04										
0	0	0	1	1	03										
0	0	0	1	0	02										
0	0	0	0	1	01										
0	0	0	0	0	00										

Outputs 4: O.S. ROM; 3: O.S. RAM; 2: 16K RAM; 1: 4K RAM

Outputs 9, 7, 6, 5: User ROM/PROM; Outputs 4, 3: O.S.; Outputs 2, 1: User RAM

Figure 21

TRUTH TABLE FOR 256 x 4 BIPOLAR PROM (U32) 1K SEGMENT DECODE

Jumpers (Hex)	Address (Hex)	Output (Hex)		$\overline{\text{ROM 4}}$	ROM 3	$\overline{\text{ROM 2}}$	$\overline{\text{ROM 1}}$	Jumpers (Hex)	Address (Hex)	Output (Hex)		$\overline{\text{ROM 4}}$	ROM 3	$\overline{\text{ROM 2}}$	$\overline{\text{ROM 1}}$	Jumpers (Hex)	Address (Hex)	Output (Hex)		$\overline{\text{ROM 4}}$	ROM 3	$\overline{\text{ROM 2}}$	$\overline{\text{ROM 1}}$	Jumpers (Hex)	Address (Hex)	Output (Hex)		$\overline{\text{ROM 4}}$	ROM 3	$\overline{\text{ROM 2}}$	$\overline{\text{ROM 1}}$
1	F							3	F							5	F							7	F						
1	E							3	E							5	E							7	E						
1	D							3	D							5	D							7	D						
1	C							3	C							5	C							7	C						
1	B							3	B							5	B							7	B						
1	A							3	A							5	A							7	A						
1	9							3	9							5	9							7	9						
1	8							3	8							5	8							7	8						
1	7							3	7							5	7							7	7						
1	6							3	6							5	6							7	6						
1	5							3	5							5	5							7	5						
1	4							3	4							5	4							7	4						
1	3							3	3							5	3							7	3						
1	2							3	2							5	2							7	2						
1	1							3	1							5	1							7	1						
1	0							3	0							5	0							7	0						
0	F							2	F							4	F							6	F						
0	E							2	E							4	E							6	E						
0	D							2	D							4	D							6	D						
0	C							2	C							4	C							6	C						
0	B							2	B							4	B							6	B						
0	A							2	A							4	A							6	A						
0	9							2	9							4	9							6	9						
0	8							2	8							4	8							6	8						
0	7							2	7							4	7							6	7						
0	6							2	6							4	6							6	6						
0	5							2	5							4	5							6	5						
0	4							2	4							4	4							6	4						
0	3							2	3							4	3							6	3						
0	2							2	2							4	2							6	2						
0	1							2	1							4	1							6	1						
0	0							2	0							4	0							6	0						

Figure 22

Jumpers (Hex)	Address (Hex)	Output (Hex)		$\overline{\text{ROM 4}}$	$\overline{\text{ROM 3}}$	$\overline{\text{ROM 2}}$	$\overline{\text{ROM 1}}$
9	F						
9	E						
9	D						
9	C						
9	B						
9	A						
9	9						
9	8						
9	7						
9	6						
9	5						
9	4						
9	3						
9	2						
9	1						
9	0						
8	F						
8	E						
8	D						
8	C						
8	B						
8	A						
8	9						
8	8						
8	7						
8	6						
8	5						
8	4						
8	3						
8	2						
8	1						
8	0						

Jumpers (Hex)	Address (Hex)	Output (Hex)		$\overline{\text{ROM 4}}$	$\overline{\text{ROM 3}}$	$\overline{\text{ROM 2}}$	$\overline{\text{ROM 1}}$
B	F						
B	E						
B	D						
B	C						
B	B						
B	A						
B	9						
B	8						
B	7						
B	6						
B	5						
B	4						
B	3						
B	2						
B	1						
B	0						
A	F						
A	E						
A	D						
A	C						
A	B						
A	A						
A	9						
A	8						
A	7						
A	6						
A	5						
A	4						
A	3						
A	2						
A	1						
A	0						

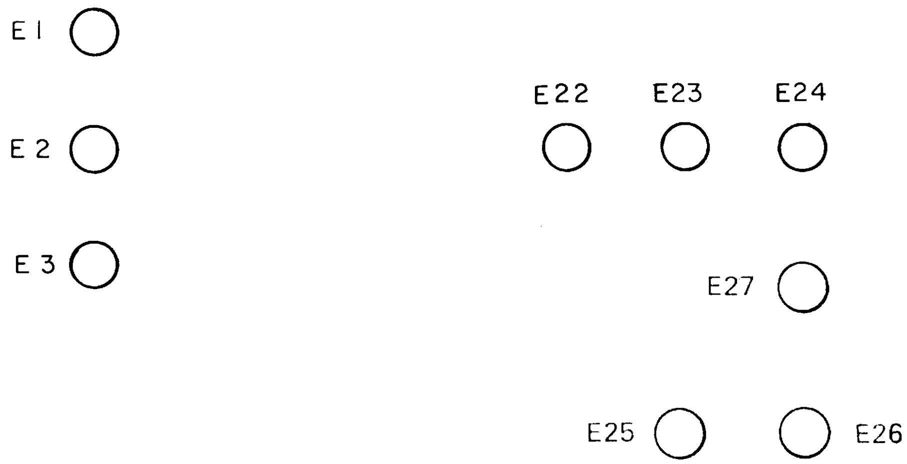
Jumpers (Hex)	Address (Hex)	Output (Hex)		$\overline{\text{ROM 4}}$	$\overline{\text{ROM 3}}$	$\overline{\text{ROM 2}}$	$\overline{\text{ROM 1}}$
D	F						
D	E						
D	D						
D	C						
D	B						
D	A						
D	9						
D	8						
D	7						
D	6						
D	5						
D	4						
D	3						
D	2						
D	1						
D	0						
C	F						
C	E						
C	D						
C	C						
C	B						
C	A						
C	9						
C	8						
C	7						
C	6						
C	5						
C	4						
C	3						
C	2						
C	1						
C	0						

Jumpers (Hex)	Address (Hex)	Output (Hex)		$\overline{\text{ROM 4}}$	$\overline{\text{ROM 3}}$	$\overline{\text{ROM 2}}$	$\overline{\text{ROM 1}}$
F	F						
F	F						
F	D						
F	C						
F	B						
F	A						
F	9						
F	8						
F	7						
F	6						
F	5						
F	4						
F	3						
F	2						
F	1						
F	0						
E	F						
E	E						
E	D						
E	C						
E	B						
E	A						
E	9						
E	8						
E	7						
E	6						
E	5						
E	4						
E	3						
E	2						
E	1						
E	0						

Figure 22 (Con't)

4K/16K WIRE WRAP PIN OPTIONS

4K DYNAMIC RAM OPTION



16K DYNAMIC RAM OPTION

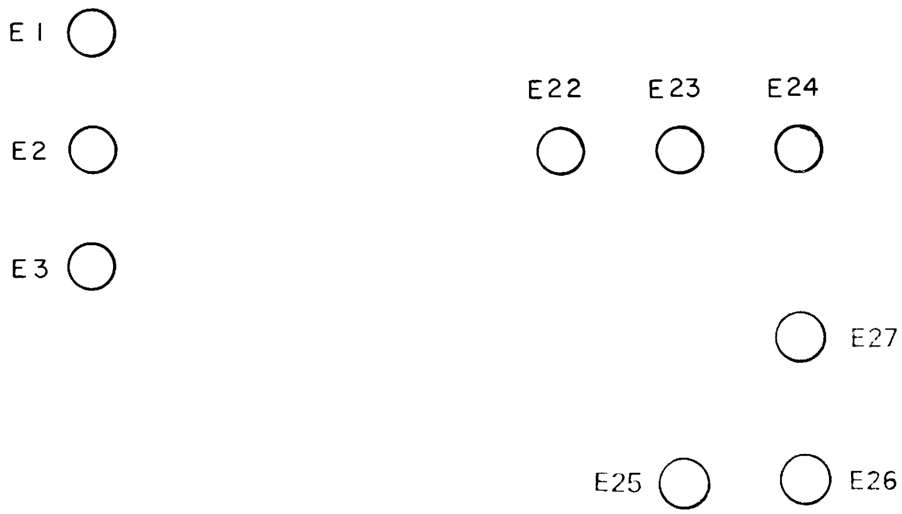


Figure 23

SDB-80 ROM / PROM SOCKETS #1 — #5
JUMPER CONNECTIONS.

THIS ILLUSTRATES THE WIRE WRAP JUMPER CONNECTIONS FOR THE FOLLOWING:

ALL 1K PARTS	2K AND 4K OPTIONS
MK 2708 1K MOS PROM	MK 34000 2K MOS ROM
MK 30000 1K MOS RAM	MK 32000 4K MOS ROM
8252708 1K BIPOLAR PROM	

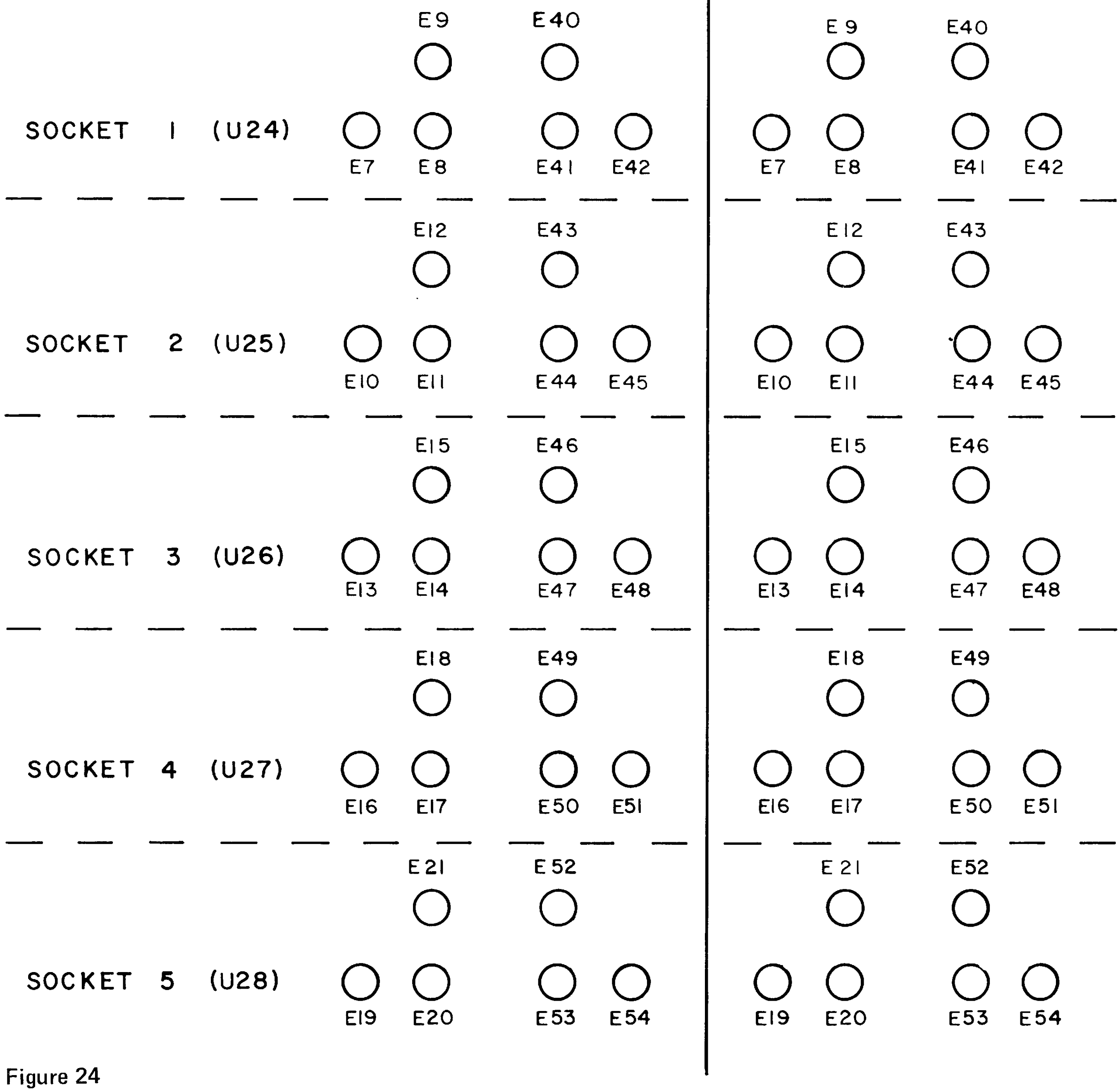


Figure 24

JUMPER CONNECTION EXAMPLES

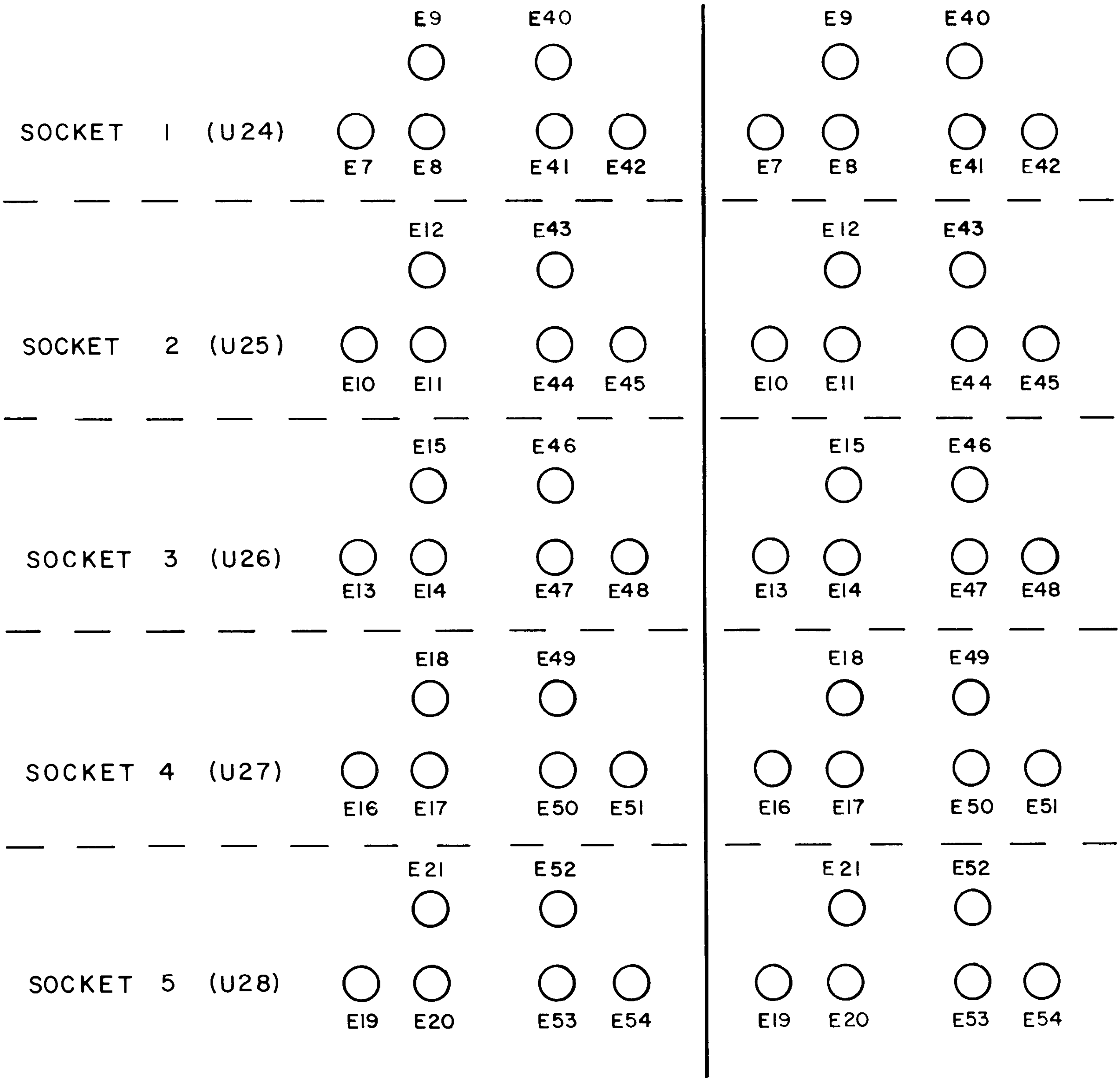


Figure 25

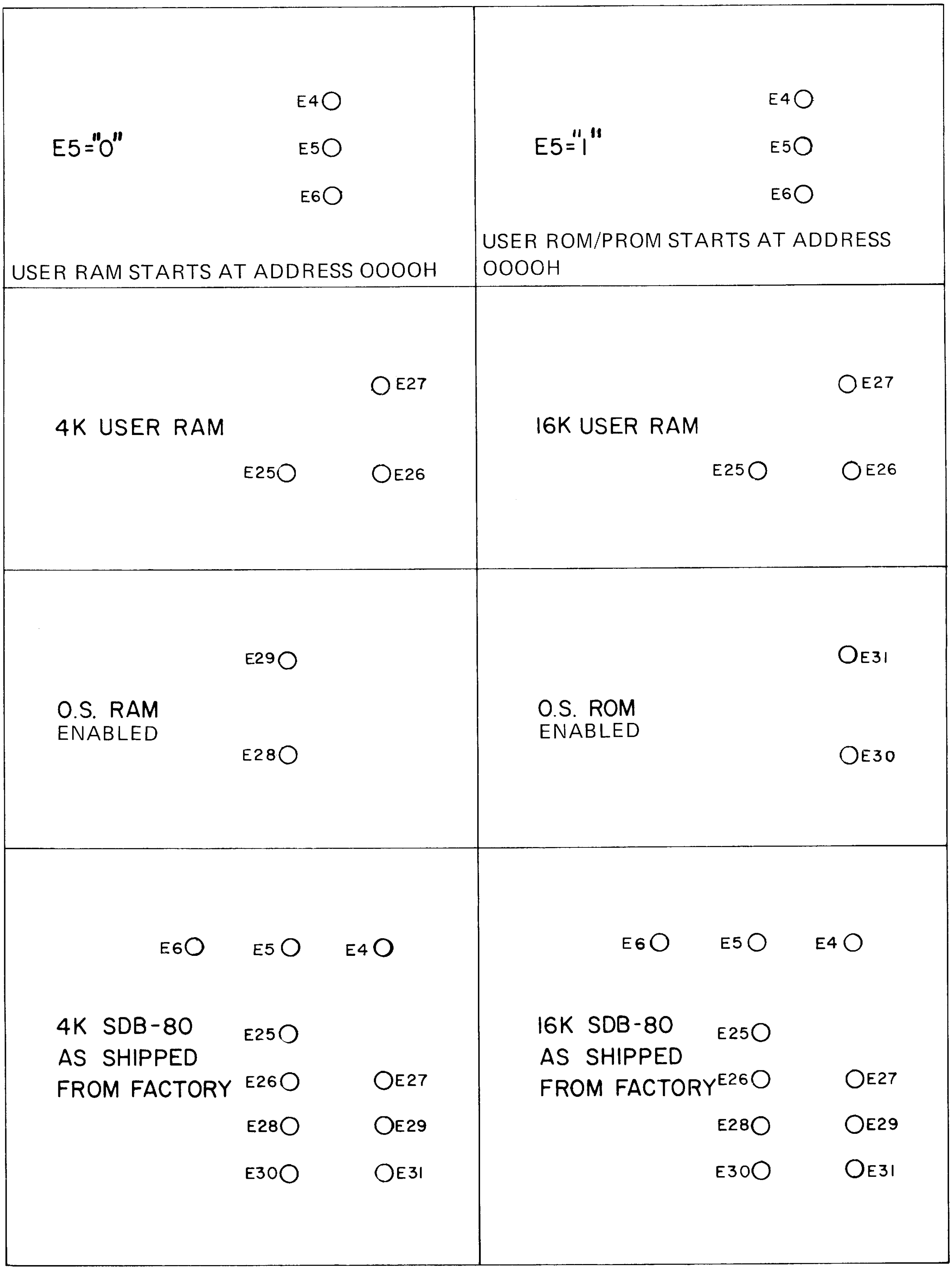

E5="0"
E4
E5
E6
USER RAM STARTS AT ADDRESS 0000H
E5="1"
E4
E5
E6
USER ROM/PROM STARTS AT ADDRESS 0000H
4K USER RAM
E27
E25
E26
16K USER RAM
E27
E25
E26
E29
O.S. RAM ENABLED
E28
E31
O.S. ROM ENABLED
E30
E6
E5
E4
4K SDB-80 AS SHIPPED FROM FACTORY
E25
E26
E27
E28
E29
E30
E31
E6
E5
E4
16K SDB-80 AS SHIPPED FROM FACTORY
E25
E26
E27
E28
E29
E30
E31

Figure 26

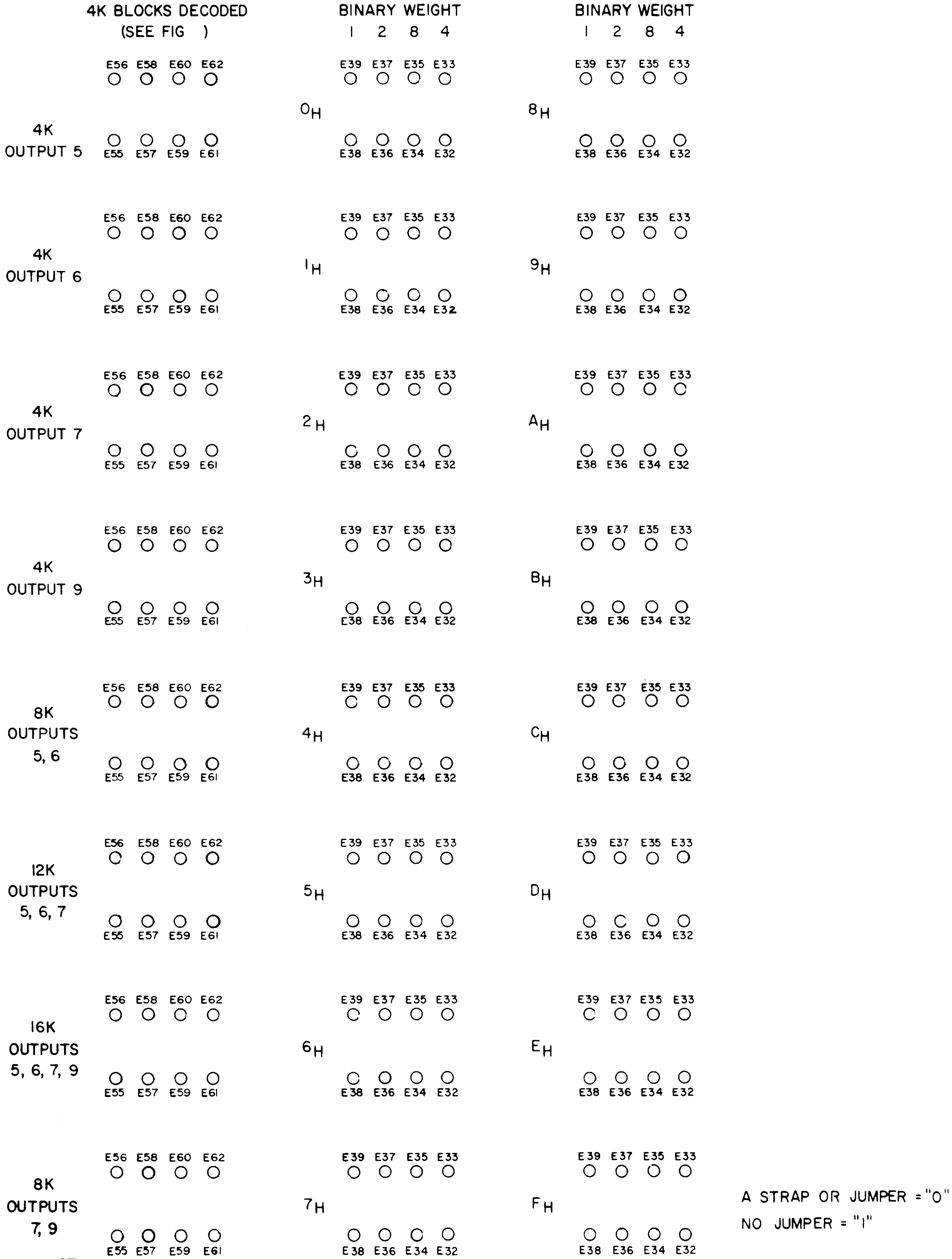

4K BLOCKS DECODED
(SEE FIG)
BINARY WEIGHT
1 2 8 4
BINARY WEIGHT
1 2 8 4
E56 E58 E60 E62
E55 E57 E59 E61
E39 E37 E35 E33
E38 E36 E34 E32
4K OUTPUT 5
4K OUTPUT 6
4K OUTPUT 7
4K OUTPUT 9
8K OUTPUTS 5, 6
12K OUTPUTS 5, 6, 7
16K OUTPUTS 5, 6, 7, 9
8K OUTPUTS 7, 9
0_H
1_H
2_H
3_H
4_H
5_H
6_H
7_H
8_H
9_H
A_H
B_H
C_H
D_H
E_H
F_H
A STRAP OR JUMPER = "0"
NO JUMPER = "1"

Figure 27

C. RAM MUX, BUFFER CONTROL, & MEMORY TRI-STATE BUFFER

I) RAM MULTIPLEXER (REFER TO FIGURES 1 & 6).

Two multiplexer parts (U18-U19) are required by the 16 pin 4K (16K) dynamic RAM. The purpose of the multiplexers is to alternately supply one set of addresses to the memories and then upon command supply a second set. These two sets comprise the entire address. ROW address information (A0-5(6)) is strobed on the negative (leading) edge or $\overline{RAS}$ while the second or COLUMN address information (A6-11 (7-13)) is strobed on the negative (leading) edge of $\overline{CAS}$. The decoding logic supplies a signal to the multiplexers controlling which set of address lines is selected.

II) BUFFER CONTROL & MEMORY TRI-STATE BUFFER

Two buffers are controlled by the logic in the memory section: 1) Memory Tri-State Buffer (U1) and 2) Data Bus Buffer (U58-U71). The condition of each is directly dependent upon the board memory and addressing.

The function of the Memory Tri-State Buffer is to isolate the output data lines of both the dynamic RAM and the 5 ROM/PROM sockets from the on board data bus. This is shown clearly in Figure 1. This buffer is enabled (allowed to talk to the DATA BUS) whenever the dynamic RAM, USER ROM, or the O. S. ROM is being read. Information from these memories is buffered onto the DATA BUS.

The DATA BUS buffer is bidirectional and is controlled by two lines:

1) Signal $\overline{DINB}$ and 2) $\overline{DRIVEB}$ generated in the memory section. The bidirectional operation of the DATA BUS BUFFER is covered in the section entitled "OUTPUT BUFFERS".

RESTART

Figure 1 shows the circuitry associated with the restart function of the SDB-80. Depending upon the state of S2 the board will restart to either 0000_H, the bottom of the memory map, or the $E000_H$, the location of the operating system. Restart occurs during board power up or by pressing push button S1.

I. Power up Restart

During power up, point A is held near ground by the timing capacitor C5 while power is applied to the rest of the board. As long as A is below the trigger point of U4 $\overline{\text{reset}}$ at the CPU is held active low. While $\overline{\text{reset}}$ is low, the CPU is initialized and the data and address lines go tri-state. The diode, CR2, causes a rapid discharge of point A whenever the power supply goes low.

II. Push Buton Restart

Prior to restart the input of U20 is high, holding the "CLEAR" input to U3 active low. This causes Q to be held low. The output of the edge triggered one-shot ($\overline{Q}$) is held high. This in turn causes the CPU $\overline{\text{reset}}$ to be inactive high. During reset S1 is grounded causing the clear input to U26-1 to go high. The next negitive edge of $\overline{M1}$ clocks a "1" to the output of U3 triggering one shot U2. U2 ($\overline{Q}$) goes active low for approximately 10 us. This causes $\overline{\text{reset}}$ to respond active low for the same period of time. The leading edge of $\overline{\text{reset}}$ is synchronized with the TI state during M1, the instruction OP code fetch cycle.

III. Restart Location, 0000_H or $E000_H$

During $\overline{\text{reset}}$ the program counter is forced to zero, the CPU is initialized

RESTART SCHEMATIC

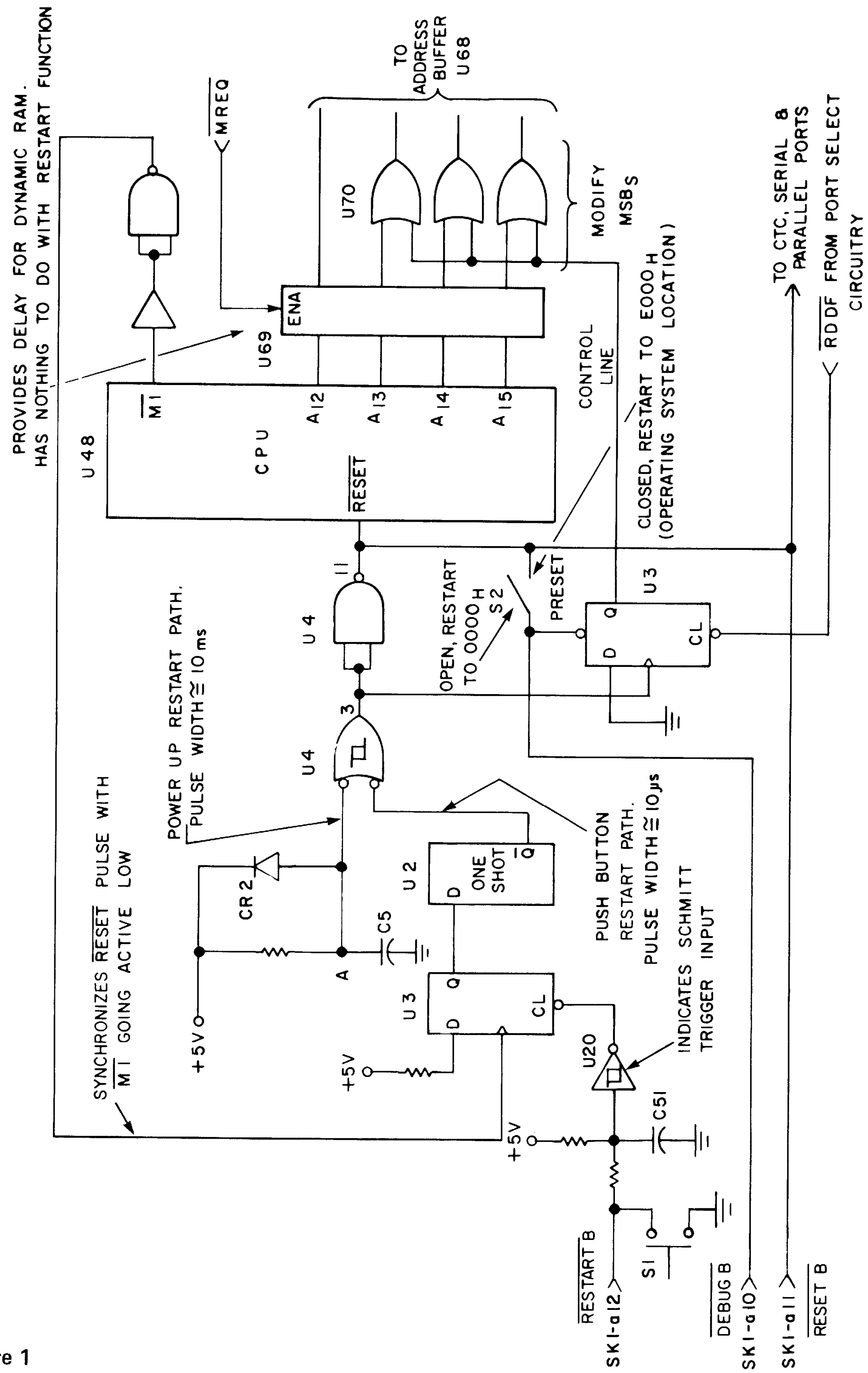


Figure 1

and the data and address lines go to a high impedance state. Upon termination of $\overline{\text{reset}}$ the program counter outputs to the Address Bus (all zeros) and the CPU does an OP code fetch. Thus the CPU tries to do a fetch from memory location 0000_H. However, this may be modified by the board such that restart may occur to $E000_H$ instead.

The difference between address $E000_H$ and 0000_H is in the most significant bits. Looking at the four MSBs for 0 and E.

HEX		BINARY
0	=	0000
E	=	1110

One sees that by changing the three most significant binary bits one can change the address from 0000_H to $E000_H$. This change is accomplished by the OR gates, U70. Each gate has two inputs; one data and one control signal. The data input (A13, A14, A15) is normally passed on through. However, when these three most significant address lines need to be forced to a "1" the control line goes high. Thus during $\overline{\text{reset}}$ the CPU address lines output all zeros. If, however, restart to the operating system is desired the control line goes high changing the address to the card (and memory).

	HEX	BINARY
From:	0000	0000 0000 0000 0000
To:	E000	1110 0000 0000 0000

IV Control Line Circuitry

Restart to 0000_H or $E000_H$ is determined by toggle switch S2 and the D FF U26. Under restart to 0000 (or no restart command) the output of U3 (control line) is held at ground and U70 simply provides a non-inverting buffer function for A13-15. With S2 open, restart to 0000_H, Q26-2 remains at ground for any restart pulse. A positive going edge of point B clocks the FF U3 and transfers data from D to Q. In this case, with D grounded, a "0" is always transfered.

Restart to $E000_H$ involves pulling the PRESET line of U3 active (ground) through one of two ways:

1) $\overline{\text{DEBUGB}}$ line from SK1-a10

or

2) logically through Switch S2

With S2 closed restart to $E000_H$ is generated in the following way:

1) A restart pulse is initiated either by Power Up or by S1.
2) The positive edge of B tries to clock U3 to a "0".
3) $\overline{\text{Reset}}$ goes low forcing U3 high and overriding the effect of 2.
4) The CPU outputs all zeros on the Address Bus.
5) The OR gates (U70) force the three MSBs of the address lines.
6) Memory is addressed at location $E000_H$ (operating system) for the OP code fetch.
7) During the operating system initialization port DF_H is read which clears U3 (Q = 0).

V. Four Bit Latch Function

The four bit latch U69 is used to ensure the upper four address lines remain stable during the entire memory cycle. The lower address bits are latched in the dynamic RAMs and do not need external latches.

OUTPUT BUFFERS: ADDRESS, DATA AND CONTROL

Non-inverting party line transceivers (DM 8833) are used to buffer both the Address and Data Bus lines. The receiver portion incorporates hystersis for noise immunity considerations while the driver provides high sink and source currents. Either inverting (8835) or non-inverting (8833) buffers may be used depending upon the user's requirement. A block diagram of the buffering circuit is shown in Figure 1.

I. Address Bus Buffer

The CPU address bus drives only the 8833 buffer which in turn drives both the external and internal bus. The one exception to this (the gating of A12-A15) is described in the RESTART section. The receive portion of the buffer is continuously active. The drive portion is active until a peripheral device requests and gains access to the bus. At that time BUSAK will respond by tri-stating the driver so that both the CPU and the requesting peripheral device will not drive the bus at the same time.

II. Data Bus Buffer

In contrast to the CPU address lines, the CPU Data Bus drives the internal board bus as well as the output buffers. These buffers, in turn, drive the external Data Bus. One reason the drive arrangement is different from the address lines is that the CPU Data Bus has twice the drive capability of the Address Bus. Bus direction control is somewhat more complex than for the address section. The driver (to the external bus) direction is controlled from on board by $\overline{\text{DRIVEB}}$ while the receiver (from the external bus) is controlled externally by $\overline{\text{DINB}}$.

BUFFERING OF ADDRESS, DATA AND CONTROL LINES

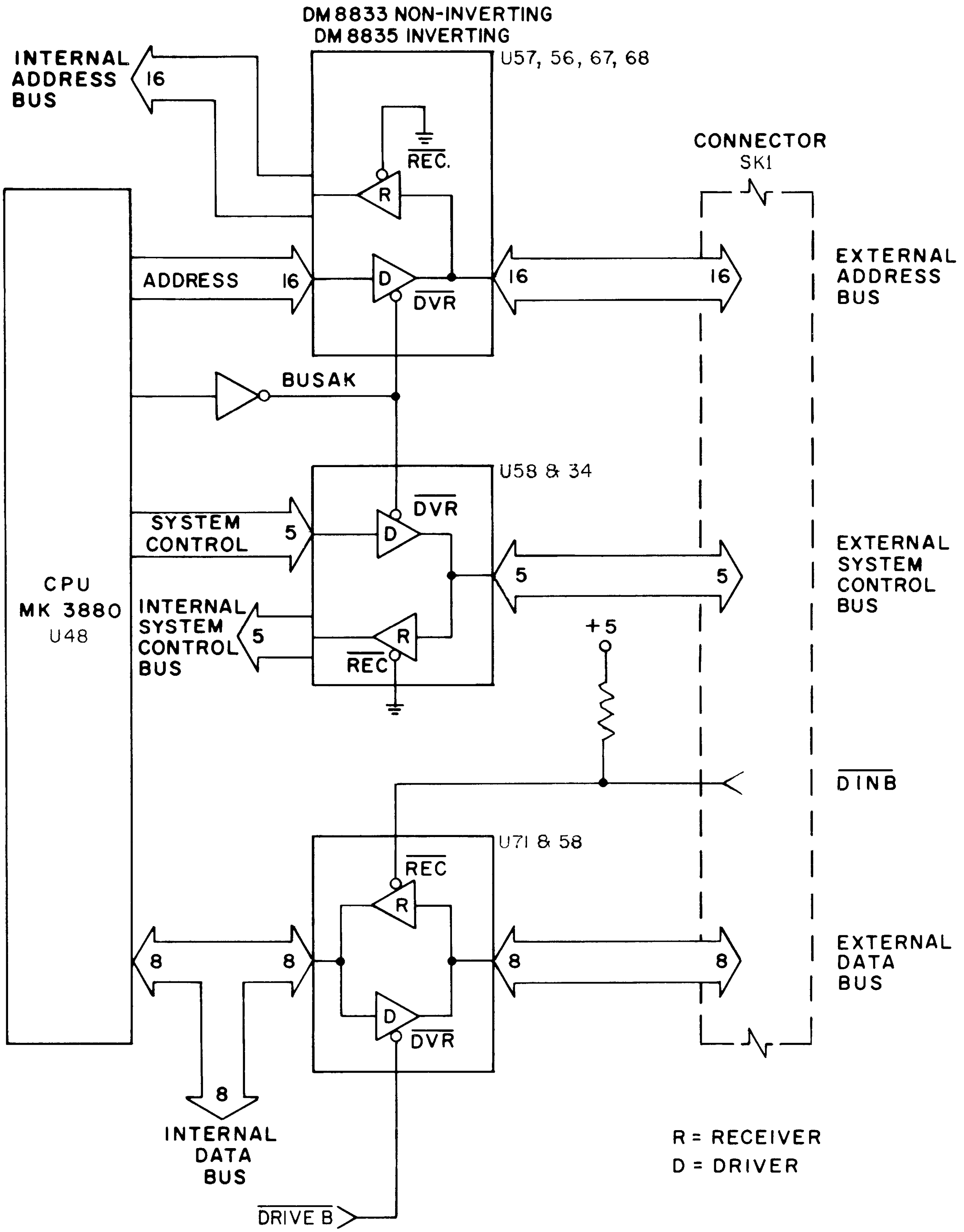


Figure1

The arrangement for switching between the driver and receiver has been done in such a way as to try to reduce current spikes and transients caused when driver or receiver are ON and momentarily fighting a low impedance drive on either the internal or external bus. Major bus conflicts (between the driver and some low impedance buffer on the external bus, for example) are avoided only through proper design of the external interface circuitry.

A timing diagram of an arbitary memory cycle is shown in Figure 2. Assume for the first cycle conditions are such that the driver is active ($\overline{\text{DRIVEB}}$ = 0). As seen from the figure the two buffers are never active at the same time. In fact, there is approximately a 400 ns dead time where neither are on. This dead time will help prevent any conflict between the driver and a low impedance on the external bus or the receiver and a low impedance on the internal bus.

Control of the driver and receiver is dependent upon two criteria:

1) Preventing a conflict between either the driver or receiver and another buffer elsewhere.

2) Determining where the data is required once it is generated. For example, data generated at the CPU may be required on both internal and external bus lines. In this case the driver portion of the data buffer would become active.

Control of the driver ($\overline{\text{DRIVEB}}$ active low) is determined by the following:

1) Any non-DMA memory write cycle or I/O write cycle. ($\overline{\text{DRIVEB}} = \overline{\text{MREQ}\cdot\text{WR}\cdot\text{DECODE} + \text{IORQ}\cdot\text{DECODE}}$; + BUSAK). This condition implies that any time the CPU writes to a memory or I/O device (on

DRIVER RECIEVER TIMING DIAGRAM

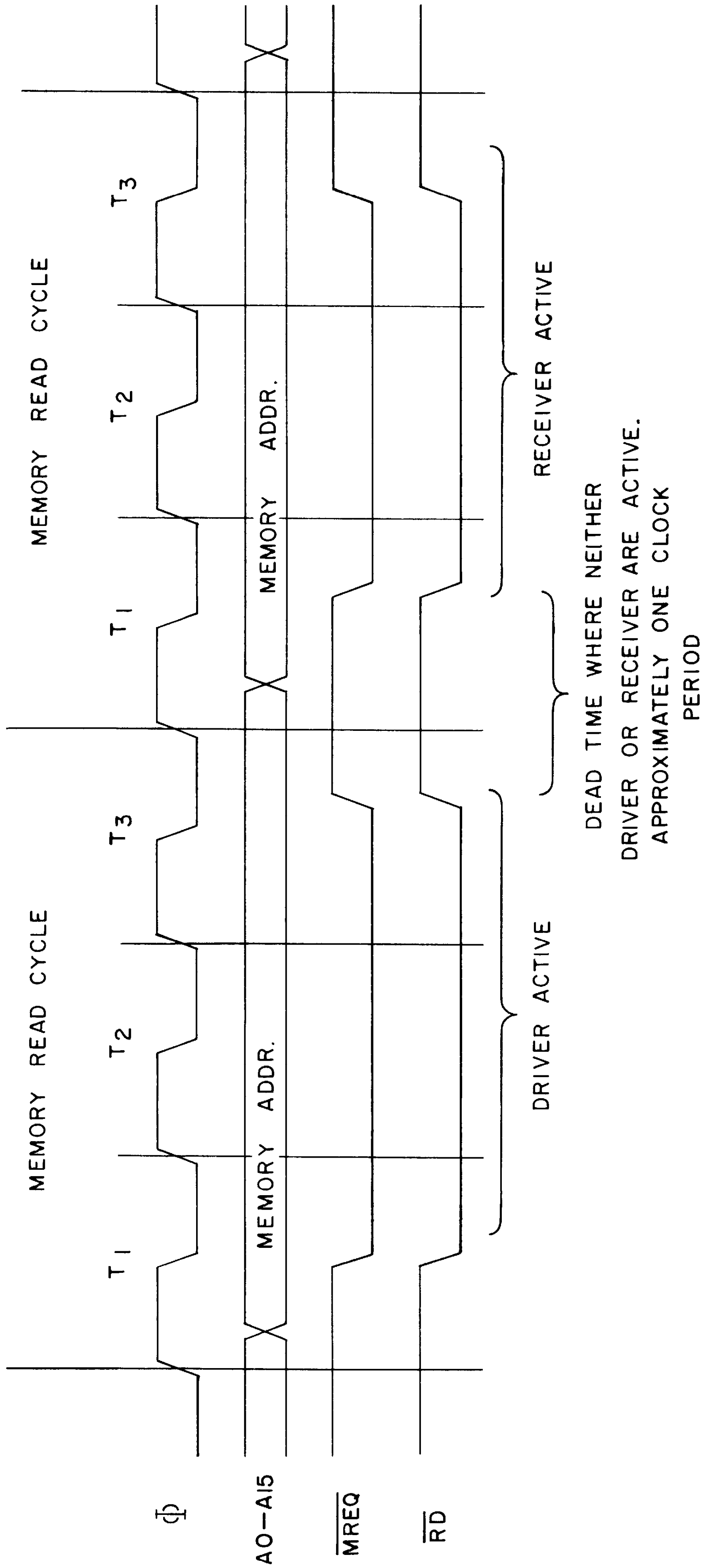


Figure 2

or off) the data must appear on both the internal and external bus lines because the receiving device could be either place. Only the actual device that is uniquely decoded will make use of the data. No conflicts occur because when the CPU and Driver are outputting, all other bus drivers are high impedance.

2) On board memory read cycle ($\overline{DRIVEB} = \overline{MREQ} \cdot \overline{RD} \cdot \overline{DECODE}$)

An on board memory read operation may be initiated by the CPU (as in OP-CODE fetch) or may be executed by the external bus (as in DMA operations). During this condition data needs to be placed on both the internal and external Data Bus to allow PIO chips (either internal or external) to interpret their return from interrupt instructions appearing on the Data Bus.

Control of the receiver ($\overline{DINB}$ is active low) is determined by the following: (on-board = SDB-80, off board = other cards).

1) Any memory write cycle or any I/O write cycle <u>initiated off the board</u> ($\overline{DINB} = (\overline{MREQ \cdot WR \cdot DECODE + IORQ \cdot WR \cdot DECODE)\ BUSAK}$). An off board write signal could be generated by a direct memory access command. This condition implies that any time the <u>external bus</u> writes to memory or I/O (on board or off) the data must appear on both the internal and external bus lines because the receiving device could be either place.

2) Any memory or I/O read cycle <u>initiated on the board</u> to devices <u>off the board</u> ($\overline{DINB} = \overline{MREQ \cdot RE \cdot DECODE + IORQ \cdot RD \cdot DECODE}$). This states that any time the CPU reads peripheral I/O or memory, that data must be brought to the Internal Data Bus.

3) Interrupt acknowledge to an off board peripheral device ($\overline{DINB} + \overline{MI \cdot IORQ}$; IEO $\neq$ IE1). When an external peripheral chip has requested

an interrupt and been granted an interrupt acknowledge (M1·IORQ) by the CPU the receiver must go active to place the interrupt vector on to the Internal Data Bus for the CPU.

III. System Control Bus

These five lines ($\overline{WR}$, $\overline{RD}$, $\overline{MREQ}$, $\overline{IORQ}$, $\overline{RFSH}$) do the memory control functions. The circuitry is essentially the same as for the Address Bus shown in Figure 1. Under most situations both the dirver and receiver are active. When the external bus gains control (BUSAK = 1) only the receiver is active, allowing the external bus to control the internal bus. All other bus lines are simply bussed onto or off of the card as the case may be.

COUNTER/TIMER

For counting, timing, digital delay applications, etc., a four channel Z-80 Counter/Timer Circuit (MK3832) has been included on the SDB-80 Board. One channel (0) is used (in conjunction with the operating system) for baud rate measurement and clock generation leaving three available to the user. Channel 0 is availavle, however, in OEM applications if the CTC baud rate output is not used. The block diagram for this section is shown in Figure 1.

I) I/O Buffers

Each I/O line that is brought out to the connector SK2, is buffered by a 7404 hex inverter (U54). This buffering provides added drive capability and protection for the MOS CTC chip. The polarity of the I/O lines can be seen from Figure 1.

II) Port Addresses

Channel 0	$D8_H$
Channel 1	$D9_H$
Channel 2	DA_H
Channel 3	DB_H

III) Functional Description

Each Counter Timer Circuit (CTC) has four counting channels. Each channel, in turn, has two 8-bit counters; a down counter and a pre-scale counter. The down counter may be driven from either the pre-scaler or from an external count input (CLK/TRG). Loading of the down counter is under software control and the count modulus may range from 1 to 256.

BLOCK DIAGRAM COUNTER/TIMER SECTION

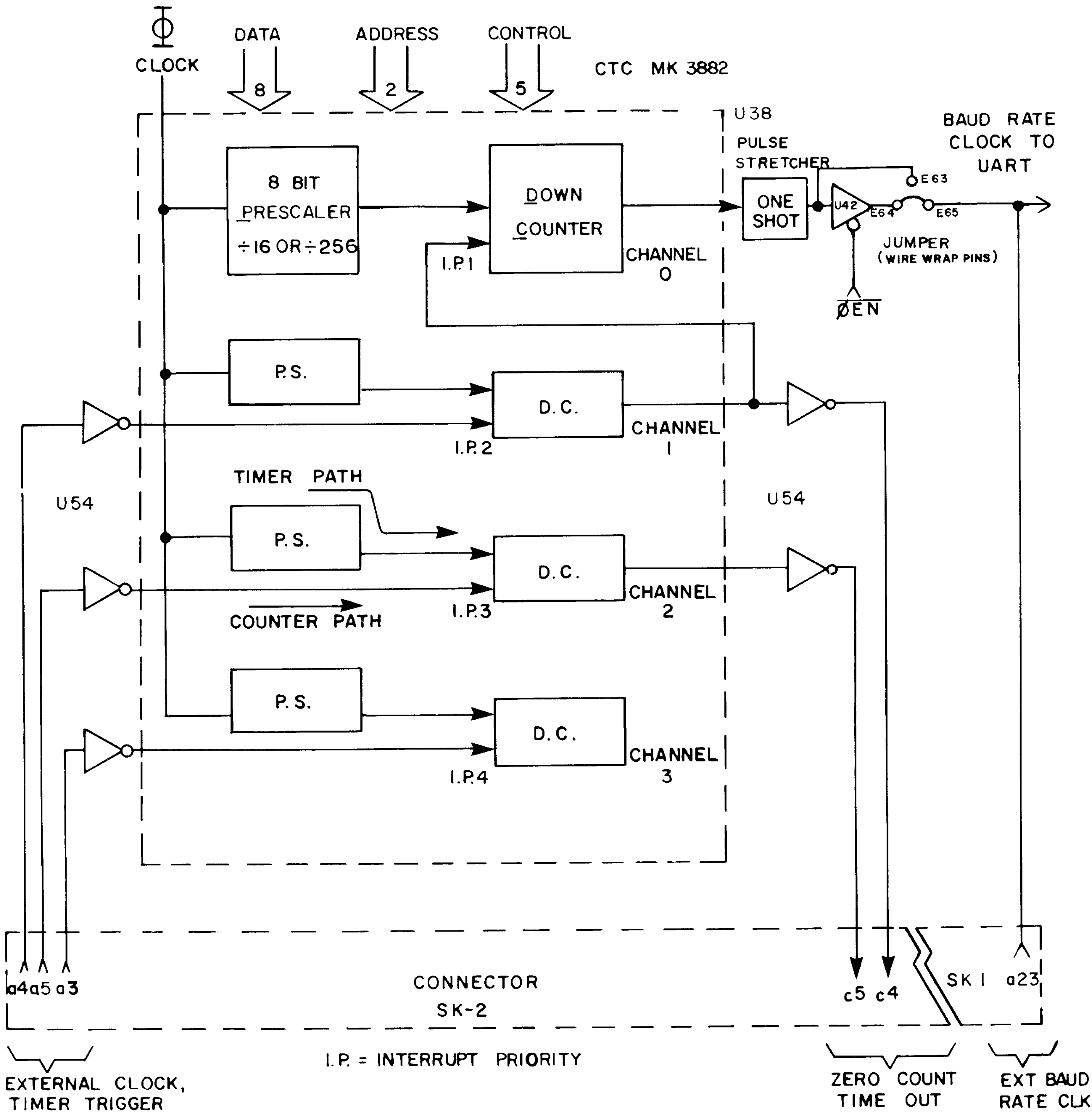


Figure 1

Each channel is considered to have two modes of operation. The timer mode counts the clock (Ø) after it has been pre-scaled by a factor of 16 or 256 while the counter mode counts positive transitions on the CLK/TRG line. The output of a given channel occurs when the down counter reaches the all "0" state.

Several channels may be cascaded for longer count sequences. The zero count output of channel #1 is hardwired to the CLK/TRG input of channel #0 as shown in Figure 1. Because of pin limitations on the CTC package, the zero count output of channel 3 is not brought out. Channel 3 can react with the SDB-80 through the Z-80 interrupt structure.

IV) Jumper Option

The jumper option wire wrap pins (shown in Figure 1) are located near the mid right hand side of the PC board. Figure 1 shows the option as wired at the factory. Operation of this section is as follows:

1) As long as the on board clock is used, the clock enable ($\overline{ØEN}$) enables buffer U42 which supplies the UART with the correct baud rate clock. An external clock (Ø) supplied to the board will cause $\overline{ØEN}$ (through ØEN SK1-a14) to go high which Tri-States U42. This allows an external baud rate clock to be supplied through SK1-a23. Because the clock frequency will, in all likelihood, be changed, the operating system can no longer be counted upon to determine the correct baud rate. Thus, the need for an external baud rate clock. This clock of course, must be a factor of 16 times the desired baud rate for proper UART operation.

2) The above function may be disabled by rewiring the jumper so that E64 connects to E65.

3) An external baud rate clock may be applied at any time through SK1-a23 by not connecting E65 to either E64 or E63.

V) Counting

A typical timing wave form of counter output #1 is shown in Figure 2.

TIMER MODE:

1) From the CTC data sheet it can be determined that the pulse width is fixed at 1.5 x system clock period. For the SDB-80 (2.458 MHz clock) This is PW = 1.5 x 406.8 ns = 610 ns.

2) The output period (T) is given by:

T = tc x P x TC

where tc = system clock period

P = Prescale units (16 or 256)

TC = Time constant or modules count of down counter (1 to 256)

The minimum period for the SDB-80 is:

T = 406.8 x 16 x 1 = 6.51 us

The maximum period for one channel is:

T = 406.8 x 256 x 256 = 26.66 ms

COUNTER MODE:

1) From the CTC data sheet it is determined that the pulse width is fixed at 1.5 x system clock period. For the SDB-80 this is:

PW = 1.5 x 406.8 ns = 610 ns

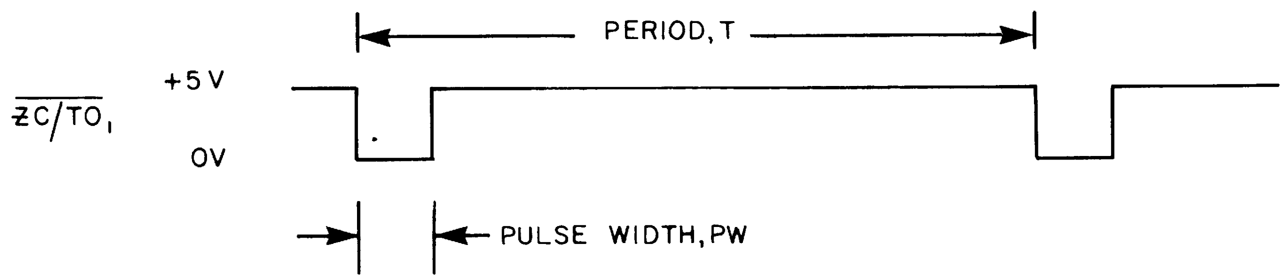
2) The output period (T) is given by:

T = TC x TI

Where TC = Time constant or modules of down counter (1 to 256)

TI = Period of input pulse.

TIMING WAVEFORM OF COUNTER OUTPUT



TIMER MODE:

PULSE WIDTH (PW) = 1.5 X SYSTEM CLOCK PERIOD (t_c) = 610 ns FOR SDB-80

PERIOD (T) = t_c X P X TC

WHERE t_c = SYSTEM CLOCK PERIOD
P = PRESCALE UNITS (16 or 256)
TC = TIME CONSTANT OR MODULUS COUNT OF DOWN COUNTER (1 to 256)

COUNTER MODE:

PERIOD (T) = TC X TI

WHERE TC = TIME CONSTANT OR MODULUS COUNT OF DOWN COUNTER (1 to 256)
TI = PERIOD OF INPUT PULSE

Figure 2

The period of the input pulse, TI, must be a minimum of twice the system clock period i.e., TI can count at only 1/2 the clock rate, $\emptyset$).

The minimum period for the SDB-80 is:

$$T = 1 \text{ (TC)} \times 2 \text{ (406.8 ns)}$$
$$= 813.6 \text{ ns}$$

The maximum period for one channel is ∞. This implies an ∞ period input pulse).

CLOCK GENERATOR

The SDB-80 has an on board crystal controlled clock generator. The crystal frequency, set at twice the desired CPU clock rate, is divided by two to form the square wave required to drive the CPU. A clock frequency of 2.458 MHz has been selected as being a multiple of the baud rate frequencies as well as being close to the maximum frequency for the standard chip set.

Buffering the 2.458 MHz system clock to the outside world creates potential skew problems. To minimize these problems, the SDB-80 clock is derived from the master system clock (ØB), buffered, and routed to all chips requiring Ø (as shown in Figure 1). On any additional boards the master system clock would be brought into the card edge, buffered, and routed in the same fashion as the SDB-80. By using the same type buffering arrangement on all cards, the clock skew, board-to-board, can be minimized.

Figure 2 shows the buffering scheme used to distribute the clock on the SDB-80. This buffer uses a push-pull approach to provide a low impedance drive to each end of the power supply, i.e., from 0V to + 5V. Some of the major features of this buffer are:

1) Push-pull action is provided by Q1 (PNP) and the pull-down transistor (NPN) in inverter 12.
2) R3 provides the D. C. drive to turn Q1 ON.
3) C1 is a speed up capacitor providing transient drive to Q1.
4) R2 helps keep Q1 OFF when no drive is supplied.
5) R1 limits the surge current through Q1.

CLOCK BUFFER BLOCK DIAGRAM

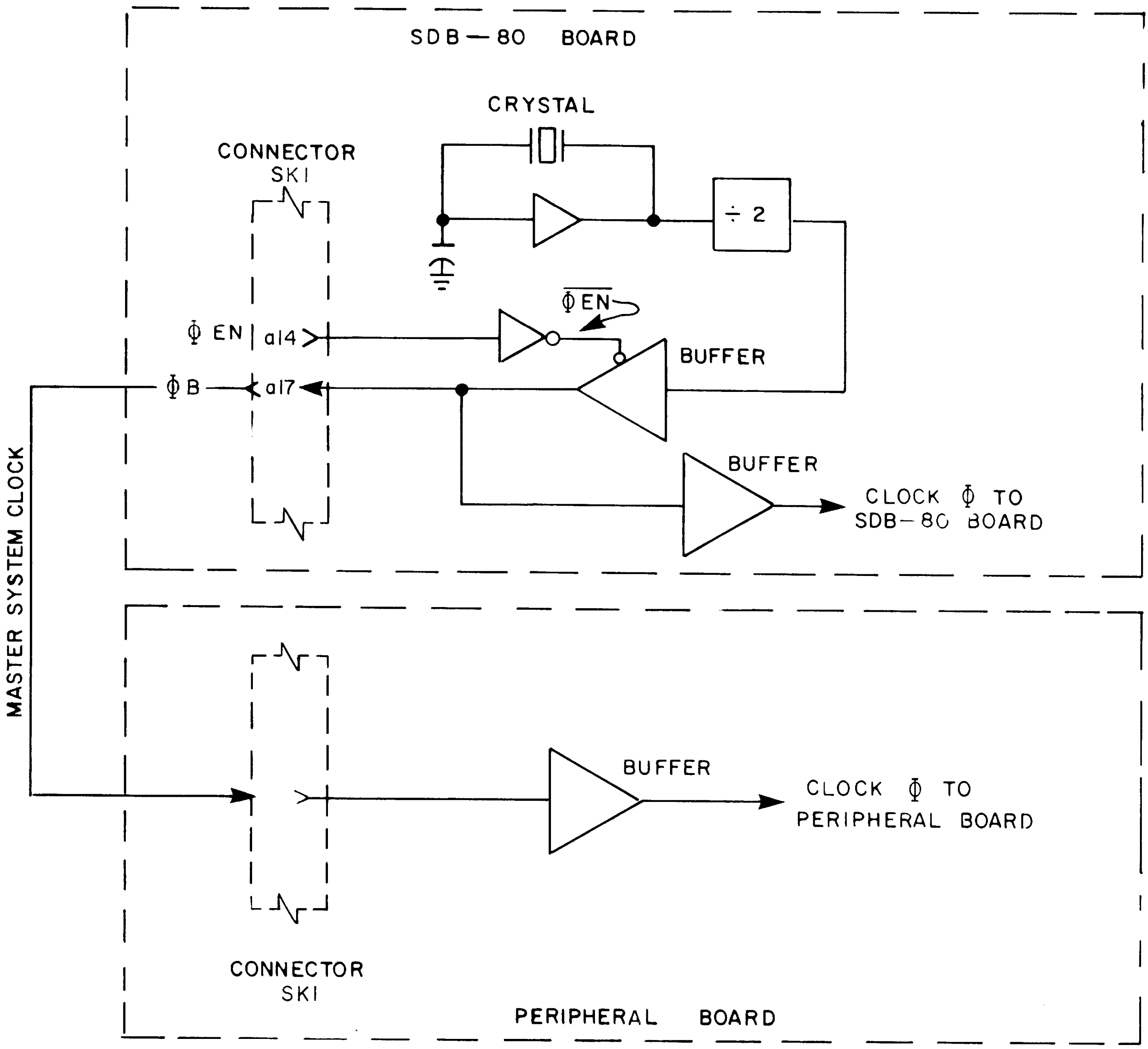


Figure 1

CLOCK BUFFER SCHEMATIC

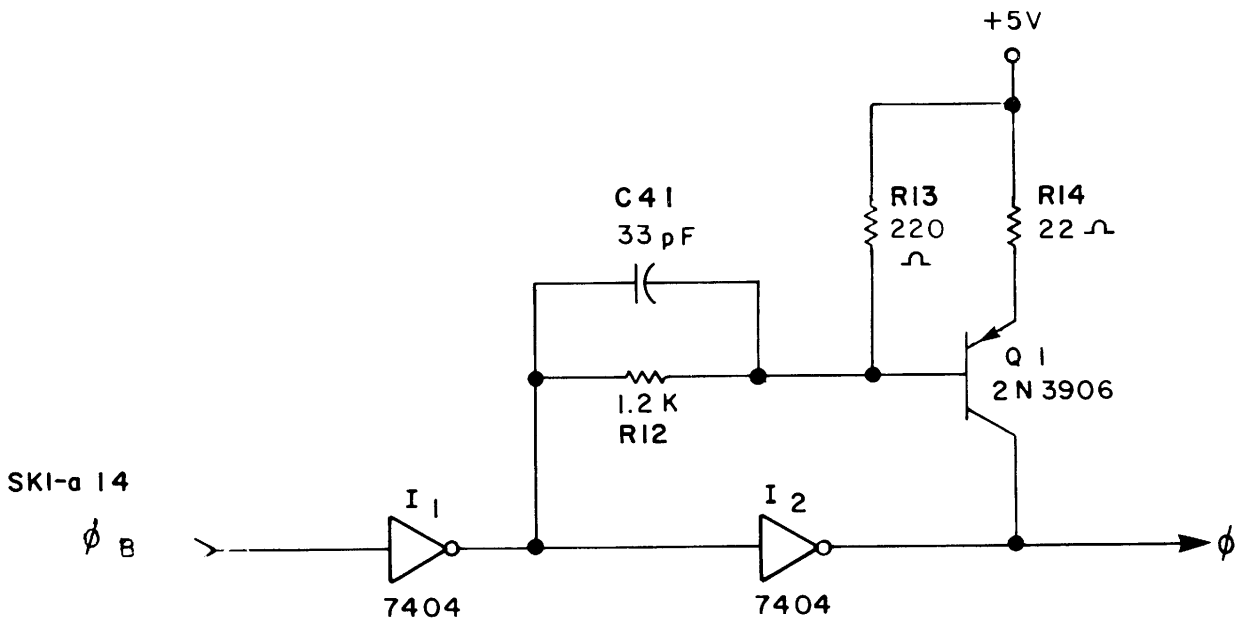


Figure 2

INTERRUPTS

The purpose of an interrupt is to allow peripheral divices to suspend CPU operation in an orderly manner and force the CPU to start a peripheral service routine. Usually this service routine is involved with the exchange of data, or status and control information, between the CPU and the peripheral. Once the service routine is completed, the CPU returns to the operation from which it was interrupted. The block diagram of this portion of the SDB-80 is shown in Figure 1.

I. Modes

Non-maskable:

A nonmaskable interrupt will be accepted at all times by the CPU. When this occurs, the CPU ignores the next instruction that it would have fetched and instead does a subroutine call to location 0066_H. Thus, it behaves exactly as if it had executed a call instruction but, the return instruction must be a RETN.

Maskable:

Mode 0

This mode is indentical to the 8080A interrupt response mode. With this mode, the interrupting device can place any instruction on the data bus and the CPU will execute it. Thus, the interrupting device needs to provide the

PRIORITY INTERRUPT STRUCTURE

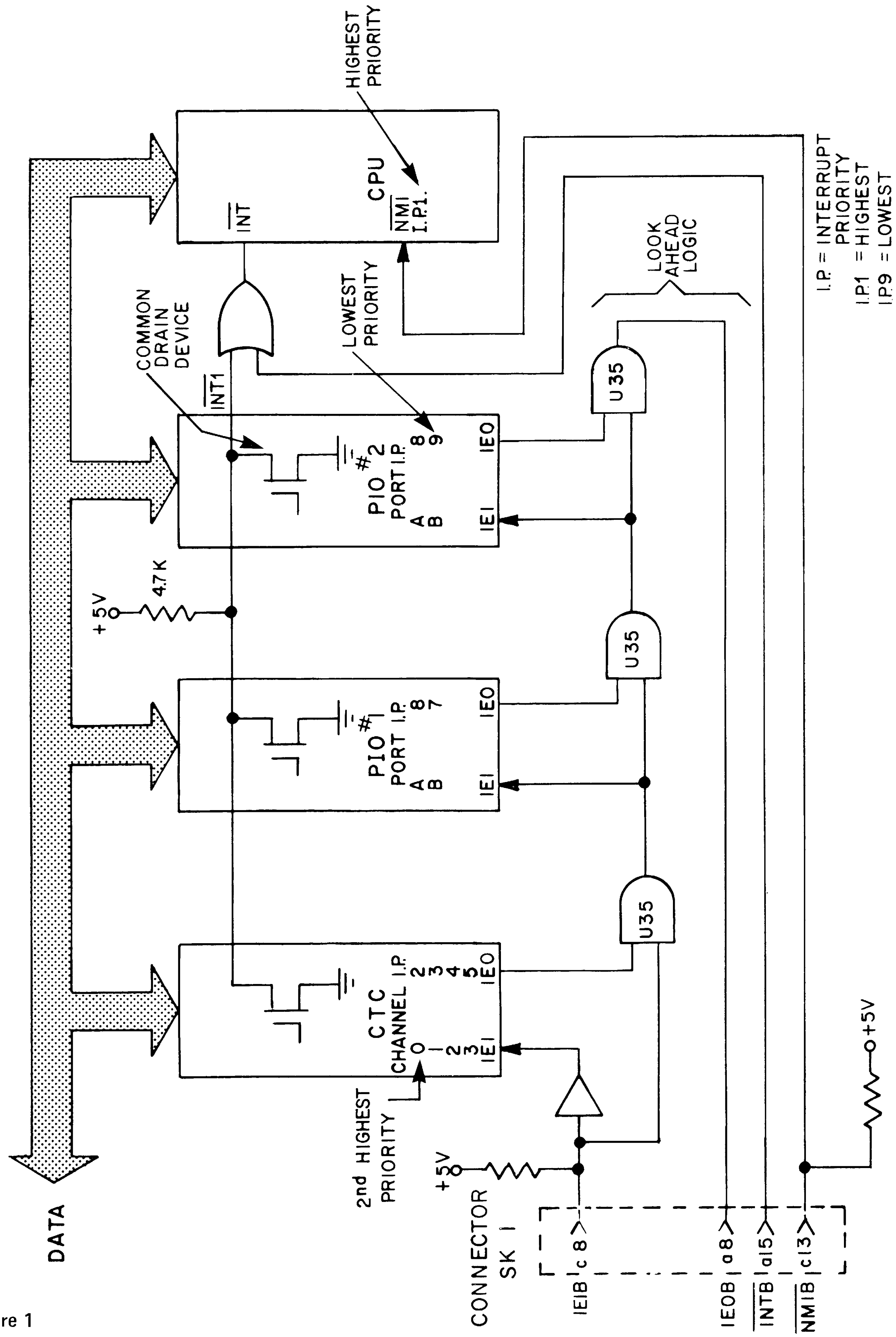


Figure 1

next instruction to be executed instead of the memory. Typically this is a restart or call instruction.

Mode 1

When this mode has been selected by the programmer, the CPU will respond to an interrupt by executing a restart to location 0038H. Thus the response is identical to that for a non-maskable interrupt except that the call location is 0038H instead of 0066H.

Mode 2

This mode is the most powerful interrupt response mode. With a single 8-bit byte from the user an indirect call can be made to any memory location. With this mode the programmer maintains a table of 16-bit starting addresses for every interrupt service routine. This table may be located anywhere in memory. When an interrupt is accepted, a 16 bit pointer must be formed to obtain the desired interrupt service routine starting address from the table. The upper 8-bits of this pointer is formed from the contents of the I register. The lower eight bits of the pointer must be supplied by the interrupting device, which is typically software programmable.

II. Interrupt Servicing

At some predetermined condition, such as CTC counter timeout or data being strobed into a PIO port, a peripheral chip will generate a condition for interrupting the CPU. During this time the common open drain interrupt line $\overline{\text{INT}}$ or $\overline{\text{INTB}}$ will be pulled active low by the peripheral requesting the interrupt. Sometime later the CPU will send out an interrupt acknowledge ($\overline{\text{INTA}}$). During the $\overline{\text{INTA}}$ the interrupt logic of the peripheral chip will determine the highest priority port which is requesting an interrupt. This device then places the contents of its 8-bit interrupt vector on the data bus for the CPU. The interrupt condition is maintained until the end of the INTA cycle. Lower priority interrupts are inhibited until this device decodes an RETI instruction. If more than one peripheral requests interrupt servicing at the same time, a priority status is established.

Priority is determined by the interrupt enable lines- IEI and IEO - and internal logic on each peripheral chip.

The following table defines interrupt priority status:

IEI	IEO	STATUS
0	0	requesting interrupt but low priority
0	1	undefined (not allowed)
1	0	requesting interrupt highest priority
1	1	no interrupt

III. Daisy Chain

All Z-80 peripheral devices include daisy chain priority interrupt logic that automatically supplies the programmed vector (from the highest priority interrupting peripheral) to the CPU during interrupt acknowledge. The daisy chained peripherals on the SDB-80 board are one CTC and two PIO chips. To ensure that more than the three on board peripherals (from a speed standpoint) can be included in the interrupt priority loop, "look-ahead" logic has been implemented on the board. Both ends of the board daisy chain logic have been brought to connector SK1 so that the board priority within a larger daisy chain system can be established. Board priority is determined in the same fashion as an individual peripheral chip; i.e., thru the high or low state of IEIB or IEOB.

SPECIFICATIONS

BASIC CPU SET

Z80

1 CPU - MK3880
2 PIO - MK3881
1 CTC - MK3882

WORD SIZE

Instruction: 1-4 Bytes 8-32 Bits
Data: 8 Bits
Address: 16 Bits

CRYSTAL FREQUENCY

4.916 MHz

CLOCK FREQUENCY, Ø

2.458 MHz

CLOCK PERIOD, T

406.8 ns

CYCLE TIME FOR SIMPLE 8 BIT ADD

4 x 406.8 = 1.627 µs

MEMORY CAPACITY

On Board RAM:

256 Byte Static	3539
4K Byte Dynamic	MK4027
16K Byte Dynamic	MK4116

On Board ROM/PROM:
5 sockets for any combination of

1K Bytes MOS PROM	MK2708
1K Bytes Bipolar PROM	82S2708 (Signetics)
1K Bytes MOS ROM	MK30000
2K Bytes MOS ROM	MK34000
4K Bytes MOS ROM	MK32000

INTERRUPTS

Response Modes:
- 0 - fetch instruction, interrupt mode (8080 mode)
- 1 - Vector to fixed restart location
- 2 - Prioritized and Vectored interrupt mode

Daisy Chain Interrupt Servicing

Priority (Mode 2):

Part	Priority
CPU (NMI)	1 (highest)
CTC Ch 0	2
1	3
2	4
3	5
PIO #1 A	6
B	7
PIO #2 A	8
B	9
Board ($\overline{\text{INT}}$)	Board priority

within a larger daisy chain system is determined by the interconnection of the pin signals - IEIB & IEOB. High to low delay between IEIB & IEOB is 60 ns max.

I/O ADDRESSING (Ports)

Port	Address
PIO #1	
D0 - Data	$D0_H$
D0 - Control	$D1_H$
D2 - Data	$D2_H$
D2 - Control	$D2_H$
PIO #2	
D4 - Data	$D4_H$
D4 - Control	$D4_H$
D6 - Data	$D6_H$
D6 - Control	$D6_H$
CTC	
Channel 0	$D8_H$
Channel 1	$D9_H$
Channel 2	DA_H
Channel 3	DB_H
UART	
Data	DC_H
Control	DD_H

SPECIFICATIONS

System Control DE_H
Debug Control DF_H

Ports Reserved For Future System Expansion: E0 thru FF

USER I/O EXPANSION

208 port addresses available to user. 00 thru CF

SERIAL I/O CHARACTERISTICS

On Board UART
Start bit verification
5-8 data bits
1 or 2 stop bits
Odd, Even or No Parity Select
On board baud rate generator

BAUD RATES:	
110	2400
300	4800
600	9600
1200	

Serial Peripheral Compatability:
RS-232
20 mA current loop

Reader step compatability
Capable of driving a mechanical isolator or an electro-optic (Solid State Relay) isolator
This board is not intended to drive a 115VAC Reader Step teletype signal directly.

PARALLEL I/O CAPACITY

PIO Chips: 2
8-Bit Ports: 4 (total)
Handshake lines per port: 2
Total lines: 4x10=40

TIMERS

CTC chips: 1
Channels: 4
Each channel has a prescale counter driving an 8 bit (256 state) down counter

Input Sources:
Counter mode: External input directly into down counter
Timer mode: Internal system clock driving prescale counter

Prescale Counter: ÷16 or ÷256
Down Counter: ÷1 thru ÷256

INPUT MODE	FUNCTION	SINGLE CHANNEL min	SINGLE CHANNEL max
Counter	Input frequency	dc	1/2 system clock (1.2288MHz)
	Programmable One-shot	813.6 ns	∞
	Real time interrupt	813.6 ns	∞
	rate generator	dc	1.228MHz
Timer	Input Frequency	System Clock	
	Programmable one-shot	6.51 us	26.66 ms
	Real time interrupt	6.51 us	26.66 ms
	Rate Generator	37.5 Hz	153.60 KHz

Multiple channels can be cascaded for longer counting intervals.

SPECIFICATIONS

Interfaces

BUS: All signals TTL compatible
Parallel I/O: All signals TTL compatible
Interrupt Requests: All signals TTL compatible
Counter/Timer: All signals TTL compatible
Serial I/O: All signals RS-232 or 20mA loop compatible

Drivers and Terminators

The following components are compatible with sockets at the appropriate location

	TYPE	OUTPUT	SINK CURRENT (mA)
Address & Data Bus:	DM8833	NI Tri-State Bi-directional	50
	DM8835	I Tri-State Bi-directional	50
Control: WRB, RDB, IORQB, RFSHB			
	DM8833	NI Tri-State Bi-directional	50
	DM8835	I Tri-State Bi-directional	50
: MREQB, MIB, HALTB, BNSAKB			
	DM8097	NI	32

	Type	Output	Current (MA)
I/O:	Ports D0, D4		
	DM8833	NI	50
		Tri-state Bidirectional	50
	DM8835	I	
		Tri-state Bidirectional	50
	Ports D2, D6		
	7400	I	16
	7403	I,OC	16
	7408	NI	16
	7409	NI,OC	16
	7426	I,OC,HV	16
	7432	NI	16
	7437	I	48
	7438	I,OC	48
	7486	I/NI	16
Handshake: RDY			
	7486	I/NI	47Ω series resistor

Terminators:
220 Ω/330 Ω divider or 1K Ω pullup resistors.

Connectors

DESIGNATION	NUMBER OF PINS	P/N
SK1, SK2	64	ELCO #00-8257-096-000-524

NOTES:

MKXXXX = Supplied by MOSTEK

NI = Non Inv

I = Inver

OC = Open Connector

SPECIFICATIONS

Power Supplies (Current measured with no peripherals connected to the board. The board is populated with 16K RAMS, 4 PROMS - 2708 and 1 ROM - 34000).

	Max	Typ
V_{DD} = +12V $\pm$ 5%	@ 480 mA	180 mA
V_{CC} = + 5V $\pm$ 5%	@ 2.6 A	1.7 A
GND		
V_{AA} = -12V $\pm$ 5%	@-180 mA	-90mA

-5V is regulated down
from -12V on card
Under no circumstznces is it permitted to
connect 115 VAC to this board!

SDB-80 BOARD OUTLINE

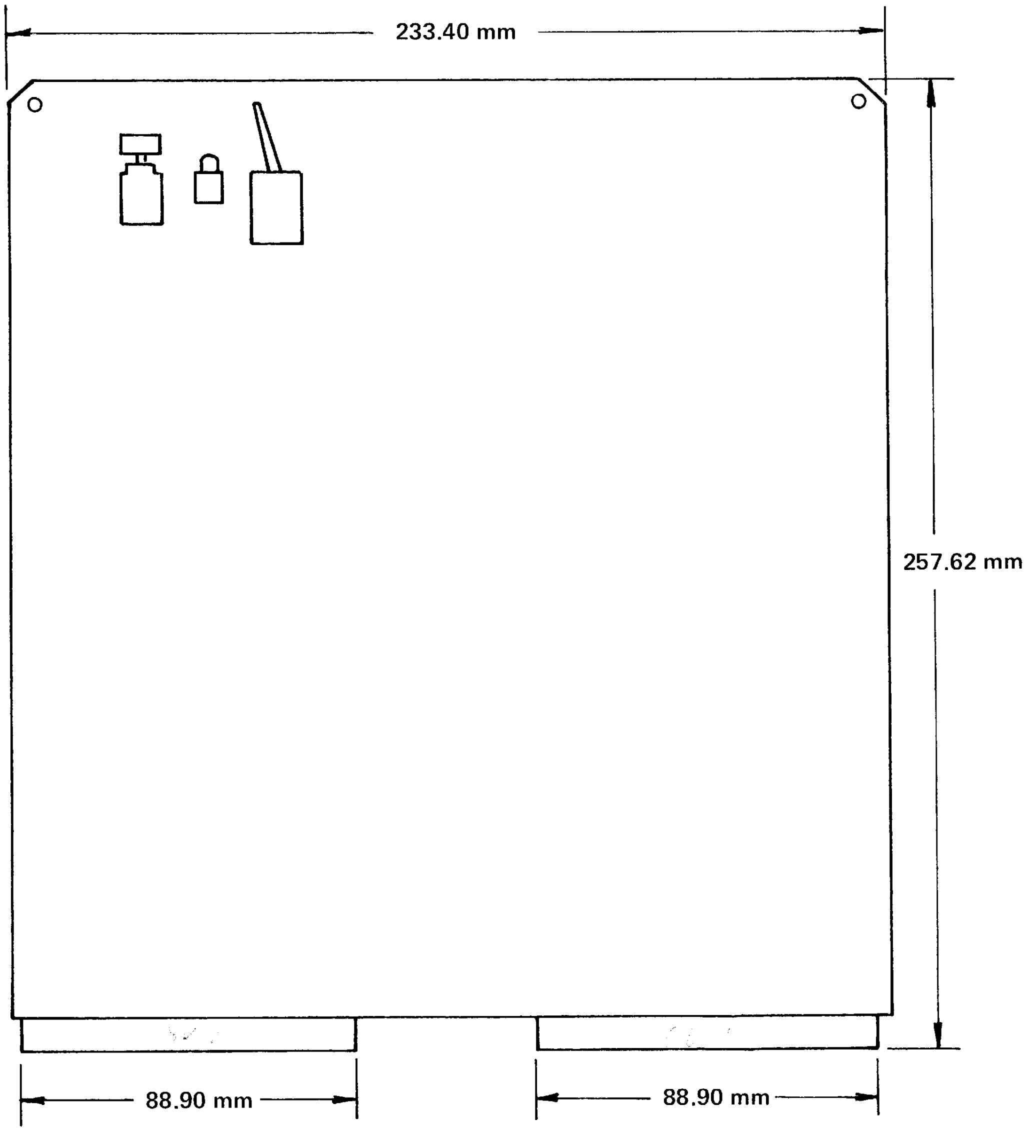


Figure 1

SDB-80 CONNECTOR PIN OUTS

SK 1

c	Pin	a
GND	1	GND
GND	2	GND
−12 V	3	−12 V
+ 5 V	4	+ 5 V
+ 5 V	5	+ 5 V
+ 12 V	6	+ 12 V
*	7	*
IEIB	8	IEOB
BAI	9	$\overline{\text{BUSAKB}}$ (BAO)
	10	$\overline{\text{DEBUGB}}$
$\overline{\text{MEMDISB}}$	11	$\overline{\text{RESETB}}$
$\overline{\text{MIB}}$	12	$\overline{\text{RESTARTB}}$
$\overline{\text{NMIB}}$	13	$\overline{\text{MREQB}}$
$\overline{\text{BUSRQB}}$	14	ΦENAB
$\overline{\text{HALTB}}$	15	$\overline{\text{INTB}}$
$\overline{\text{RDB}}$	16	$\overline{\text{WAITB}}$
AOB	17	ΦB
A1B	18	$\overline{\text{IORQB}}$
A2B	19	$\overline{\text{WRB}}$
A3B	20	$\overline{\text{RFSHB}}$
A4B	21	$\overline{\text{DRIVE B}}$
A5B	22	$\overline{\text{DINB}}$
A6B	23	EXTBRB
A7B	24	$\overline{\text{XPNDB}}$
A8B	25	DOB
A9B	26	D1B
A10B	27	D2B
A11B	28	D3B
A12B	29	D4B
A13B	30	D5B
A14B	31	D6B
A15B	32	D7B

*Reserved for −5V

SK2

c	Pin	a
GND	1	GND
GND	2	GND
RECEIVE DATA AT TERMINAL (RS–232)	3	$\overline{\text{CK/TG3B}}$
$\overline{\text{ZC/TOIB}}$	4	$\overline{\text{CK/TG1B}}$
$\overline{\text{ZC/TO2B}}$	5	$\overline{\text{CK/TG2B}}$
DATA SET READY	6	DATA TERMINAL READY
CLEAR TO SEND	7	REQUEST TO SEND
READER STEP +	8	READER STEP –
20mA LOOP–(RECEIVE FROM TERM.)	9	20mA LOOP +(RECEIVE FROM TERM.)
20mA LOOP + SEND	10	20mA LOOP – SEND
$\overline{\text{STBD6}}$	11	RDYD6
P(D6) 3	12	P (D6) 4
P(D6) 2	13	P (D6) 5
P(D6) 1	14	P (D6) 6
P(D6) O	15	P (D6) 7
$\overline{\text{STBD4}}$	16	RDYD4
P (D4) 3	17	P (D4) 4
P(D4) 2	18	P (D4) 5
P (D4) 1	19	P (D4) 6
P (D4) O	20	P (D4) 7
	21	
	22	
$\overline{\text{STBD2}}$	23	RDYD2
P (D2) 3	24	P (D2) 4
P (D2) 2	25	P (D2) 5
P (D2) 1	26	P (D2) 6
P (D2) O	27	P (D2) 7
$\overline{\text{STBDO}}$	28	RDYDO
P (DO) 3	29	P (DO) 4
P (DO) 2	30	P (DO) 5
P (DO) 1	31	P (DO) 6
P (DO) O	32	P (DO) 7

A-1 INC. DCN # 52670

ITEM	QTY.	PART NO.	DESCRIPTION	REF. DESIG.	NOTES
79	1	4210054	BUSSBAR (X=6)		
78	1	4210055	BUSSBAR (X=8)		
77	1	4210056	BUSSBAR (X=11)		
76	1	4640022	SWITCH TOGGLE	S2	
75	1	4640021	SWITCH P B	S1	
74	2	4140003	EJECTOR		
73	2	4210057	CONNECTOR	SK1-2	
72	105	4210001	WIREWRAP TERMINAL	E1-101,TP1-4	
71	1	4620038	SOCKET 28 PIN	X38	
70	1	4620037	SOCKET 22 PIN	X39	
69	4	4620019	SOCKET 40 PIN	X34,37,40,48	
68	5	4620018	SOCKET 24 PIN	X24-28	
67	14	4620017	SOCKET 16 PIN	49,51,60,62 X10-17,31,32,	
66	8	4620016	SOCKET 14 PIN	63,72-75 X50,52,61,	
65	1	4230007	CRYSTAL 4.916 MHZ	Y1	
64	5	4150114	CAP. 68 M.F.	43 C5,6,33,42,	
63	44	4150111	CAP. .1 M.F.	34-37,39,44-56 C1,4,7,10-32,	
62	1	4150107	CAP. .01 M.F.	C38	
61	2	4150101	CAP. 1000 P.F.	C2,3	
60	1	4150083	CAP. 10 P.F.	C40	
59	1	4150086	CAP. 33 P.F.	C41	
58	2	4150078	CAP. 1.0 M.F.	C8,9	
57	1	4480026	DIODE IN914	CR2	
56	1	4240011	LED PC MNT.	CR1	
55	1	4470179	RES. PKG. 1.K SIP 8 PIN	U43	
54	1	4470178	RES. PKG. 1K SIP 6 PIN	U6	
53	4	4470176	RES. PKG. 1K 14 PIN DIP	U72-75	
52	2	4470175	RES. PKG. 4.7K SIP 10 PIN	U44,46	
51	2	4470174	RES. PKG. 1K SIP 10 PIN	U23,33	
50	1	4470182	RES. 22Ω 1W	R7	
49	3	4470097	RES. 10K 1/4W	R1,4,5	
48	1	4470084	RES. 3K 1/4W	R25	
47	1	4470075	RES. 1.2K 1/4W	R12	
46	6	4470073	RES. 1K 1/4W	28 R3,8,19-21,	
45	2	4470071	RES. 820Ω 1/4W	R10,11	
44	7	4470041	RES. 47Ω 1/4W	26 R15-18,23,24,	

ITEM	QTY.	PART NO.	DESCRIPTION	REF. DESIG.	NOTES
43	4	4470057	RES. 220Ω 1/4W	R6,13,22,27	
42	1	4470033	RES. 22Ω 1/4W	R14	
41	1	4470089	RES. 4.7K 1/4W	R9	
40	1	4470093	RES. 6.8K 1/4W	R2	
39	1	4480011	TRAN. 2N3906	Q1	
38	1	4313310	IC MK 34070	U28	
37	1		IC MK 3400D	U27	
36	1		IC MK 3400C	U26	
35	1		IC MK 3400 B	U25	
34	1		IC MK 3400A	U24	
33	1	4313274	IC 74132	U4	
32	1	4313271	IC MK 3880	U48	
31	2	4313270	IC MK 3881	U36,37	
30	1	4313269	IC MK 3882	U38	
29	1	4313268	IC 35392	U39	
28	3	4313267	IC 74S32	U5,8,70	
27	1	4313266	IC 74S74	U21	
26	1	4313265	IC 81LS97	U1	
25	2	4313264	IC 74S157	U18,19	
24	1	4313282	IC MK 6260	U31	
23	1	4313281	IC MK 6261	U32	
22	1	4313261	IC 74LS75	U69	
21	11	4313260	IC DM 8833	62,67,68,71 U49,51,56-60	
20	2	4313259	IC 75150P 8 PIN	U64,77	
19	1	4313258	IC RC 1489	U55	
18	8	4313246	IC MK 4027-4	U10-17	
17	1	4313223	IC LM 320H-5	U78	
16	1	4313220	IC 74221	U2	
15	1	4313215	IC 75451BP 8 PIN	U76	
14	1	4313181	IC AY-5-1013	U40	
13	1	4313170	IC 7414	U20	
12	1	4313169	IC 8098	U47	
11	2	4313154	IC 8097	U34,42	
10	2	4313116	IC 74155	U29,30	
9	1	4313075	IC 74175	U65	
8	2	4313035	IC 7486	U45,53	
7	1	4313032	IC 7474	U3	
6	7	4313010	IC 7408	52,61,63 U7,22,35,50,	
5	1	4313006	IC 7404	U54	
4	1	4313284	IC 74S00	U9	
3	1	4313285	IC 74S04	U41	
2	1	4610038	PCB (450-00179-00)		
1	REF.	450-00181-00	SCHEMATIC		

USED ON	NEXT ASSY
MICROPROCESSOR	

DRN T. BARNETT 3/72
CHKD J. Potts

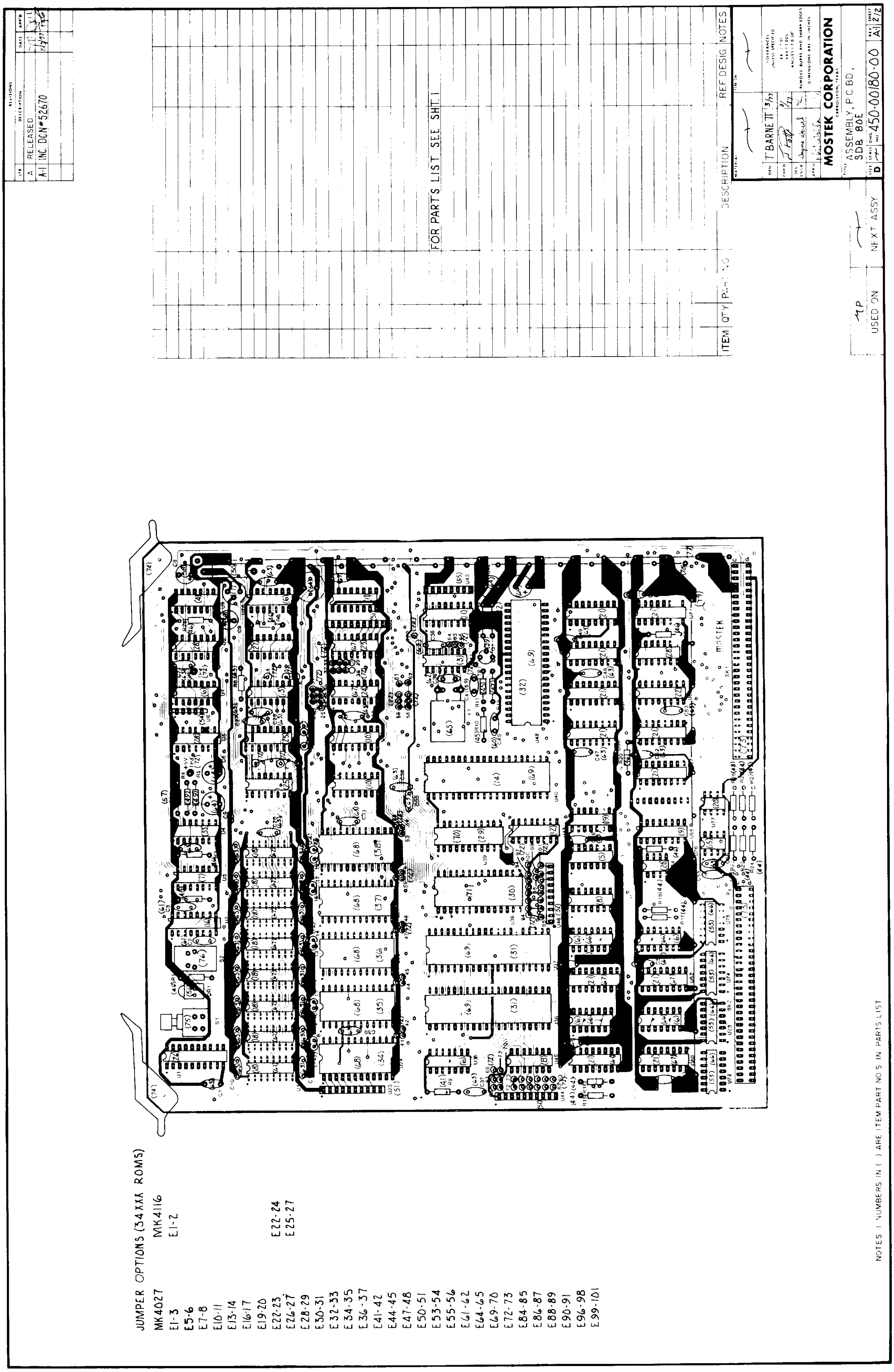
MOSTEK CORPORATION
CARROLLTON, TEXAS

TITLE ASSEMBLY, P.C. BD., SDB 80 E

SIZE D DWG NO 450-00180-00 REV A-1 SHEET 1/2

NOTES: 1. NUMBERS IN ( ) ARE ITEM PART NO.S IN PARTS LIST.

Figure 1



REVISIONS
A RELEASED
A-1 INC. DCN#52670
FOR PART'S LIST SEE SHT. 1
ITEM QTY PART NO. DESCRIPTION REF DESIG NOTES
T BARNETT
MOSTEK CORPORATION
ASSEMBLY, P C BD, SDB 80E
450-00180-00
USED ON NEXT ASSY
MOSTEK
JUMPER OPTIONS (34XXX ROMS)
MK4027 MK4116
E1-3 E1-2
E5-6
E7-8
E10-11
E13-14
E16-17
E19-20
E22-23 E22-24
E26-27 E25-27
E28-29
E30-31
E32-33
E34-35
E36-37
E41-42
E44-45
E47-48
E50-51
E53-54
E55-56
E61-62
E64-65
E69-70
E72-73
E84-85
E86-87
E88-89
E90-91
E96-98
E99-101
NOTES: 1. NUMBERS IN ( ) ARE ITEM PART NO'S IN PARTS LIST

Figure 2

MOSTEK®

EUROPEAN MARKETING OFFICES

MOSTEK INTERNATIONAL
150 CHAUSSÉE DE LA HULPE
B 1170 BRUSSELS
BELGIUM
02/6 60 25 68
TELEX 62 011 MK BRU

GERMANY
MOSTEK GMBH
TALSTRASSE 172
7024 FILDERSTADT 1
07 11/70 10 96
TELEX 07 255 792

FRANCE
MOSTEK FRANCE S.A.R.L.
1 PLACE DES ETATS-UNIS
SILIC 217
F 94 518 RUNGIS-CEDEX
01/686-0153
TELEX 204 049

UNITED KINGDOM
MOSTEK UK LIMITED
MASONS HOUSE
1-3 VALLEY DRIVE
KINGSBURY ROAD
LONDON NW 9
01-204 9322
TELEX 25 940

ITALY
MOSTEK ITALIA S.P.A.
VIA G. DA PROCIDA 10
I-20 149 MILANO
02/318 5337
TELEX 25 601

PRINTED IN USA December 1977
Publication No. MK 78548