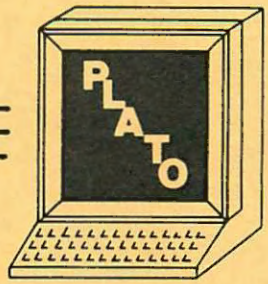
Computer-based Education
Research Laboratory



University of Illinois Urbana Illinois

CERL Report X-50 August 1977

THE PLATO® V TERMINAL

J.E. STIFLE

THE PLATO® V TERMINAL

J.E. Stifle

Computer-based Education Research Laboratory

University of Illinois, Urbana, Illinois 61801

ABSTRACT

This report describes the architecture and programming of the PLATO V terminal. This terminal contains an 8080 microprocessor and is capable of being operated by programs resident in the terminal or by programs located in a host computer. The terminal contains 8k of memory for storing local programs, a 4k ROM resident program which supervises terminal operation, a 2k ROM character set and 2k of spare ROM memory.

Acknowledgments

The development of this terminal was due in large part to the efforts of several people.

Mike Hightower and Ron Klass played a major role in the development of the terminal firmware and the design of the ROM programming system. Leonard Hedges performed the layout of the processor assembly and supervised the construction of the terminal prototypes.

A special mention is due Donald Lee, who wrote an 8080 assembler which greatly simplified and speeded the development of the terminal firmware.

Bruce Sherwood helped define the external IO protocol and Paul Tenczar made a significant improvement in the character plotting program.

A special thanks is due Joyce Lipschutz, who very patiently did all of the typing and editing of this report, and to Roy Lipschutz, who did all of the drawing.

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

1. TERMINAL ARCHITECTURE

1.0 General

The PLATO V terminal is actually a micro computer system containing all of the standard features of large computer systems including a processing unit, memory, and an input-output (IO) structure. Attaching a plasma panel to the IO structure and loading the appropriate program in memory will make this system behave as a PLATO terminal. As with any computer system, the contents of the memory determine the system operating characteristics. This programmable feature enables the terminal to be operated as a standard PLATO IV terminal or as a more powerful PLATO V terminal with enhanced graphics capabilities.

This terminal may be operated by programs located in the central computer, as in PLATO IV, or by local programs residing in the terminal memory. The latter may be loaded into the terminal in much the same manner as character set data is loaded, or they may be loaded from floppy disks or other storage devices attached to the terminal. The terminal is also capable of being operated in a time shared mode between local and centrally resident programs.

It is this local programming feature which dramatically expands the powerful and unique graphics capabilities of the PLATO system. Included in these graphics enhancements are:

- A character plotting rate in excess of 3000 characters per second.
- Character plotting in both horizontal and vertical modes.
- Character plotting in two directions in both modes.
- Character magnification.
- A 6000 character per second selective block erase function.

A block diagram of the terminal is shown in Figure 1.0. Operation of the terminal is under the supervision and control of an 8080 microprocessor constructed on a single LSI chip. This chip is an 8-bit parallel processor operating with a 2μs instruction time containing 7 internal working registers.

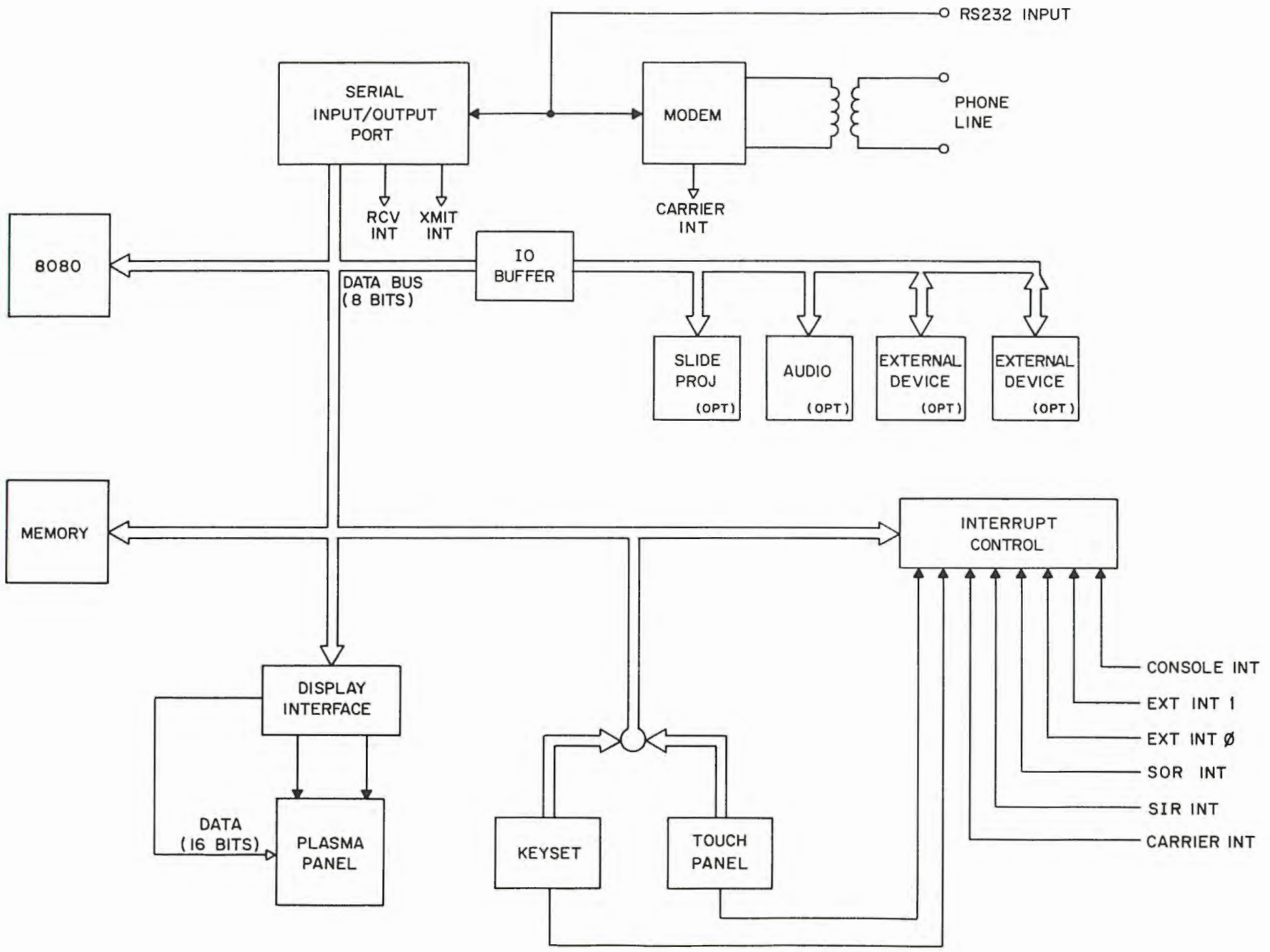


Figure 1.0. Terminal Block Diagram

An 8-bit bidirectional data bus connects the 8080 to the other functional units within the terminal and to IO devices external to the terminal. The data bus acts as a highway over which information flows between internal (and external) portions of the terminal. In addition to performing display generation functions, the 8080 must manage the flow of information on this highway. Also present but not shown is a 16-bit address bus which is used to specify memory and input-output device addresses.

This terminal operates as an interrupt driven device; i.e., all activity occurs as a result of an interrupt requesting service.

All service requests to the 8080 are made via the Interrupt Control Unit (ICU). A device requesting service originates an interrupt and presents it to the ICU. Within the ICU the interrupt requests have a wired-in priority, and the ICU will pass to the 8080 the interrupt request having the highest priority.

When an interrupt request is accepted by the 8080, normal program sequencing is halted; the present contents of the program counter (PC) are pushed into the stack in memory; and an RST (unconditional jump) instruction to location 70 is forced into the instruction register. Following the interrupt, the 8080 reads a word from the ICU which contains the address of the interrupting source. Program control is then transferred to the processing routine for the interrupting source.

Within the 8080 is an interrupt enable flag which must be set before any interrupt requests will be accepted from the ICU. This flag can be set by the EI (enable interrupt) and reset by the DI (disable interrupt) instructions. Each time an interrupt is accepted by the 8080, this flag is automatically reset, thus disabling further interrupts until set by an EI instruction.

The ICU contains an 8-bit Interrupt Mask Register, each bit of which is associated with an interrupt source. An interrupt is enabled if the associated bit in the Mask Register is a "one," disabled if it is a "zero." An interrupt will not be passed through the ICU unless it is enabled.

The 8080 can, therefore, selectively enable or disable interrupts by the data loaded into the Mask Register. The data to be loaded into the Mask Register is maintained in a protected location in memory. An interrupt is said to be 'armed' if the associated bit in this location is a "one," 'disarmed' if it is a "zero." The resident program can selectively set (arm) or clear (disarm) bits in this word before sending it to the Mask Register. Thus, an interrupt may be armed, but temporarily disabled by the resident.

Figure 1.1 and 1.2 contain the flow charts describing the processing for each type of interrupt. In these diagrams the symbol RTN means return to the program in progress when the interrupt occurred; SAVE means save the present contents of the registers and flags in the stack. In performing RTN the saved information will be restored to the registers, and interrupts will be enabled as they are then armed.

The ICU provides for eight interrupt sources with the priority and addresses shown in Table 1.

1.1 Serial IO (SIR and SOR)

PLATO data from the central computer arrives as 21-bit words at a rate of 57.14 words per second (1200 bps). The format of this data is described in section 2.0. After each 8-bit byte of input data is received, the SIR interrupt is generated to indicate the arrival of a new byte.

The 8080 responds to the interrupt by reading the data byte and storing it in memory. After three bytes have been received, the data is assembled into a PLATO job (a job is a PLATO word) and placed in the job stack. The job stack is a section of memory reserved for storing incoming PLATO jobs in the event they arrive while the terminal is busy. The job stack can hold up to 1.48 seconds of PLATO output (85 jobs).

The SOR interrupt indicates that the transmitter section of the serial IO port is available for use. The 8080 responds by loading any data awaiting transmission or, if no data is waiting, by disabling the SOR interrupt.

In addition to data, the serial IO port contains three IO status flags: the Lost Data flag, the XMIT Ready flag, and the RCV Ready flag. The Lost Data flag indicates that the 8080 failed to input one or more previously received words. The XMIT Ready flag indicates the present status of the transmitter. If this bit is "zero," the transmitter is busy transmitting data; while if it is "one," the transmitter is available for use. Before loading data into the transmitter, the 8080 examines this flag to determine availability. The RCV Ready flag indicates the arrival of an 8-bit byte in the port.

Request	Interrupt Address (hex)
SIR highest priority	00
KST	01
TP	02
SOR	03
EXT0	04
EXT1	05
CONSOLE	06
CARRIER	07

Table 1. Interrupt Priority

1.2 Abort Mode

The 8080 maintains a record (word count) of the number of non-NOP (described in section 2.1) words received. Each time a non-NOP word is transferred into the 8080, the word count is incremented by 1. Upon receipt of a word containing a parity error or an indication of lost data, the 8080 automatically transmits the value of the word count to the computer center, sets the ABORT flag and enters the ABORT mode of operation. The value of the word count transmitted will indicate to the center the address of the word containing the error or the word that was lost.

The ABORT Mode flag indicates the error mode status of the terminal. If this flag is "zero," the terminal is operating normally; while if it is "one," the terminal is in the ABORT mode of operation. A red

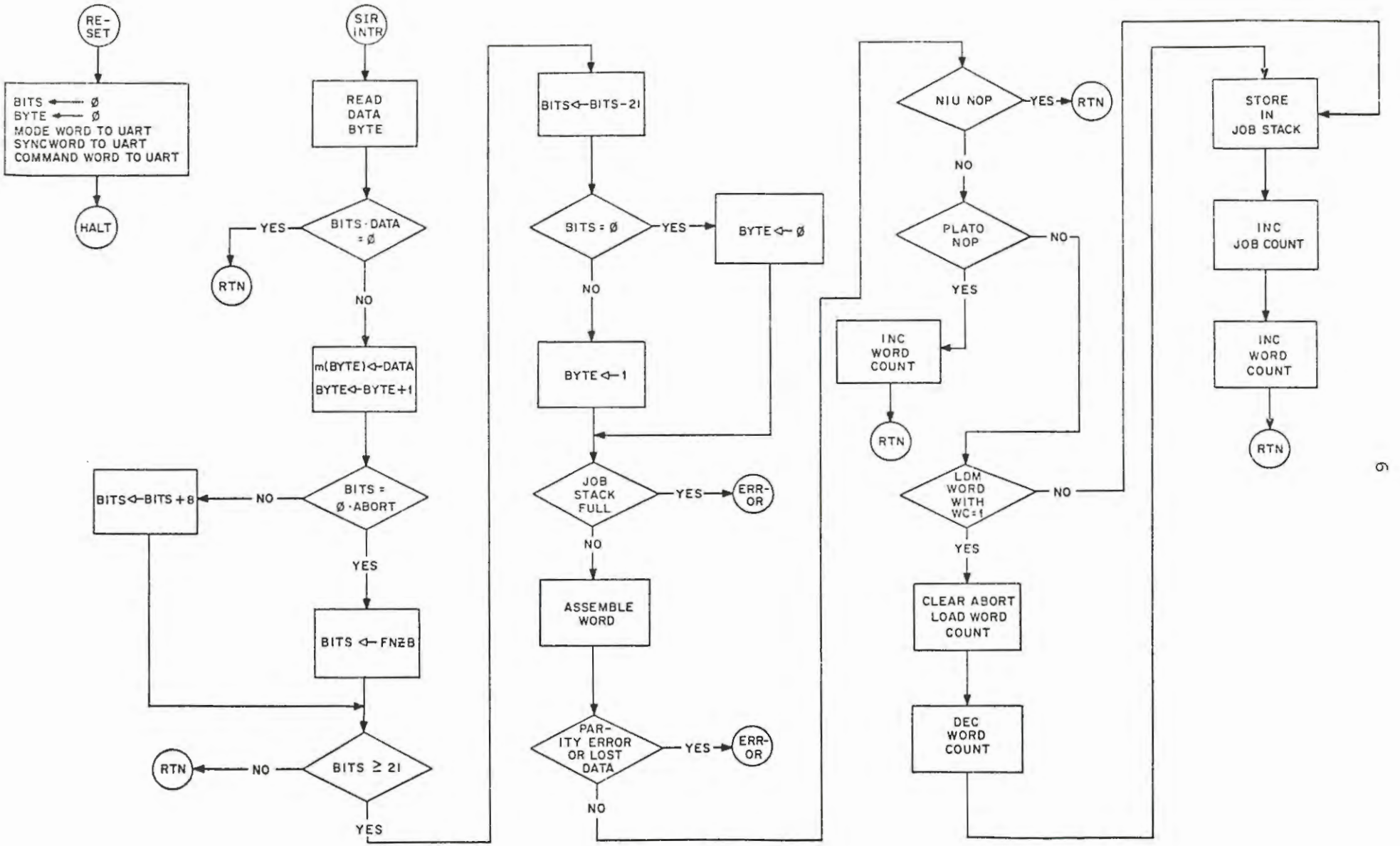


Figure 1.1. SIR Flow Chart

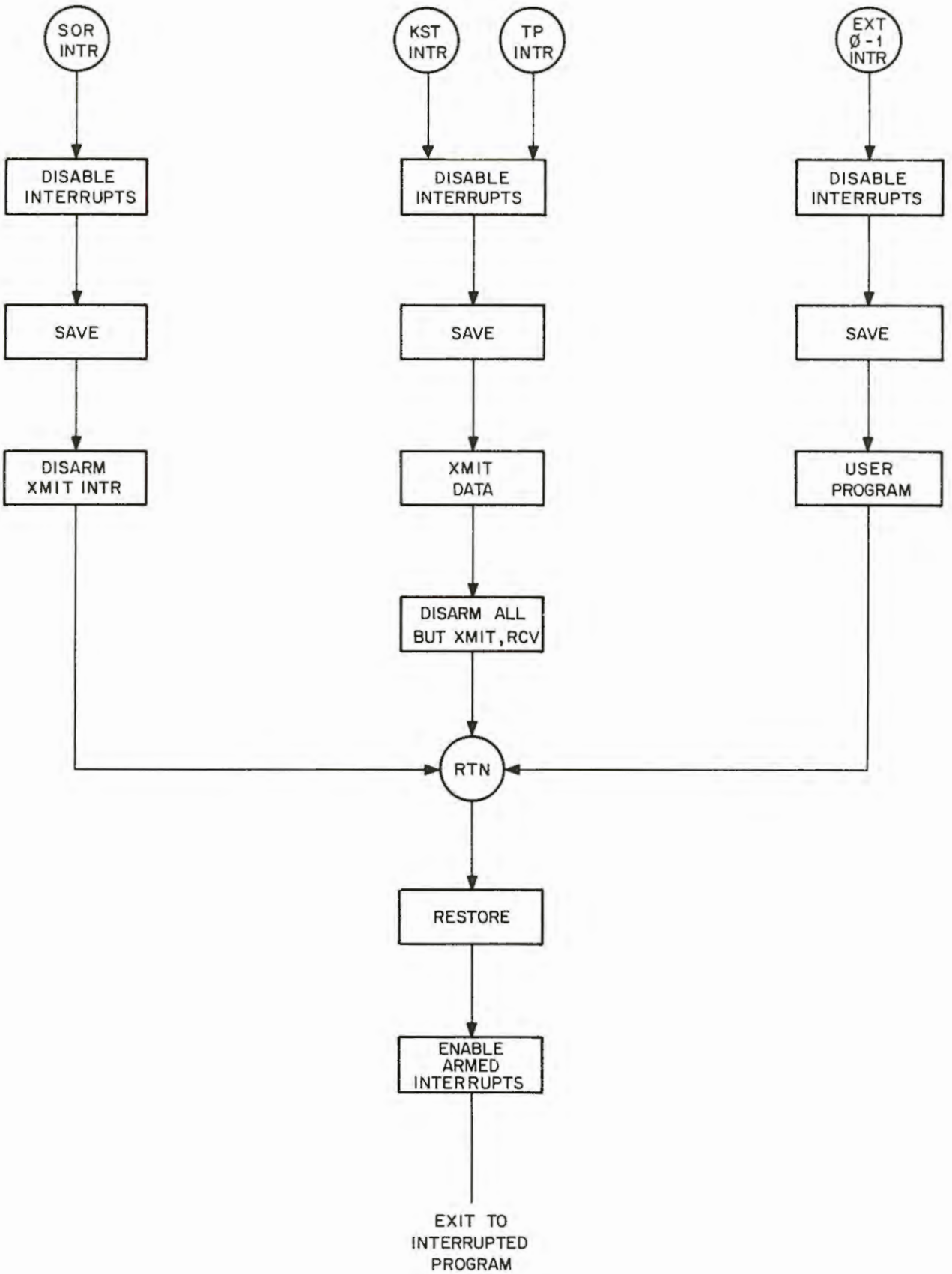


Figure 1.2. SOR, KST, TP, EXT Interrupts

indicator located on the front of the terminal indicates the status of the ABORT flag.

Once in the ABORT mode, the terminal will refuse to accept any further information except for an LDM instruction (described in section 2.1). Receipt of an LDM instruction with bit 14, a "one" will clear the ABORT flag and return the terminal to normal operating mode. This method of error control prevents the terminal from processing data in the wrong mode in the event an erroneous mode change word is received.

Overflowing the job stack will also cause the terminal to enter the ABORT mode. In this case the offending word is treated as though it arrived containing an error.

1.3 Keyset (KST)

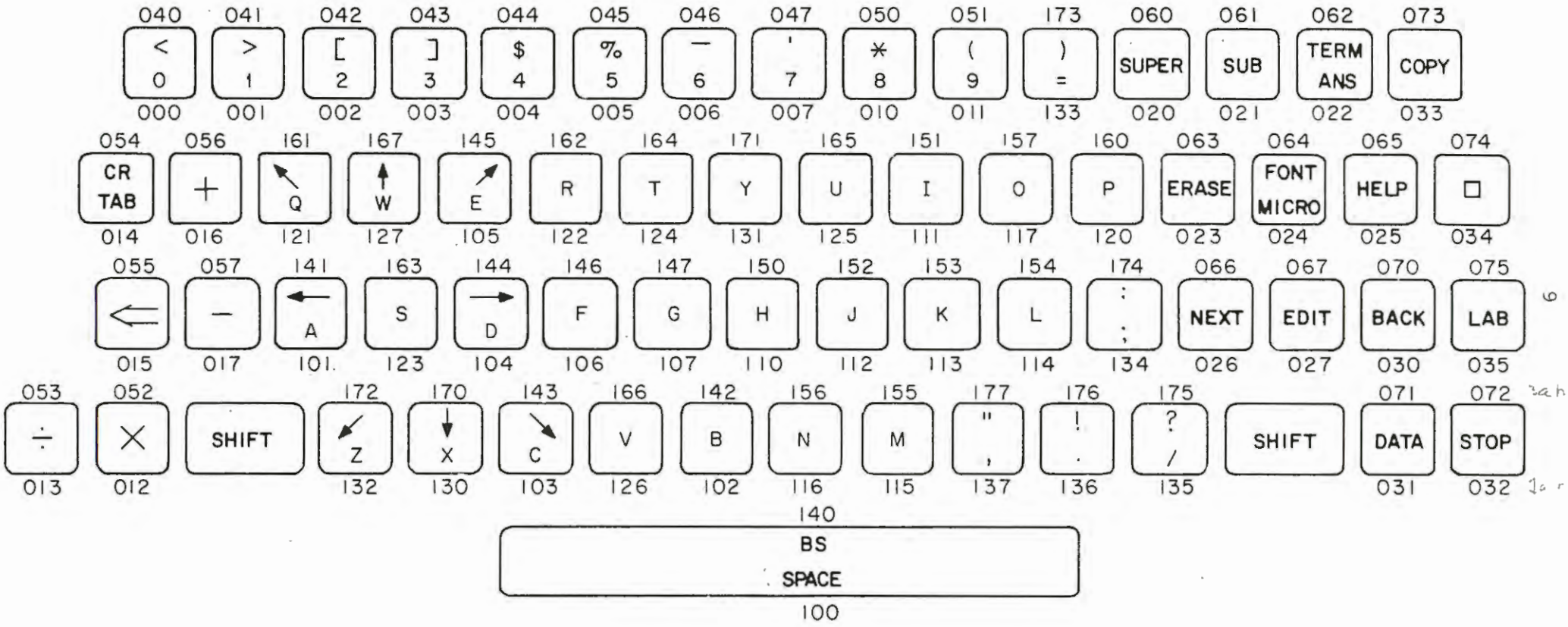
The KST interrupt indicates that a key has been pushed on the keyset. The layout and coding of the keyset are shown in Figure 1.3.

1.4 Touch Panel (TP)

The Touch Panel is an input device which allows the terminal user to touch the display surface and input positional information directly to the microprocessor. The touch sensitive surface is a 16 x 16 array of squares and the TP interrupt is generated whenever any square is touched. A short audio tone is generated each time the TP interrupt is accepted by the microprocessor.

1.5 External Input-Output (EXTØ)

The EXTØ interrupt indicates that a device attached to the external data bus is requesting service. The terminal provides for the attachment to the IO bus of up to 32 input and 32 output devices. Such devices, all optional, include a random-access slide projector for the projection of slide images on the rear of the plasma panel, a random-access audio unit which can play back to the terminal user pre-recorded audio messages, a ROM programmer, and a floppy disk system. Other user-defined output devices may also be attached. Data rates in excess of 25K bytes per second may easily be accomplished on the IO Bus.



Note

1. Each key has two different inputs. The octal number below the box is the input when a key is pressed singly (normal state), and the number above the box is the input when the "Shift" key is held down as a key is pressed (shift state).

   The "Shift" key alone does not initiate input data transfer, but merely causes an addition of 040 (octal) to a normal input.
2. There is a total of 124 different inputs.
3. The input codes 036, 037, 076, and 077 are not used.

Figure 1.3. Keyset Coding

The EXT 1 interrupt is used internally by the terminal.

1.6 Console Mode (CONSOLE)

Contained within the resident program is a routine which permits the keyboard and display to be used as program debugging aids. To enter this routine, the RUN-CONSOLE switch must be placed in the CONSOLE position.

In CONSOLE mode the user may display the contents of the 8080 registers, the contents of memory, alter the contents of memory, step through programs one instruction at a step, set a breakpoint, and jump to other programs. This feature is described in detail in section 3.3.

1.7 Carrier Interrupt (CARRIER)

This signal is used to indicate an interruption of communication with the central computer. The resident program will generate a message on the display indicating loss of communication. (Use of this feature requires a modem with carrier detect capability.)

1.8 Display Interface Unit (DIU)

The DIU contains the registers and control circuits required to efficiently attach a plasma panel to the data bus. The DIU, shown in Figure 1.4, contains the 9-bit x and y display address registers, a 16-bit parallel data register (PDL/U), a 3-bit display mode (PDM) register and the write-erase control circuits. Data for the registers and control information is supplied to the DIU from the data bus. The x and y registers are bidirectional counters which can be independently controlled by the 8080. The parallel data register consists of the 8-bit parallel data upper (PDU) and parallel data lower (PDL) registers. The PDU/L registers are used only when operating parallel input display devices.

The format of the PDM register is shown in Figure 1.5. Bits 0 and 1 specify the write/erase mode, and bit 2, the panel operating mode. If bit 2 is "zero," the display is operating in the serial mode; and the contents of the x and y registers specify the address of a point to be

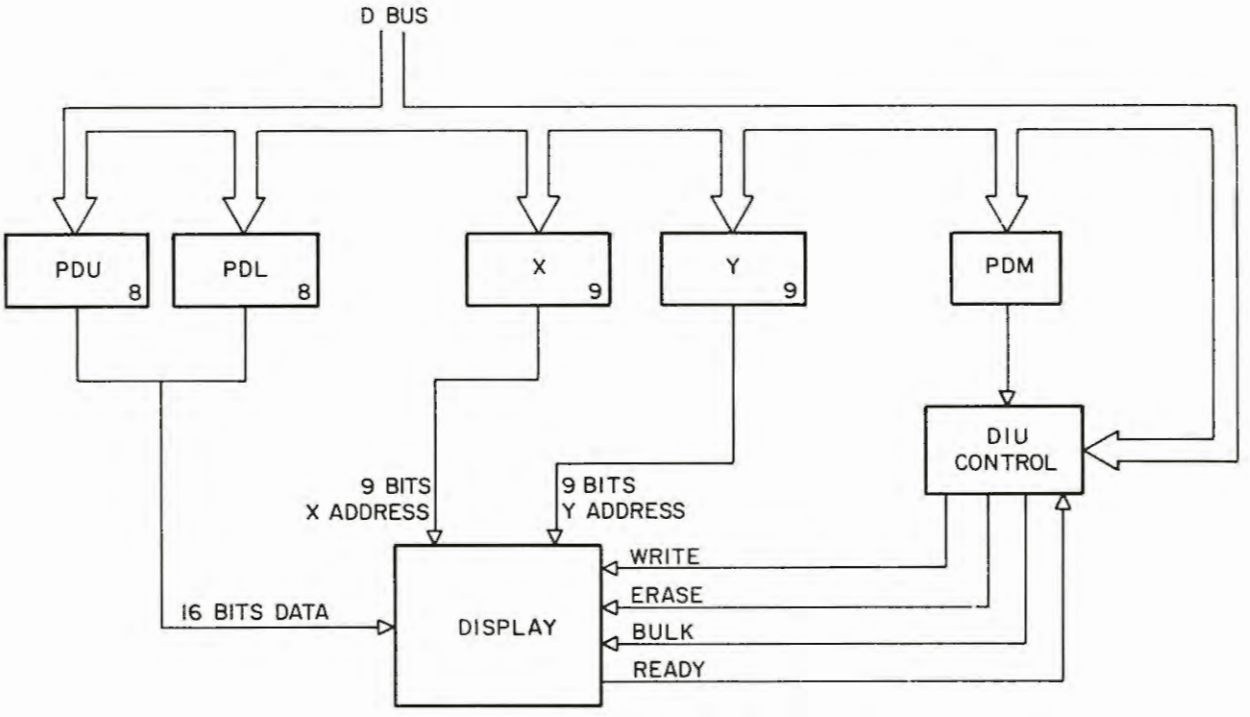


Figure 1.4 Display Interface Unit

written or erased. If bit 2 is "one," the display is operating in the parallel mode; and the contents of PDL/U will be written or erased on the panel at the address specified by the contents of the x and y registers. The data will be displayed in a vertical column with bit 0 of PDL at the bottom and bit 7 of PDU at the top.

Information is written on the display if $WE_1 = 1$ or erased of $WE_0 = 0$. The use of these bits is explained in section 2.5 in the discussion of Mode 3.

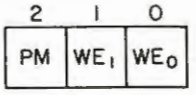


Figure 1.5, PDM Register

1.9 Memory

The terminal memory consists of 16k (k = 1024) words, half of which is ROM and half RAM. The ROM portion of memory contains the resident program and the data for the standard PLATO character set. The resident provides all of the programs necessary to process PLATO data plus programs to service all interrupts. All graphical and IO routines in the resident are callable and may be accessed by user programs located in RAM.

More will be said about the resident in section 3.

2. OPERATING MODES

2.0 PLATO Word Format

The data to be processed by the terminal consists of 20-bit words (with start bit removed) with the format shown in Figure 2.0.

Figure 2.0. Terminal Word Format

Bit 00	Parity bit - even parity
Bits 01 - 18	Data
Bit 19	Control bit - 0 = control word - 1 = data word

Terminal words may be of two types: control words and/or data words. Data words (c = 1) contain the data to be processed by the terminal while control words (c = 0) are instructions used to establish operating conditions within the terminal.

2.1 Control Word Formats

The PLATO control word format is shown in Figure 2.1.

Figure 2.1. Control Word Format

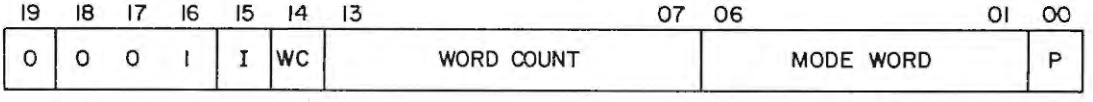
Bits 01 - 15	Control Information
Bits 16 - 18	Type of Control Word

D = 000 (NOP)

This word is a NOP (no-operation instruction). There are two types of NOP words, those generated by the PLATO communications hardware (bit 01 = 0) and those generated by the PLATO software (bit 01 = 1). The hardware NOP is generated automatically when the central computer has no data to be transmitted to the terminal. The software NOP can be used by system software to insert timing delays in the output data stream.

The software NOP will cause the terminal word count to be incremented while the hardware NOP will not affect the terminal status in any way. Neither of the NOPs is stored in the job stack.

D = 001 (LDM) Load Mode (267μs)



This instruction establishes the operating mode of the terminal. For each mode of terminal operation, there is an associated mode word (bits 01 - 06) which directs the processing of incoming data. Once placed in a given mode, the terminal remains in that mode until receipt of a new LDM instruction.

Eight different processing modes are available, five of which are incorporated in the terminal resident and three of which are reserved for local programs. The processing modes are described later in this section.

If bit 14 (WC) of the LDM word is "one," the word count register will be set to the value specified by bits 07 - 13. It is the receipt of this instruction with bit 14 set which will restore the terminal to normal mode if it is in the ABORT mode. This is the only instruction which the terminal will accept if it is in ABORT mode. Receipt of the LDM instruction while in the ABORT mode will clear the ABORT flag and initialize the word count, but will not alter the terminal processing.

Bit 15 of the LDM word is used to actuate or inhibit external

devices attached to the terminal. Receipt of an LDM word with bit 15 a "one" will disable the TP and EXT0 interrupt.

The terminal mode word format is shown in Figure 2.2.

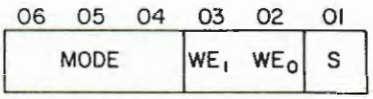


Figure 2.2. Mode Word Format

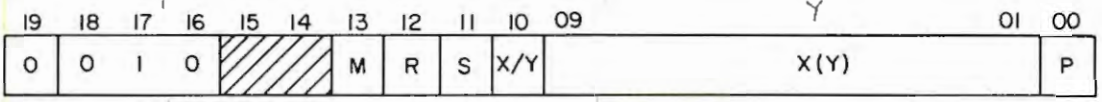
Bit 1 — Screen Command. If this bit is "1," the entire display is erased.

Bits 02 - 03 — Select write or erase function in the DIU as follows:

WE_1	WE_0	
0	0	Erase, write character background.
0	1	Write, erase character background.
1	0	Erase, suppress character background write.
1	1	Write, suppress character background erase.

Bits 04 - 06 — Specify terminal processing mode.

D = 010 (LDC) Load Co-ordinate (225µs)

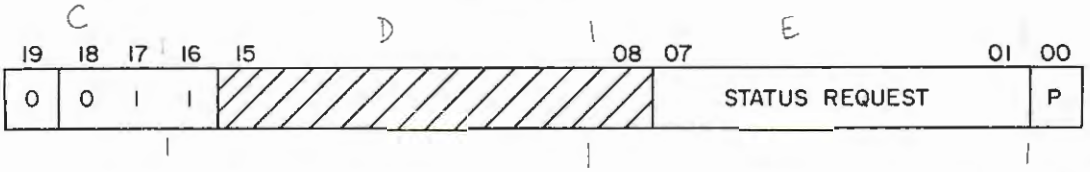


This instruction loads the x register (bit 10 = 0) or the y register (bit 10 = 1) with data as follows:

R	S	
0	x	Load register with bits 01 - 09.
1	0	Load register with the arithmetic sum of its present value and bits 01 - 09.
1	1	Load register with the arithmetic difference between its present value and bits 01 - 09.

In addition, if M (bit 13) = 1, the resultant value is also stored in memory location m.margin. All carriage returns performed by the terminal will set the x register (y register if plotting vertically) to the value contained in m.margin.

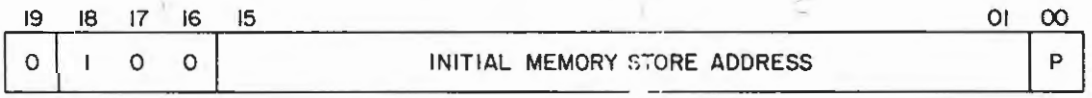
D = 011 (LDE) Status Request (267μs)



This word is used to request information from the terminal or to request certain operations to be performed. Presently used codes are:

Code (hex)	Operation
70	Request terminal type. Terminal responds with code 73.
7B	Generate 1 second audible tone. (Touch panel must be attached for this operation.)

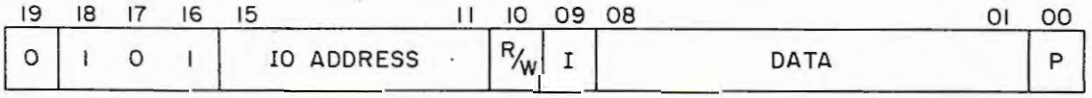
D = 100 (LDA) Load Memory Address (229μs)



This instruction loads the Memory Address Register (MAR). This data word specifies the first storage address to be used upon entry into a Mode 2 operation.

Note: Memory addresses below 2300 (HEX) are reserved for use by the resident program. Attempting to write below this address may result in erratic terminal behavior.

D = 101 (SSF) Function (223μs)



This instruction is used to read and write data to devices attached to the external bus and to enable interrupts.

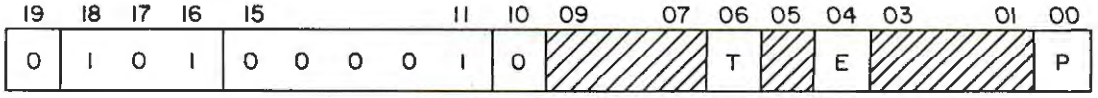
Bits 11 - 15	specify the device address.
Bit 10	specifies a read (input) operation if a "1," a write (output) if a "0."
Bit 09	inhibits the actual read or write function, but permits the device address to be saved by the terminal. The inhibit write function can be used to establish a write address for later use by the EXT command. The inhibit read function can be used to establish a read address to be used upon the occurrence of an external interrupt. If location m.extpa contains 0, the resident will perform a read from the selected device and transmit the data to PLATO via an External input word. PLATO may use a local program to process the interrupt by previously loading the program and storing the program address in m.extpa.

Write addresses 0 and 1 are special cases of the SSF instruction. Address 0 is assigned to the slide selector and for this device only the data field is 10 bits long. The SSF word format for this device is shown below.

19	18	17	16	15				11	10	09	08	05	04	01	00
0	1	0	1	0	0	0	0	0	L	S	X		Y		P

Bits 01 - 08 are sent to address 00 and bits 9 - 10 to address 01. Bits 01 - 08 select one of 256 slides for display on the plasma panel. Bit 09 controls the projector shutter. For normal operation this bit is always "0." However, if this bit is a "1," the shutter will be closed and remain closed until receipt of a load slide command with bit 09 = "0." Bit 10 controls the project lamp. The lamp will be turned on if bit 10 is a "1" and off if bit 10 is a "0."

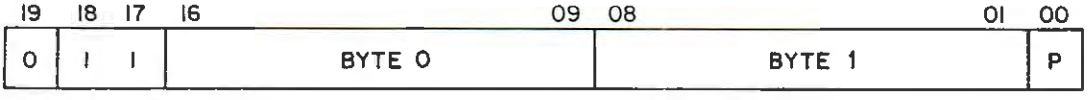
Device address 1 is assigned to the Interrupt Mask register located in the ICU. The SSF word format for this address is shown below. Memory location m.enable will be loaded with a copy of the interrupt mask data.



Bit 06 enables the Touch Panel if a "1," disables it if a "0." Bit 04 enables the External bus interrupt if a "1," disables it if a "0." (This data is not actually transmitted to address 01 because the Interrupt Mask Register is internal to the terminal.)

Following execution of this instruction, memory location m.enab will contain a copy of the mask data.

D = 11x (EXT) Load External (275μs)



This instruction transfers two 8-bit bytes, byte 0 first, to the external device selected by a previous SSF instruction.

2.2 Processing Modes - Mode 0

In normal operation, the terminal is assigned an operating mode by sending it a LDM instruction followed by all of the data to be processed in that mode.

The terminal resident program contains the programs for processing data in five modes. In addition, up to three additional user-defined-mode programs can be loaded into RAM.

Mode 0 is a point-plotting mode. Each mode 0 data word, Figure 2.3, specifies the address of a point on the panel to be written or erased.

The W/E_0 bit in the mode word determines which operation is performed.

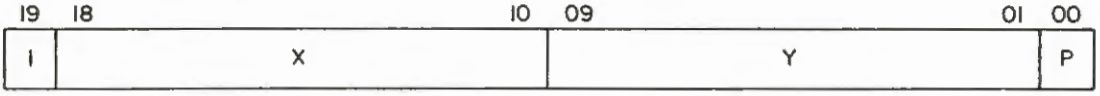


Figure 2.3. Mode 0 Data Word

The processing time for a Mode 0 word is 238μs.

2.3 Mode 1

Mode 1 is a line drawing mode. Each data word, Figure 2.4, specifies the terminal coordinates of a line, the origin of which is contained in the x and y registers.

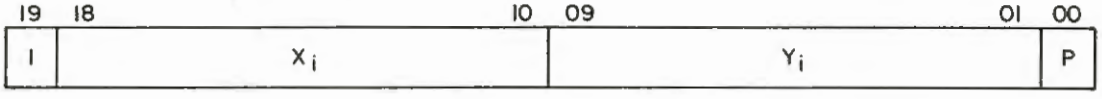


Figure 2.4. Mode 1 Data Format

The terminal point of a given line is also interpreted as the origin of the next line. Line origins may be relocated, however, by the use of the LDC command without exiting from Mode 01.

The processing time for a Mode 1 word ranges from 1ms for a line length of one dot to 11.1ms for the maximum line length of 512 dots.

2.4 Mode 2

Mode 2 is a load memory mode. Each Mode 2 data word, Figure 2.5, contains two eight-bit bytes to be stored in RAM memory. These bytes are stored, lower first, in two successive locations starting with the present contents of the memory address register (MAR). After each byte is stored, the MAR is automatically incremented by 1.

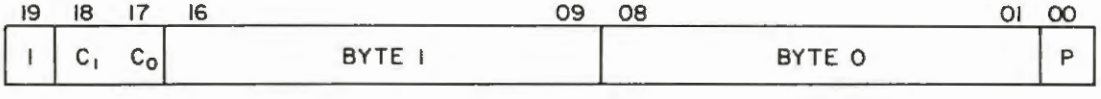


Figure 2.5. Mode 2 Data Word

As each byte is stored, a longitudinal parity check is performed by exclusive "oring" each byte with a check word and left shifting the result. This check should be all zeros at the conclusion of the transmission of a block of data.

Bits 17 and 18 (C_0, C_1) activate the block error check as follows:

C_1	C_0	
0	0	Store data.
0	1	Byte 0 is the last byte of this transmission and contains the block check character to make the longitudinal check word all "zeros."
1	0	Byte 1 is the last byte of this transmission and contains the block check character.
1	1	Not used.

After receipt of the block check byte, if the check word is not zero, the terminal automatically transmits the STATUS REPORT code (05H) to the central computer. If the data being loaded is character data, it will appear when displayed as a vertical column with bit 01 of byte 0 at the bottom and bit 16 of byte 1 at the top.

The processing time for a Mode 2 word is 288μs.

2.5 Mode 3

Mode 3 is a character-plotting mode. The data words in this mode contain three 6-bit character codes as shown in Figure 2.6. Each code selects one of 63 characters from one of the character sets contained in the terminal.

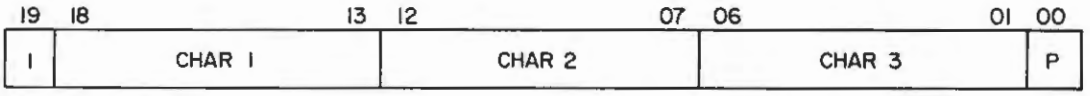


Figure 2.6. Mode 3 Data Word

The terminal provides for up to eight character sets of 63 characters each. Character sets M0 and M1 are contained within ROM memory and hold the characters shown in Table 2. The other character sets contain user-defined characters and are stored in RAM.

The contents of the character memories are processed in Mode 3 as 63 arrays of sixteen 8-bit words. The contents of 16 consecutive addresses are displayed as one character within a matrix as shown in Figure 2.7. The top three rows and bottom row of all character matrices from M0 and M1 are always unfilled.

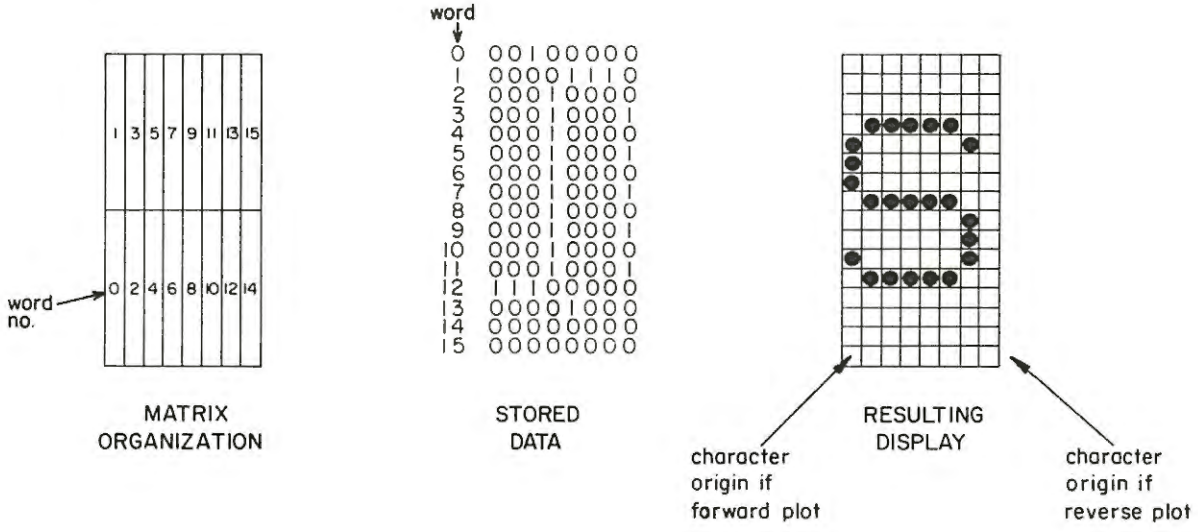


Figure 2.7. Character Matrix

The data for the characters stored in ROM is shown in Figures 2.8.0 - 2.8.3.

The contents of any character memory can be enlarged via selection of character size 2 (size 0 is normal size). Selection of size 2 will result in a 2X magnification of the characters. Figure 2.8.4 illustrates characters drawn in size 2. All character format operations will be automatically adjusted when using size 2 characters. The method of specifying character size is described later in this section.

Character write/erase is specified by the write/erase bits WE0, WE1 in the mode word. (See LDM instruction.) If WE0 = 1, characters are written; if WE0 = 0, characters are erased. The inverse of the operation called for by WE0 will be performed on the background or unfilled portion of the character matrix if WE1 = 0; while if WE1 = 1, the background remains unaltered.

Character plotting speed ranges from a minimum of 355 characters per second using a serial plasma panel and up to 3080 characters per second using a parallel plasma panel.

There are several non-plotting control characters available for formatting the display of data in Mode 3. These control characters may be accessed via the use of the "uncover" code (77). Upon receipt of a 77 code, the terminal interprets the next character code as a control character instead of a character to be plotted. Following execution of the control character, normal plotting mode is resumed. If several uncover codes are sent in sequence, the first non-uncover code will be treated as the control character.

The operations performed by each of the control characters are shown in Table 3. In the case of some characters, the operation performed is a function of the character memory, the plotting mode, horizontal or vertical, and the plotting direction, forward or reverse, which is being used.

Horizontal plotting mode is set by the 30 code; vertical plotting, by the 31 code. Forward plotting, set by the 32 code, is from left to right in horizontal mode and from bottom to top in vertical mode. Reverse plotting direction, set by the 33 code, is from right to left in horizontal mode and from top to bottom in vertical mode.

Boldface characters are selected by the 35 code, normal size by the 34 code.

The terminator code (00) is used to terminate character string plotting from local programs. See the discussion of r.chars in section 3.1.

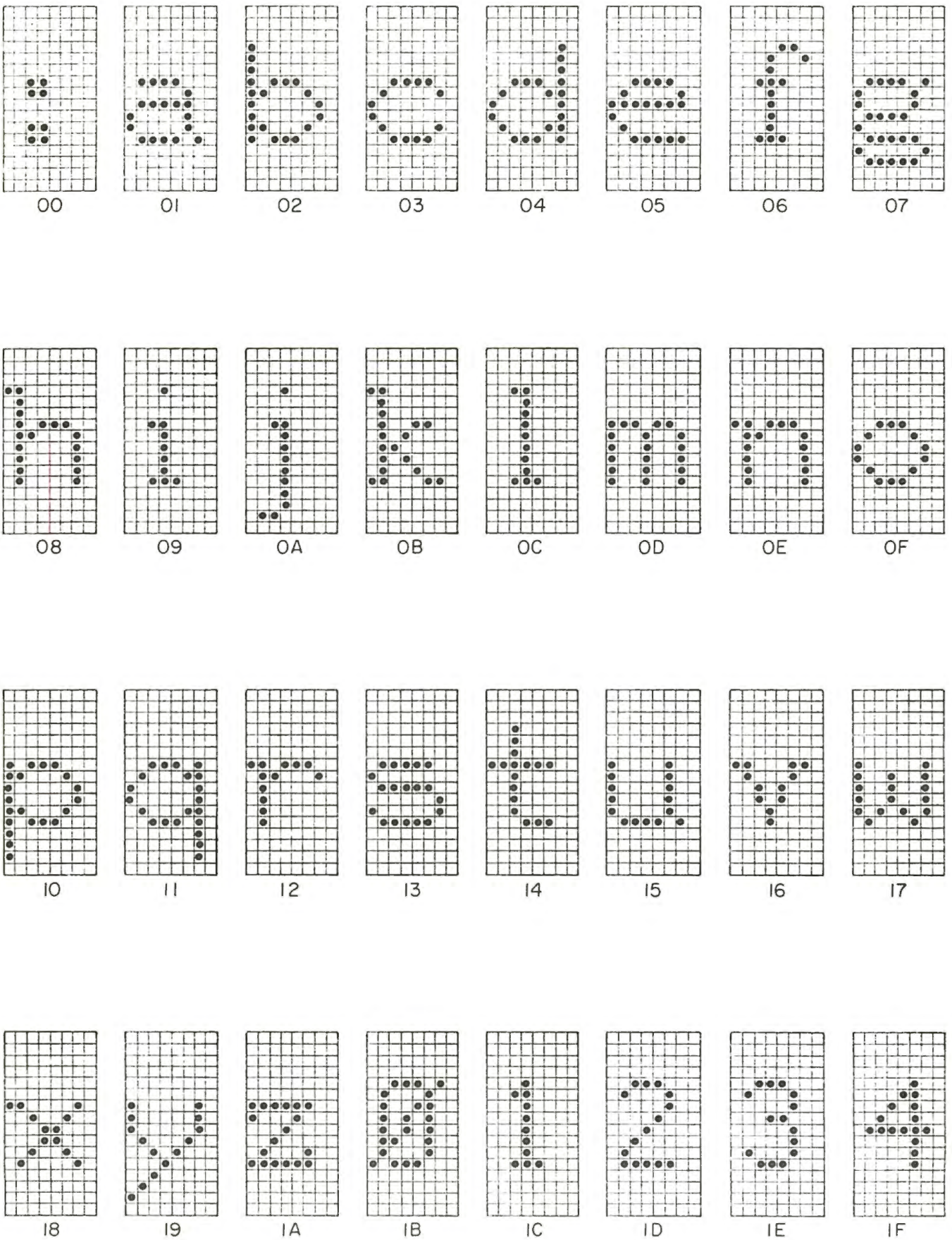


Figure 2.8.0. M0 Characters 00 - 1F

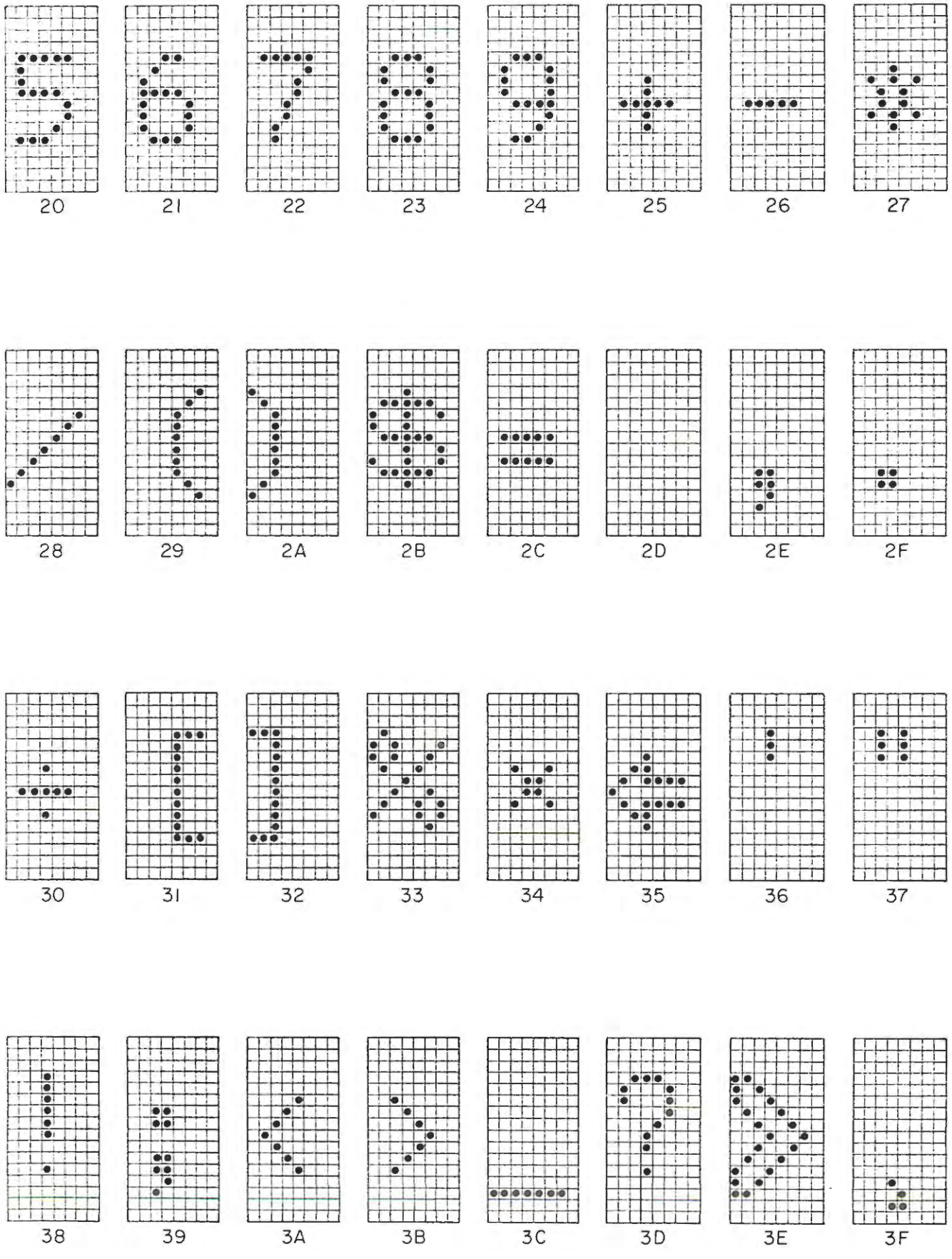


Figure 2.8.1. M0 Characters 20 - 3F

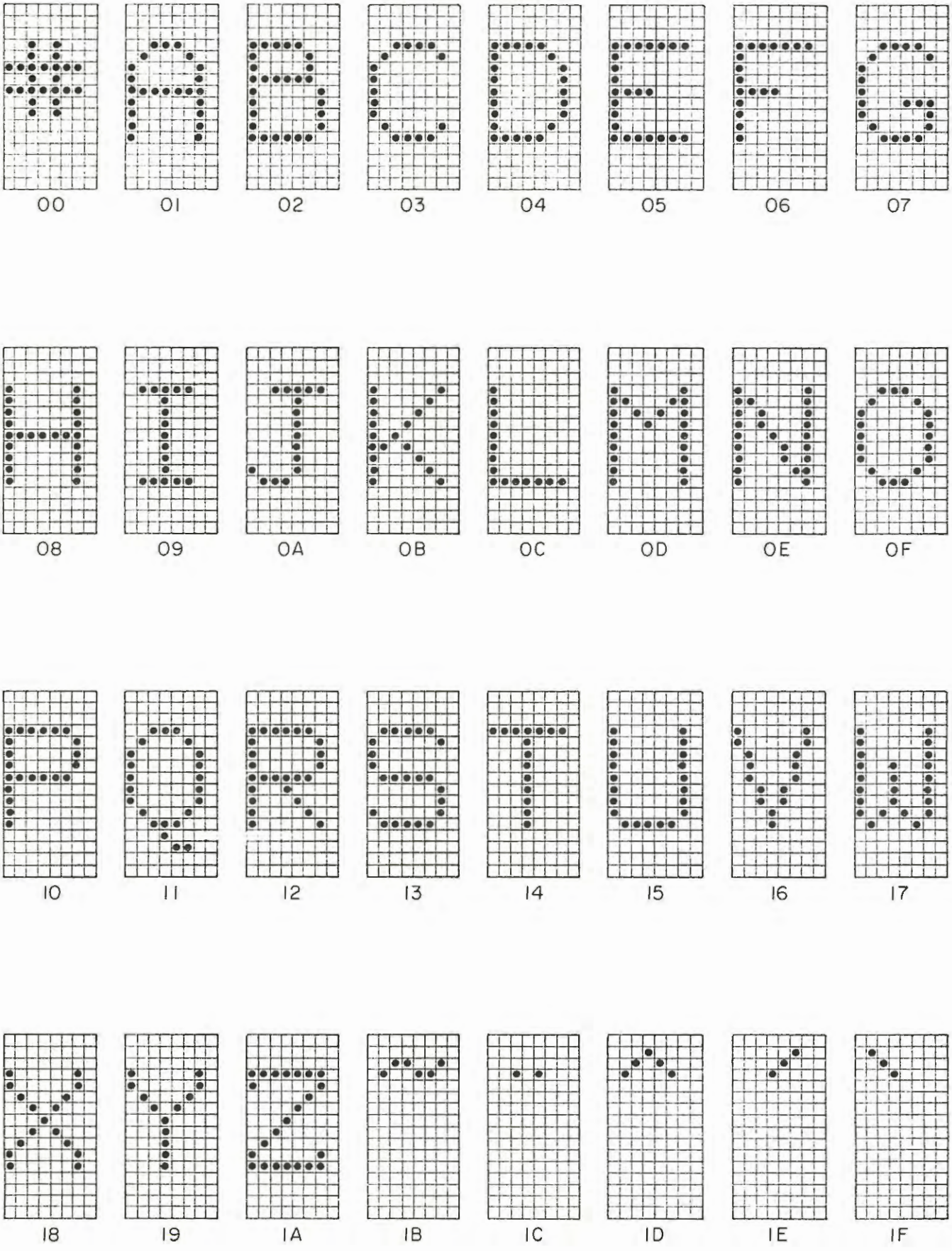


Figure 2.8.2 M1 Characters 00 - 1F

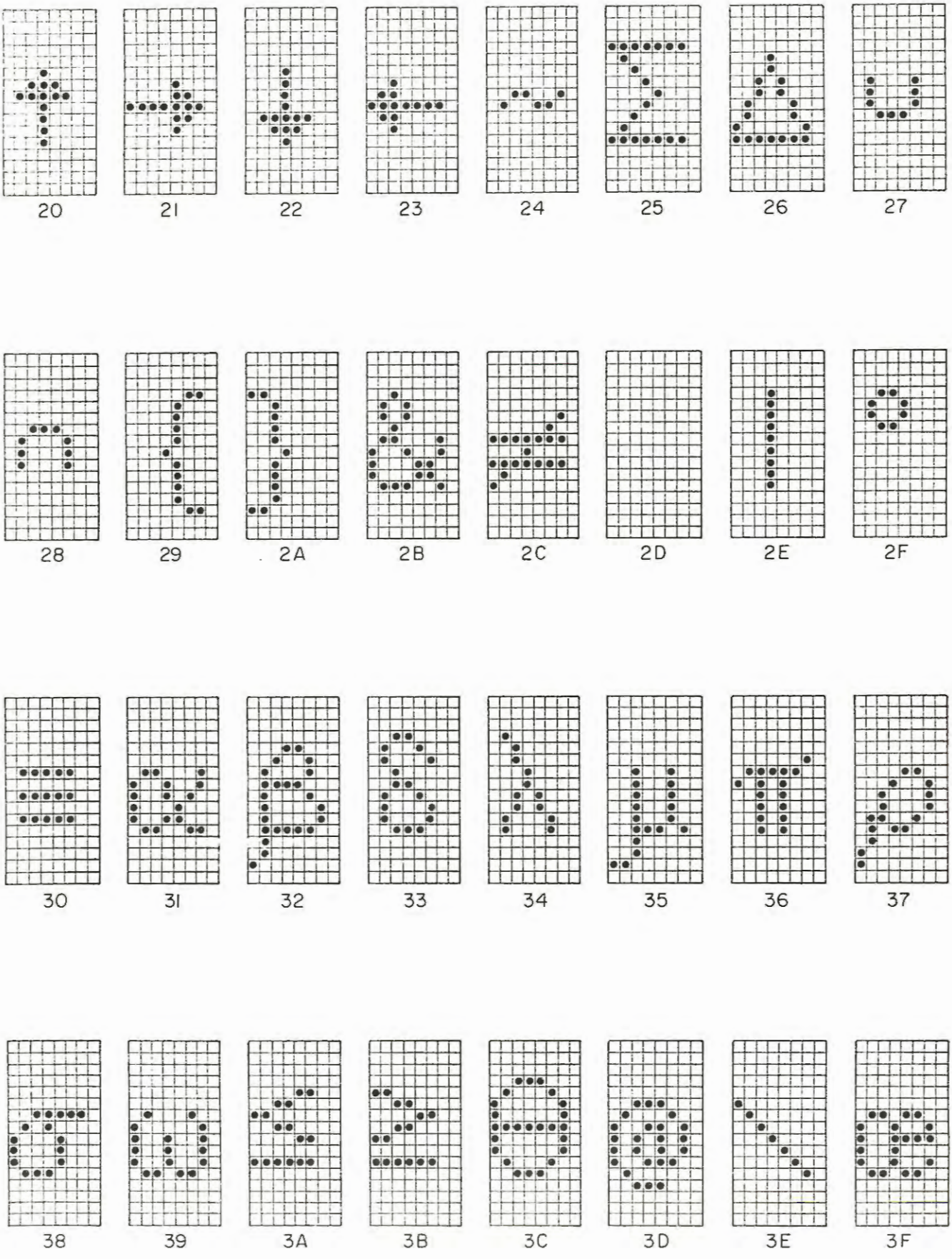


Figure 2.8.3 M1 Characters 20 - 3F

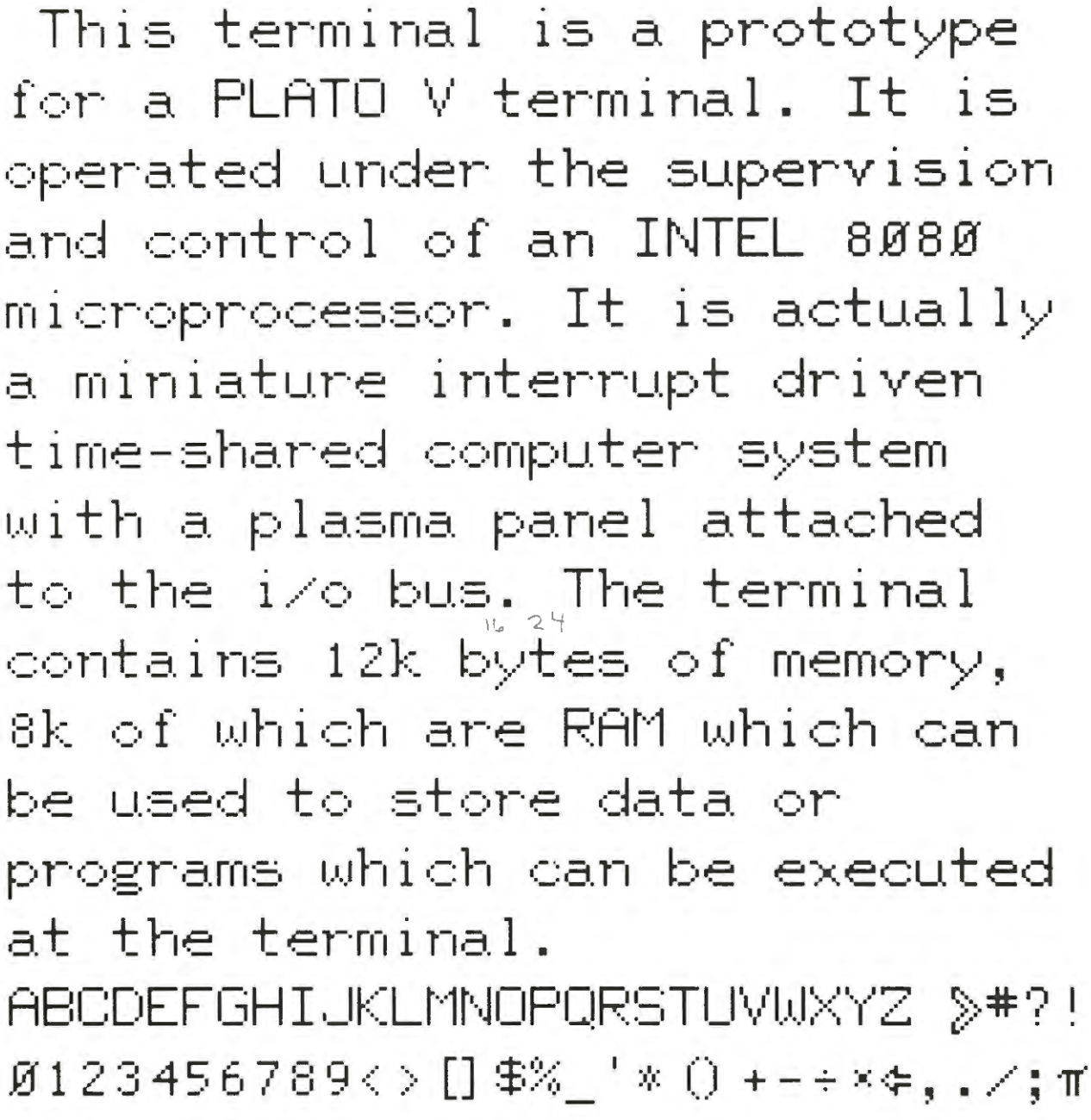


Figure 2.8.4 "Boldface" Character Set

ADDRESS (OCTAL)	M0 CHAR	M1 CHAR	ADDRESS (OCTAL)	M0 CHAR	M1 CHAR
0	:	#	40	5	↑
1	a	A	41	6	→
2	b	B	42	7	↓
3	c	C	43	8	←
4	d	D	44	9	∿
5	e	E	45	+	Σ
6	f	F	46	-	Δ
7	g	G	47	*	∪
10	h	H	50	/	∩
11	i	I	51	(	{
12	j	J	52	)	}
13	k	K	53	$	&
14	l	L	54	=	≠
15	m	M	55	SP	SP
16	n	N	56	,	|
17	o	O	57	.	°
20	p	P	60	÷	≡
21	q	Q	61	[	α
22	r	R	62	]	β
23	s	S	63	%	δ
24	t	T	64	×	λ
25	u	U	65	⇦	μ
26	v	V	66	'	π
27	w	W	67	"	ρ
30	x	X	70	!	σ
31	y	Y	71	;	ω
32	z	Z	72	<	≤
33	0	~	73	>	≥
34	1	¨	74	_	θ
35	2	^	75	?	@
36	3	´	76	⟫	\
37	4	`	77	UNCOVER	UNCOVER

Table 2. ROM Characters

HEX OCTAL CODE	FUNCTION	SIZE Ø				SIZE 2			
		HORIZONTAL		VERTICAL		HORIZONTAL		VERTICAL	
		FWD	RVS	FWD	RVS	FWD	RVS	FWD	RVS
00	Terminate	Terminate Character String Plotting							
10	Backspace	x-8	x+8	y-8	y+8	x-16	x+16	y-16	y+16
11	Tab	x+8	x-8	y+8	y-8	x+16	x-16	y+16	y-16
12	Line Feed	y-16	y-16	x+16	x+16	y-32	y-32	x+32	x+32
13	Vertical Tab	y+16	y+16	x-16	x-16	y+32	y+32	x-32	x-32
14	Form Feed	x←Ø y←496	x←5Ø4 y←496	y←Ø x←15	y←5Ø4 x←15	x←Ø y←480	x←496 y←480	y←Ø x←31	y←496 x←31
15	Carriage Return	x←(MARGIN) y-16		y←(MARGIN) x+16	x+16	x←(MARGIN) y-32	y-32	y←(MARGIN) x+32	x+32
16	Superscript	y+5	y+5	x-5	x-5	y+10	y+10	x-10	x-10
17	Subscript	y-5	y-5	x+5	x+5	y-10	y-10	x+10	x+10
20	Select MØ	select character memory Ø (ROM)							
21	Select M1	select character memory 1 (ROM)							
22	Select M2	select character memory 2 (RAM)							
23	Select M3	select character memory 3 (RAM)							
24	Select M4	select character memory 4 (RAM)							
25	Select M5	select character memory 5 (RAM)							
26	Select M6	select character memory 6 (RAM)							
27	Select M7	select character memory 7 (RAM)							
30	Horizontal	select horizontal plot mode							
31	Vertical	select vertical plot mode							
32	Forward	select forward plot direction							
33	Reverse	select reverse plot direction							
34	Select Size Ø	select normal-size characters							
35	Select Size 2	select large-size characters							
77	Uncover	next character is control character							

Table 3. Control Characters

Six memory locations are reserved as address registers specifying the origins of the loadable character sets. These locations, shown in Table 4, are normally loaded by PLATO and the character sets are referenced by the codes in Table 3.

Address	Function
2306	Character set 02 origin.
2308	Character set 03 origin.
230a	Character set 04 origin.
230b	Character set 05 origin.
230e	Character set 06 origin.
2310	Character set 07 origin.

Table 4. Character set Address Registers

2.6 Mode 4

Mode 04 is a block erase mode. In this mode each pair of data words specifies the corners of an area to be erased. The area erased is that enclosed by $|\Delta x| = |x_2-x_1|$ and $|\Delta y| = |y_2-y_1|$. If $x_2 = x_1$ and $y_2 = y_1$, a single point is erased; while if $x_2 = x_1$ and $y_2 \neq y_1$, a vertical line is erased; and if $y_2 = y_1$ and $x_2 \neq x_1$, a horizontal line is erased. The previous description assumes $WE_0 = 0$; if $WE_0 = 1$, the area is written.

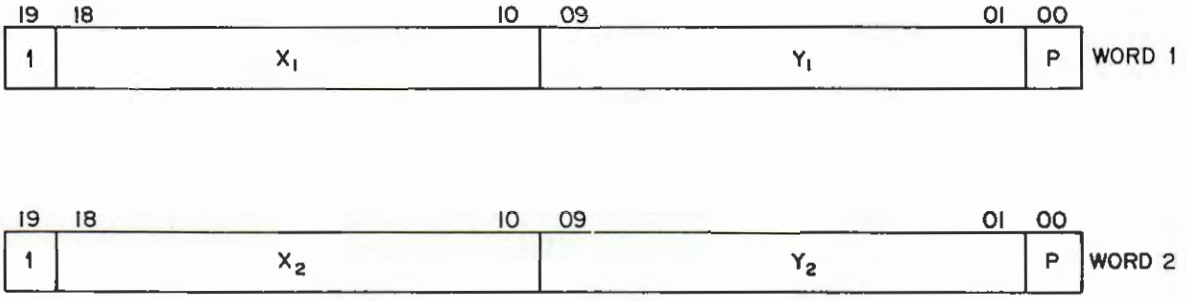


Figure 2.9. Mode 4 Word Format

After processing each pair of words in this mode, the terminal leaves the display address registers set to x_1 and y_1-15. This is the address appropriate to begin writing characters in the erased area.

2.7 Modes 5,6,7

The word format for any of these modes is defined by the user. When operating in any of these modes, the resident places the PLATO data in the C, D, and E registers as shown in Figure 2.10, and transfers control to the local program.

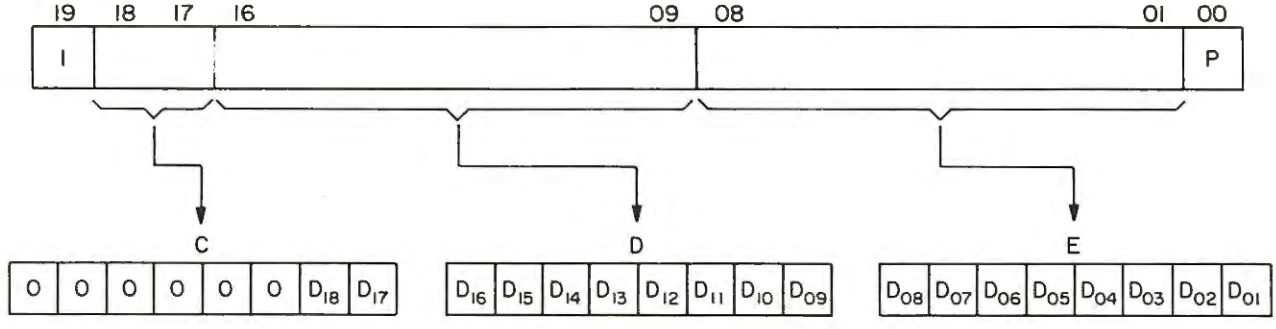


Figure 2.10. Modes 5,6,7 Word Format

Three memory locations are reserved for use as address registers specifying the origins of the local programs. These addresses are shown in Table 5.

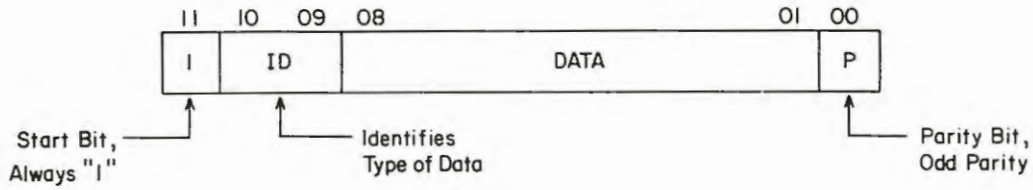
Address	Symbol	Function
2300	m5	Mode 5 program origin.
2302	m6	Mode 6 program origin.
2304	m7	Mode 7 program origin.

Table 5. Local Program Address Registers

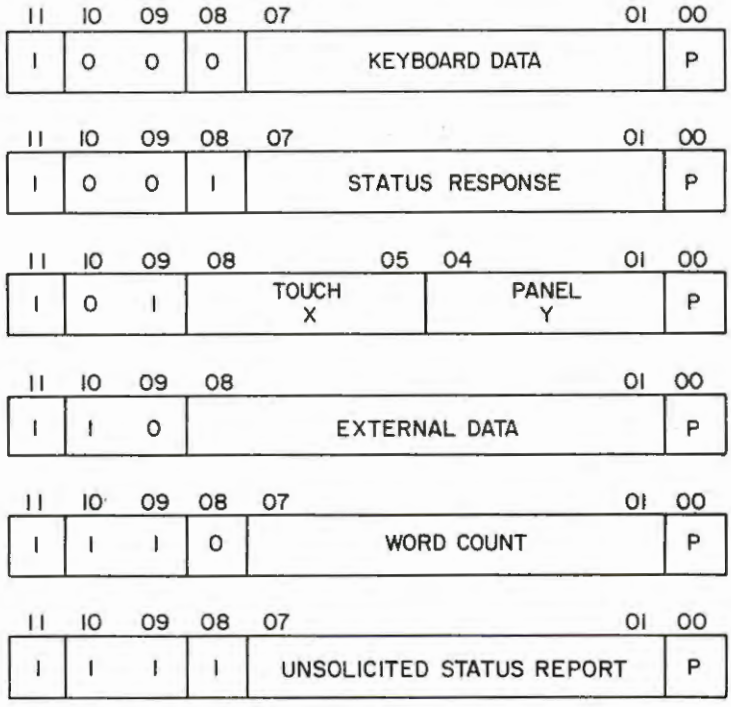
Note: Once the resident has transferred control to a local program, control of the terminal remains with that program until a return instruction is executed (or until the clear switch is depressed). If the interrupts are left enabled the resident will continue to perform all IO functions.

2.8 Output Data Format

Data transmitted from the terminal to the computer center consists of 12-bit words with the format shown below.



The six types of terminal data words are shown below:



Status Request code 70H is used to request terminal type. A response of 73H indicates an 8080-type terminal with 8k of RAM memory.

The Unsolicited Status Report codes are used to inform the central computer of the occurrence of some special event within the terminal. A list of the presently used codes is shown in Table 6.

Status Report (hex)	Event Reported
02	Reset (clear switch has been depressed).
05	Longitudinal parity error occurred in Mode 2.

Table 6. Unsolicited Status Report Codes

3. RESIDENT PROGRAM

3.0 General

The terminal memory allocation is shown in Figure 3.0. All of ROM and that portion of RAM below address 2300 (hex) is reserved for use by the resident. The resident program contains those programs required to process PLATO data plus routines for operating the serial communication port, the parallel IO bus, servicing interrupts and communicating with the Display Interface Unit.

The push down stack is used by the resident to store the status of the terminal during the processing of interrupts. The job stack provides temporary storage for incoming jobs in the event the processor is busy. The resident and PLATO variable sections of memory contain terminal status information which may be used by both the resident and user programs.

3.1 Resident Subroutines

The resident program provides several callable subroutines which may be referenced by user programs. In using these subroutines the following convention should be observed:

1. Single argument subroutines will have the argument passed in the HL register pair.
2. Double argument subroutines will have one argument passed in HL and the other in DE.
3. Results are returned from subroutines in HL.
4. The user must provide for saving and restoring any register or status he wants preserved.
5. Reference to a subroutine should always be by symbol and not memory address.

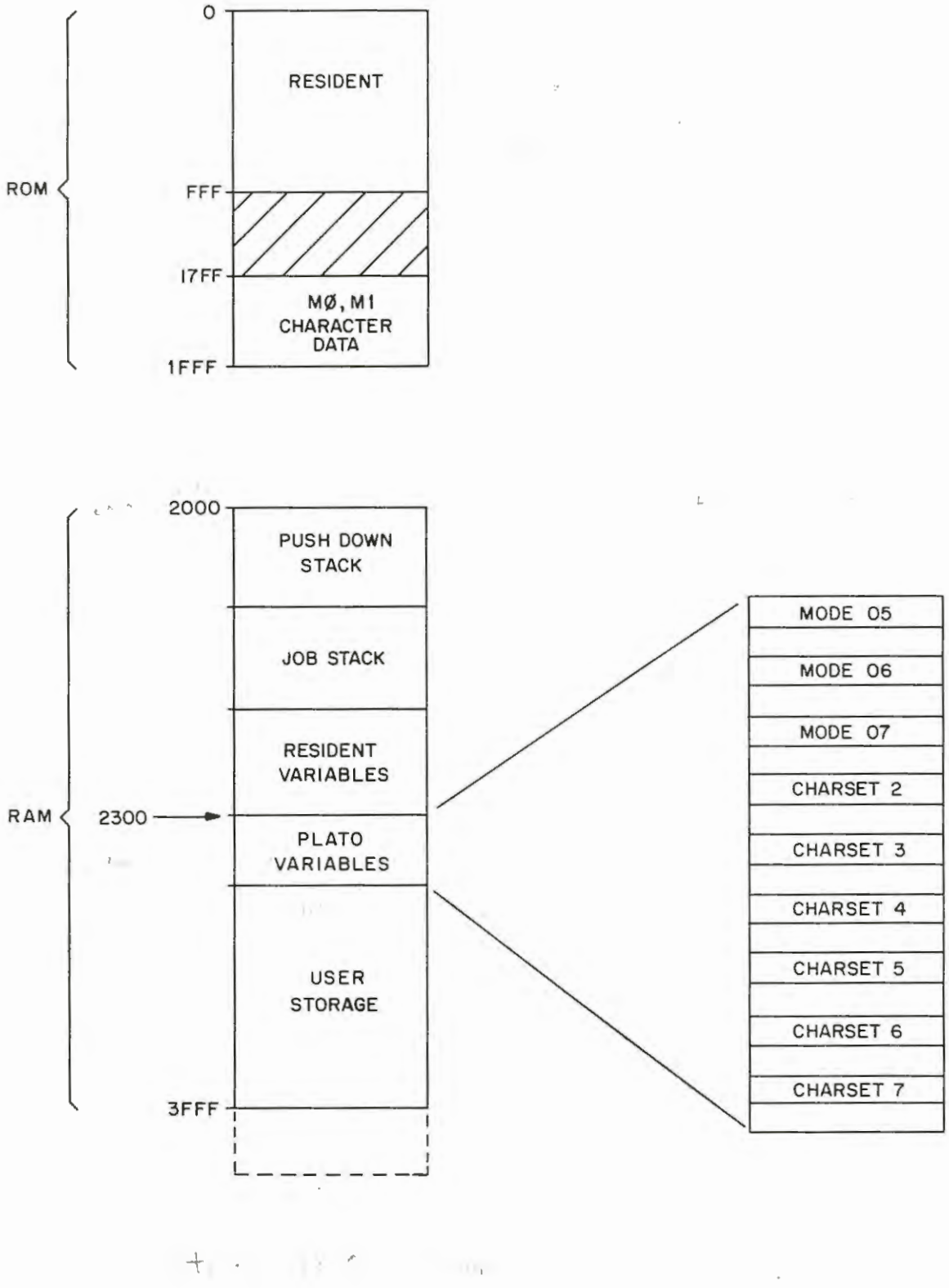


Figure 3.0. Memory Allocation

Following is a list of the subroutines.

r.init (40)

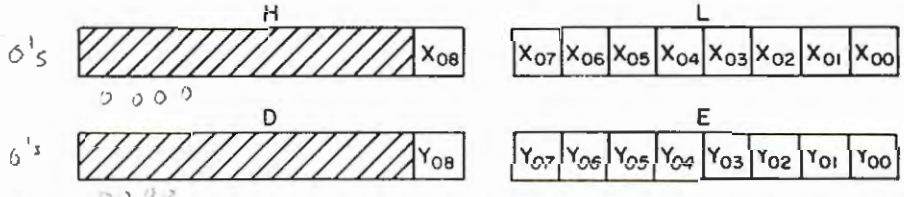
This routine will initialize the terminal operating conditions as follows:

a. The screen will be erased.
b. Memory locations m.margin, m.ksw, and m.extpa will set to zero.
c. M.enab will set to enable the serial input port (SIR, SOR, CARRIER), and the keyset (KST) interrupts.
d. Select character memory m0; select normal character size; select horizontal left to right plotting mode.
e. Initialize other pointers required by the resident.
f. Remove ABORT condition if it exists.

Note: A jump may be made to this routine in which case control will be returned to the resident after execution.

r.dot (43)

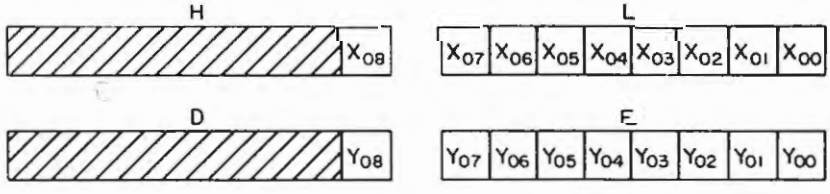
This routine will write (or erase) the point on the display screen specified by the contents of the HL and DE registers.



The WE_0 bit in location m.mode specifies a write operation if a "1," an erase operation of a "0." After execution of this routine the x and y registers in the DIU will contain the values entered in HL and DE.

r.line (46)

This routire will write (or erase) a line on the display screen originating at the current address given by the x and y registers and terminating at the address contained in HL and DE.



The WE_0 bit in m.mode specifies a write operation if a "1," an erase operation if a "0." After execution of this routine the x and y registers in the DIU will contain the values entered in HL and DE.

r.chars (49)

This routine will write (or erase) a string of characters on the display. Register pair HL specifies the string origin address. The string must be terminated with an uncover code followed by the terminator code (7700). Character coding is the same as shown in Tables 2 and 3. Character write/erase is specified by the WE_0 and WE_1 bits in location m.mode as described in the discussion of mode 3.

r.block (4c)

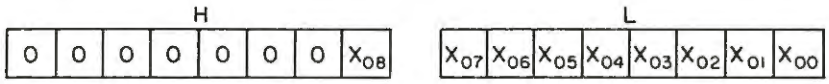
This routine will erase (or write) an area of the display screen specified by a list of coordinates stored in memory. Register pair HL contains the origin address of the list. The coordinates are stored in the following order:

hl	x7------x0	x_1 lower
hl+1	0000000x8	x_1 upper
hl+2	y7------y0	y_1 lower
hl+3	0000000y8	y_1 upper
hl+4	x7------x0	x_2 lower
hl+5	0000000x8	x_2 upper
hl+6	y7------y0	y_2 lower
hl+7	0000000y8	y_2 upper

Coordinates x_1y_1 and x_2y_2 are any two corners of the area involved. The area will be erased if the WE_0 in location m.mode is "0," written if it is a "1."

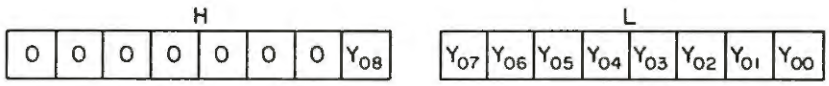
r.inpx (4f)

This routine will return the current display x address in HL.



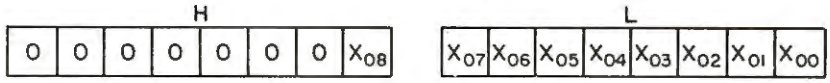
r.inpy (52)

This routine will return the current display y address in HL.



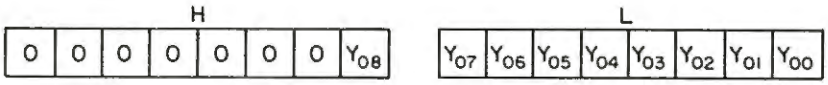
r.outx (55)

This routine will load the display x address register with the contents (lower 9 bits) of HL.



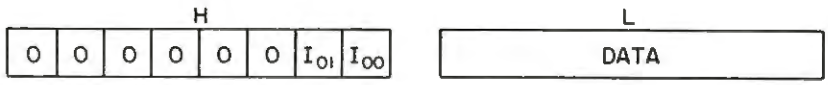
r.outy (58)

This routine will load the display y address register with the contents (lower 9 bits) of HL.



r.xmit (5b)

This routine will transmit to PLATO the contents (lower 10 bits) of HL.



$i_1 i_0$ identify the source of data as follows:

0 0	keyset or status response word.
0 1	touch panel word.
1 0	external input word.
1 1	word count or unsolicited status word.

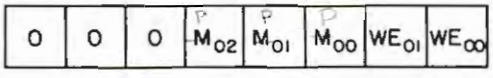
The output word formats are described in more detail in section 2.7.

r.mode (5e)

This routine will load location m.mode with the terminal and display operating mode as specified by the contents of HL.

s	specifies a full screen erase (the screen will be erased when this routine is called if s is "1," the screen is not affected if this bit is "0").
$WE_1 WE_0$	specify the write erase mode as follows: 00 erase, rewrite mode. 01 write, rewrite mode. 10 erase, overstrike mode. 11 write, overstrike mode.
$m_2 m_1 m_0$	specify terminal processing mode.

Note: The screen bit is discarded before storing the data, leaving m.mode with the following format:



r.stepx (61)

This routine will increment or decrement the display x address register. The contents of location m.dir specify the operation to be performed as follows:

0	0	0	0	0	0	XD	YD

xd specifies x direction as follows:

0 = forward (increment)
1 = reverse (decrement)

yd specifies y direction.

r.stepy (64)

This routine will increment or decrement the display y address register. The contents of location m.dir specify the operation to be performed as in r.stepx.

r.we (67)

This routine will write (or erase) the current xy address.

r.dir (6a)

This routine will setup the x and y direction flags for later use by the r.stepx and r.stepy routines. Location m.dir will be loaded with contents of HL.

0	0	0	0	0	0	XD	YD

xd specifies x direction as follows:

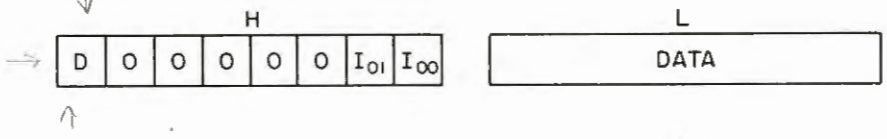
0 = forward (increment)
1 = reverse (decrement)

yd specifies y direction.

r.input (6d)

This routine will return in HL the last keyset, touch panel or external word received by the terminal. If location m.ksw is 0, the word will also have been sent to PLATO. If m.ksw = 3, the word is returned only to the user's program.

The format of the data is shown below.

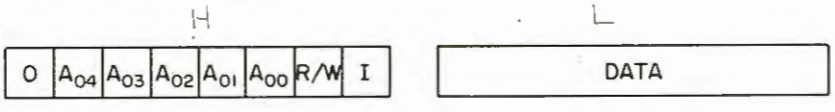


d	"0" if data present, "1" if no data.
$i_1 i_0$	specify source of data as follows:

0 0	keyset word.
0 1	touch panel word.
1 0	external word.

r.ssf (70)

This routine is used to read and write data to devices on the external bus and to enable interrupts.



a_4-a_0	specify the device address.
r/w	specifies a read (input) if "1," a write (output) if "0."
i	if this bit is "1," the device address will be saved by the resident, but the actual read or write operation will be inhibited.
data	in an ssf write operation, L contains the data to be sent to the external device. In an ssf read operation, r.ssf will return in L the data read from the external device.

The inhibit write ssf is normally used to establish a write address for later use by the r.extout routine (described later). The inhibit read ssf is used to establish a read address for later use by an external interrupt processing program. In this case, the user must have previously loaded location m.extpa with the address of the interrupt program. If m.extpa contains 0, the resident will perform the read operation when the external interrupt occurs. The data will be stored in memory and may be retrieved via the r.input routine. Note that if the resident interrupt program is used, location m.ksw must contain a 3 to prevent the data from also being sent to PLATO.

Write addresses 0 and 1 are special cases of the ssf routine. Device address 0 is assigned to the slide selector and for this device only, the data bits are sent to address 0 and bits 09 and 10 are sent to address 1.

Device address 1 is assigned to the Interrupt Mask register which is located within the terminal. The data format for this register is described below.

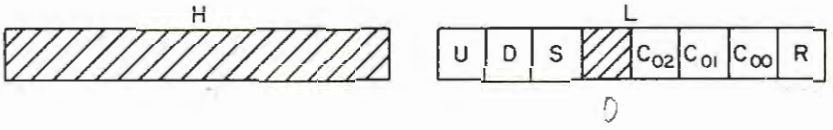
SIR	KST	TP	1	0	EXT_0	CON	CAR

sir = serial input port interrupt
kst = keyset interrupt
tp = touch panel interrupt
ext_0 = external IO interrupt
con = console switch interrupt
car = modem carrier interrupt

An interrupt is enabled if the associated bit is a "1," disabled if "0." Memory location m.enable will be loaded with a copy of the interrupt mask data.

r.ccr (73)

This routine is used to establish character plotting conditions for use by the r.chars routine.



u	specifies that the last character plotted was the "uncover" code (77). This bit should normally be set to "0."
d	specifies the character plotting direction. A "0" = forward, a "1" = reverse. Forward direction is left to right in horizontal plotting and bottom to top in vertical plotting.
s	specifies character size, 0 = normal, 1 = large.
$c_2c_1c_0$	specifies character memory.
r	specifies horizontal plot if "0," vertical plot if "1."

Location m.ccr will be loaded with the data.

r.extout (76)

This routine will transmit the data stored in a buffer to the currently selected device on the external IO bus. The device address may be set by the r.ssf routine.

Register HL specifies the origin of the string and DE the length.

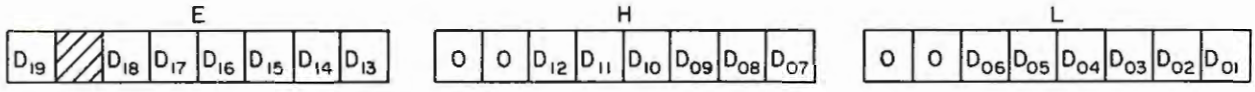
r.exec (79)

This routine may be used to execute the jobs waiting in the job stack. The job stack can hold a maximum of 1.48 seconds of PLATO output (excluding NOPs). The number of jobs awaiting execution is contained in location m.jobs.

Careful use of this routine will permit time sharing of the terminal by PLATO and local programs.

r.gjob (7c)

This routine may be used to retrieve the next job awaiting execution in the job stack. The job is returned in registers DE and HL with the format shown below.



The job retrieved will not be executed by the resident. Location m.jobs will be decremented for each job retrieved.

Note: Attempting to use this routine when m.jobs is zero (emtpy job stack) will cause erroneous data to be returned in DE and HL.

r.xjob (7f)

This routine may be used to pass to the resident for execution the job contained in registers E, H and L. The format of the job is the same as for r.gjob.

3.2 Resident Variables

This section of memory contains information used by the resident which may also be used by local programs. This data may be read directly by local programs, with the exception of m.dir and m.ksw, should be written using the callable routines described earlier.

Note: Access to this data should always be by symbol and not by address!

Address	Symbol	Function
22EB	m.type	specifies terminal type.
22EC	m.clock	16 bit clock having a period of 6.67ms. The SIR interrupt must be enabled to insure accurate timing.
22EE	m.extpa	specifies address of program to process external interrupts. A value of 0 specifies the resident interrupt processing program.

Address	Symbol	Function
22F0	m.margin	specifies present margin setting to be used for carriage returns.
22F2	m.jobs	specifies number of jobs remaining in job stack. See r.gjob and r.exec for use of this data.
22F4	m.ccr	specifies character mode plotting conditions. See r.chars for definition of data.
22F6	m.mode	specifies terminal operating mode. See r.mode for definition of data.
22F8	m.dir	specifies directional information for display address registers. See r.dir, r.stepx, and r.stepy.
22FA	m.ksw	controls transmission of data to central computer. If this location contains 0, data is transmitted to central computer; if this location contains 3, data is retained at terminal. See r.input for more details.
22FC	m.enab	specifies interrupt selection. See r.ssf for more details.

3.3 Console Program

This program is provided in the resident for use as a program diagnostic tool. It permits the keyset and display to be used as a computer console. To gain access to this program, the RUN-CONSOLE switch must be placed in the CONSOLE position. The contents of the 8080 registers will then be displayed across the top of the display screen and an arrow appears in the lower left corner where directives may be entered from the keyset. These directives and their function are described below.

Keyboard Entry	Function
dxxxx	Display 256 bytes of memory starting at address $xxxx_{16}$.
ixxxx,DD	Insert Data DD_{16} into memory location $xxxx_{16}$.
bxxxx	Set breakpoint register (BP) to $xxxx_{16}$.

Keyboard Entry	Function
jxxxx	Jump (call) to location $xxxx_{16}$. If RUN-CONSOLE switch is in CONSOLE position, one instruction will be executed at xxxx and control will return to the CONSOLE program. If the switch is in RUN position, an exit from CONSOLE mode will occur to address xxxx.
cxxxx	Call program at address $xxxx_{16}$ (called program must contain a RTN instruction to return to CONSOLE routine).
(space Bar)	Step. Execute one instruction at the present value of the program counter (PC).
□ key	Execute program starting at the present value of PC and continue until the breakpoint address is encountered. The contents of the 8080 registers are displayed during program execution.
shift □	Same as above, but do not update register display. The program will be executed faster than above.
f xxxx, nn, cc	Fill memory, beginning at address $xxxx_{16}$ and continue for nn_{16} locations, with data cc_{16}. The full character will be assumed 0 if cc is absent.
rn,xx	Load register n with data xx_{16}. The registers are numbered from left to right as they are displayed across the top of the screen. 0 = A, 1 = flags (P), 2 = B, 3 = C, 4 = D, 5 = E, 6 = H, 7 = L
stop	Stop program execution.

Placing the RUN-CONSOLE switch in the RUN position and pressing SHIFT-STOP or SHIFT-BACK will return control to the resident. Pressing the CLEAR switch will also return control to the resident, but will also initialize the terminal.

3.4 Input-Output Addresses

Input and output operations are performed by the IN and OUT instructions and involve the transmission of 8-bit data words between the accumulator (A) register and devices external to the 8080 processor. In some cases the OUT instruction is used to set control flags in the Display Interface Unit, and no data is actually transferred. In these cases the contents of A are immaterial.

The format of the I/O address is shown in Figure 3.1.

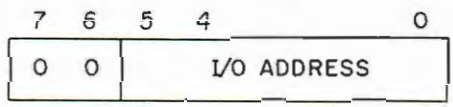


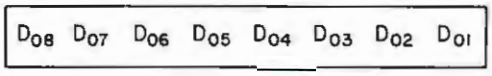
Figure 3.1. I/O Address Format

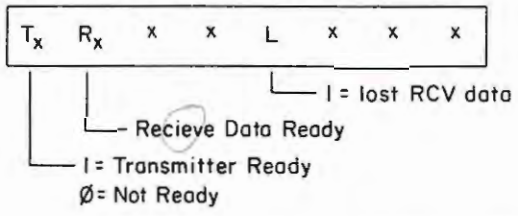
Addresses 0 through 1F (hex) are used internally by the terminal while 20 through 3F are used externally. Only external addresses may be specified by the SSF instruction.

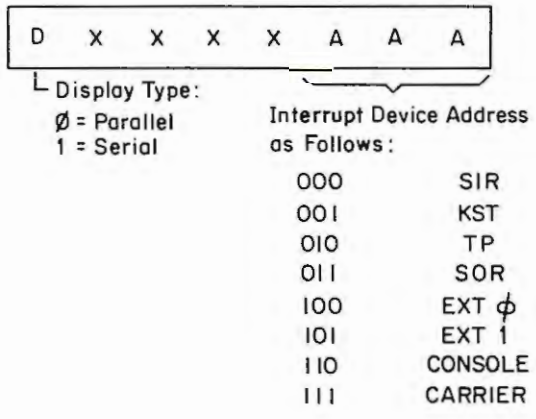
The address assignments are tabulated on the following pages. They appear here for information purposes only; the user should not write programs using these addresses, but instead should make use of the callable subroutines described earlier.

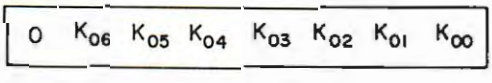
INPUT

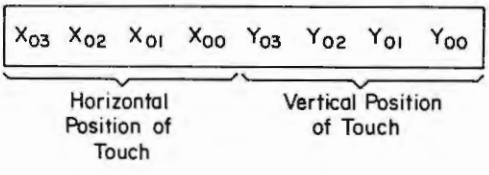
Input Address (hex)	Mnemonic	Function
ØØ	SIO	Input serial data byte
Ø1	COMSTAT	Input communication port status
Ø2	INTVECT	Input interrupt vector and display type
Ø3	Unused	
Ø4	KST	Input keyset word
Ø5	TP	Input touch panel word
Ø6-ØF	Unused	



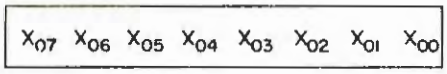


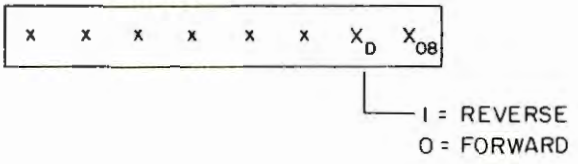


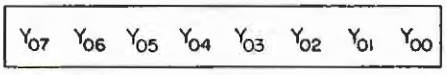


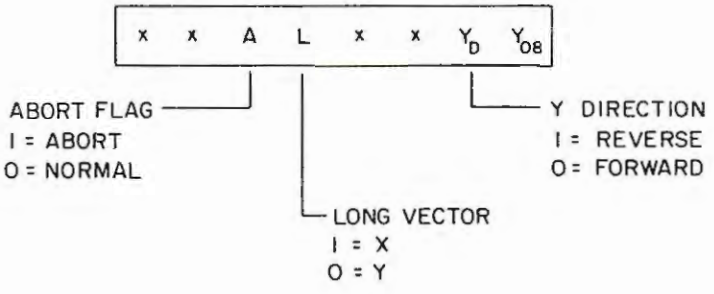


Input Address (hex)	Mnemonic	Function
10	XL	Input lower 8 bits of x register
11	XU	Input most significant bit (x_8) of x and the x direction flag
12	YL	Input lower 8 bits of y register
13	YU	Input most significant bit (y_8) of y and the y direction, long vector, and ABORT flags
14-1F	Unused	
24-3F		Available for user programs



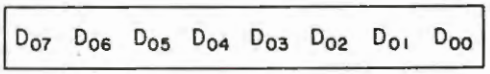


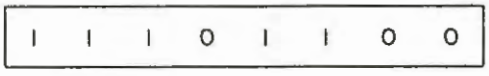


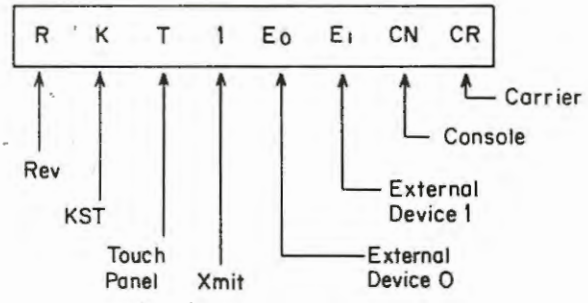


OUTPUT

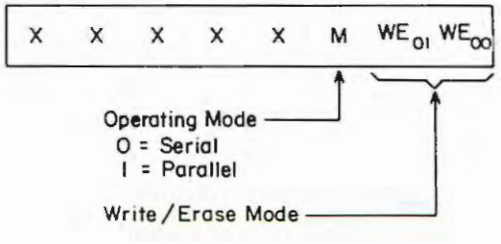
Output Address (hex)	Mnemonic	Function
ØØ	SIO	Load the transmitter with the contents of A and transmit to the central computer
Ø1	COMSTAT	Load the serial IO port with the following status word. This word conditions the port to receive PLATO data.
Ø2	Unused	
Ø3	IMASK	Load the interrupt mask register with the contents of A
Ø4-Ø7	Unused	
Ø8	XLONG	Set the long vector in the DIU to x. The contents of A are unused. See OUT CLOCKL instruction for use of this flag.
Ø9	YLONG	Set the long vector in the DIU to y. The contents of A are unused. See OUT CLOCKL instruction for use of this flag.
ØA	SETXR	Set the x direction flag in the DIU. The x register will be decremented by all subsequent clock x signals. The contents of A are unused.
ØB	SETXF	Reset (CLEAR) the x direction flag in the DIU. The x register will be incremented by all subsequent clock x signals. The contents of A are unused.







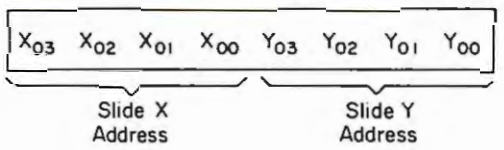
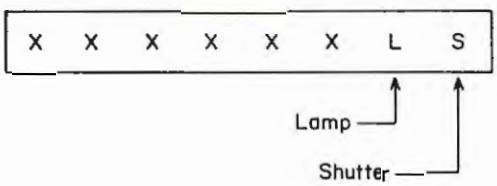
Output Address (hex)	Mnemonic	Function
ØC	SETYR	Set the y direction flag in the DIU. The y register will be decremented by all subsequent clock y signals. The contents of A are unused.
ØD	SETYF	Reset (CLEAR) the y direction flag in the DIU. The y register will be incremented by all subsequent clock y signals. The contents of A are unused.
ØE	SETABT	Set the ABORT flag. This instruction places the terminal in the ABORT mode. The contents of A are unused.
ØF	CLRABT	Reset (CLEAR) the ABORT flag. This instruction places the terminal in the normal operating mode. The contents of A are unused.
1Ø	XL	Load the lower 8 bits of the x register with the contents of A.
11	XU	Load the most significant bit (x_8) of the x register with bit $A_Ø$ of the accumulator The other bits of A are unused.
12	YL	Load the lower 8 bits of the y register.
13	YU	Load the most significant bit (y_8) of the y register with bith $A_Ø$ of the accumulator. The other bits of A are unused.
14	PDL	Load the lower 8 bits of the panel parallel data register with the contents of A.
15	PDU	Load the upper 8 bits of the panel parallel data register with the contents of A and write ($WE_Ø$=1) or erase ($WE_Ø$=4) the contents of the parallel data register (16 bits) on the display.
16	PDM	Load the DIU mode register with the lower 3 bits of A.



Output Address (hex)	Mnemonic	Function
17	PDLU	Load both PDL and PDU with the contents of A and write ($WE_\emptyset$=Ø) or erase ($WE_\emptyset$=1) the contents of the parallel data register on the display. NOTE: The write (erase) operation is reversed in this operation.
18	CLOCKX	Clock the x register. The x direction flag specifies the direction; forward if reset (Ø), reverse if set (1). The contents of A are unused.
19	CLOCKY	Clock the y register. The y direction flag specifies the direction; forward if reset (Ø), reverse if set (1). The contents of A are unused.
1A	CLOCKXY	Clock both the x and y registers and write ($WE_\emptyset$=1) or erase ($WE_\emptyset$=Ø) the resulting address on the panel. The x and y direction flags specify direction of change. The contents of A are unused.
1B	CLOCKL	Clock the long vector, x or y, (as specified by the long vector flag) in the DIU if $A_\emptyset$=Ø; clock both the x and y registers if $A_\emptyset$=1. The resulting address is then written (WE_0=1) or erased ($WE_\emptyset$=4) on the display. The direction flags specify direction of xy change.
1C	HCHAR	Clock the y register and write (or erase) the resulting address on the panel. The contents of $A_\emptyset$ and the WE bits in the Panel Mode register specify the operation performed as follows:

$A_\emptyset$	WE_1	$WE_\emptyset$	FUNCTION
Ø	Ø	Ø	write
Ø	Ø	1	erase
Ø	1	Ø	nop
Ø	1	1	nop
1	Ø	Ø	erase
1	Ø	1	write
1	1	Ø	erase
1	1	1	write

1D	VCHAR	Clock the x register and write (or erase) the resulting address on the panel as shown in the table for OUT HCHAR instruction.

Output Address (hex)	Mnemonic	Function
1E	WE	Write if $WE_{\emptyset}$=1 or erase if $WE_{\emptyset}$=Ø the address specified by the contents of the x and y registers. The contents of A are unused.
1F	SCREEN	Erase the entire display. The contents of A are unused.
2Ø	SLIDEL	Load the lower 8 bits of the slide projector register with the contents of A. 
21	SLIDEU	Load the upper 2 bits of the slide projector with the lower 2 bits of A.  Note: This address cannot be specified by an SSF instruction. This operation occurs automatically when using address 20.
22	---	Available for user defined equipment.
23	---	Available for user defined equipment.
24	---	PLATO IV Peripherals.
25	---	PLATO IV Peripherals.
26-3F	---	Available for user defined equipment.

4. IO BUS INTERFACE

4.0 IO Signal Definition

This section contains the information needed to design peripheral equipment to be attached to the external IO bus.

External devices communicate with the terminal via the IO bus shown in Figure 4.0.

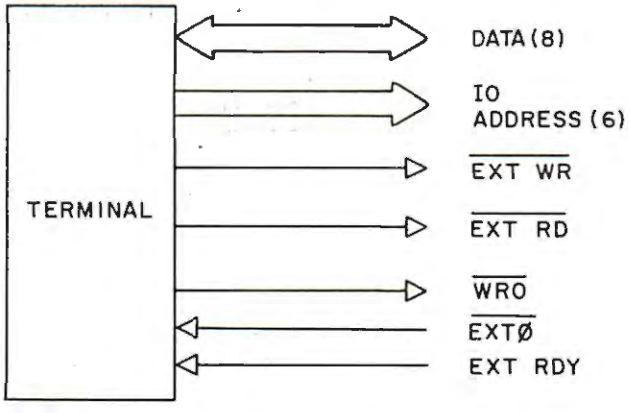


Figure 4.0. IO Bus

The function of these signals is described below. All signals are positive logic TTL levels.

DATA	8 bits of bidirectional data.
IO ADDRESS	6 bits of IO address specifying the external equipment to receive or send data.
$\overline{\text{EXT WR}}$	Low level indicates that IO address is an output (WRITE) address.
$\overline{\text{EXT RD}}$	Low level indicates that IO address is an input (READ) address.
$\overline{\text{WRO}}$	Low level indicates data bus contains valid data to be used by external device.
$\overline{\text{EXTO}}$	Low level indicates that a device attached to the IO bus requests service. This line should be held low until the requested service is performed.

EXT RDY	High level indicates that external equipment specified by address bus can perform requested operation. Addressed device should take this line low if it is unable to perform requested operation. Terminal will then halt until device is ready.

4.1 IO Timing

The timing diagram for the input operation is shown in Figure 4.1. All times are in nanoseconds (NS). No earlier than 90NS after placing the IO address on the address bus, the terminal issues the $\overline{\text{EXT RD}}$ signal. The selected device places the data on the bus within 350NS after receipt of the $\overline{\text{EXT RD}}$ signal. If the device cannot supply data within this time, it must take the EXT RDY line low within 50NS after receipt of the $\overline{\text{EXT RD}}$ signal or within 140NS after decoding the address.

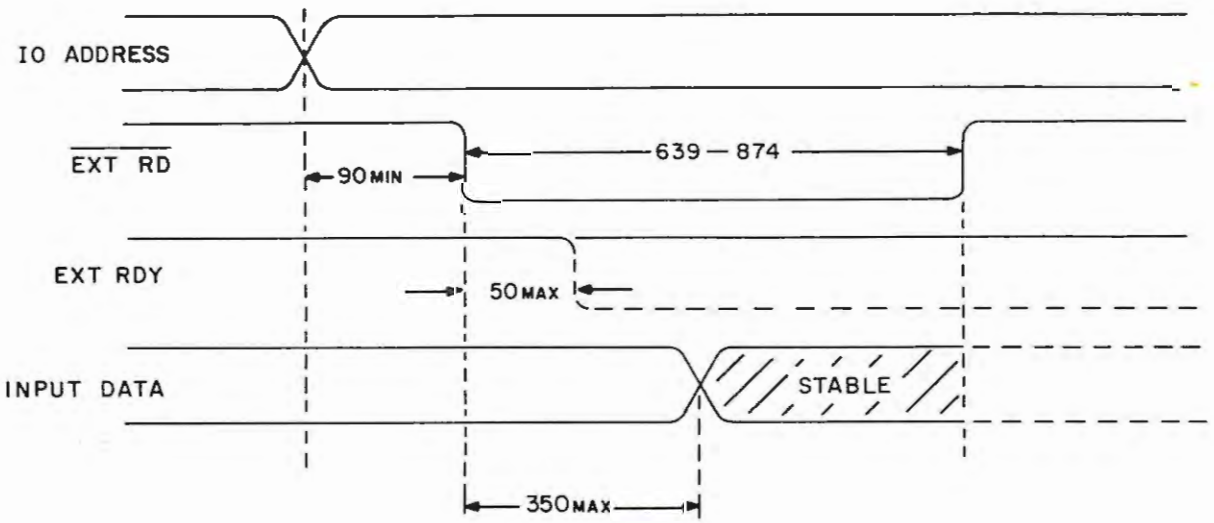


Figure 4.1. Input Timing

The terminal will then wait until the device has data ready. When ready, the selected device places the data on the bus and raises the EXT RDY line. Once issued, data must remain stable until the $\overline{\text{EXT RD}}$ is terminated by the terminal.

The timing diagram for the output operation is shown in Figure 4.2. No earlier than 90NS after placing the IO address on the address bus, the terminal issues the $\overline{\text{EXT WR}}$ signal indicating to the selected device that data is forthcoming. The terminal places the data on the data bus at least 200NS before issuing the $\overline{\text{WRO}}$ signal. The selected device can use this signal to read the data. If the selected device is not ready to receive data, it must take the EXT RDY line low within 50NS after receipt of the $\overline{\text{EXT WR}}$ signal or within 140NS after decoding the address. The terminal will still issue the $\overline{\text{WRO}}$ signal, but will halt with this signal low until the selected device raises the EXT RDY signal. After receipt of the EXT RDY signal, the terminal will hold the $\overline{\text{WRO}}$ signal and the data signals stable for 500NS before terminating the output operation.

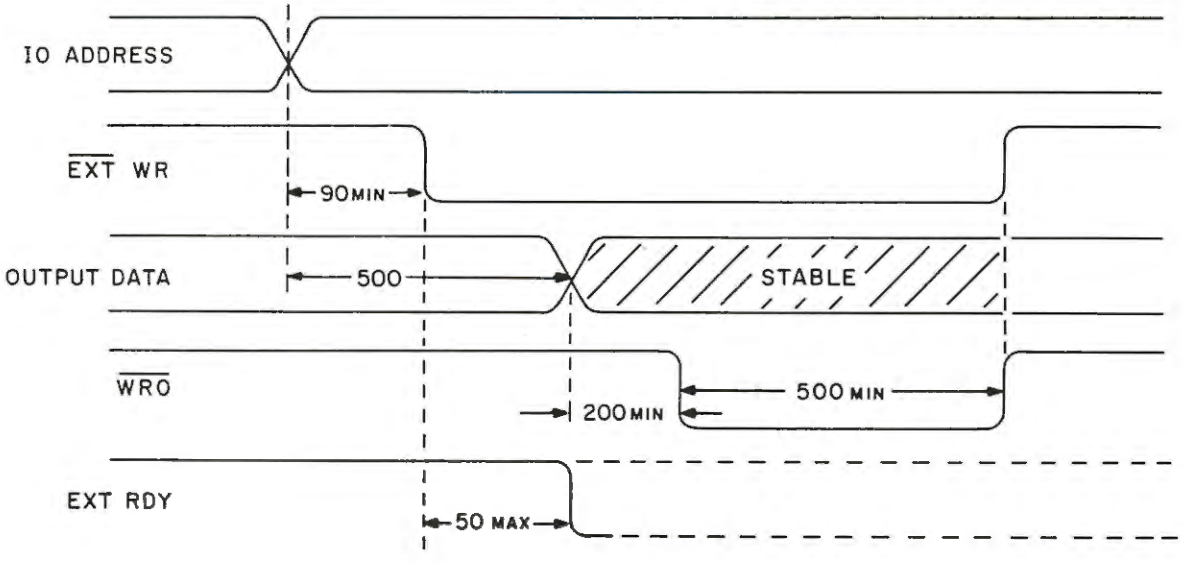


Figure 4.2. Output Timing

4.2 External Device Specifications

All equipment designed to operate on the external bus should adhere to the following specifications:

1. All voltage levels must be TTL compatible.
2. All receivers should present no more than one standard TTL load to the bus.

   Note: Address line A_{05} is a special case. This line should contain a 3.3k pull-up resistor to +5vdc. This resistor will help insure that the equipment will operate with other PLATO terminals.
3. All transmitters must be tri-state and be capable of driving 20 TTL loads. Recommended types are National 8097 or 81LS97, Signetics 8T97, and TI74367.
4. All equipment must look at the address lines at all times and must place signals on the bus only when the equipment address appears on the bus.
5. No equipment should interfere with bus operation when the power is removed.
6. Two identical connectors should be provided on each device to permit "daisy chaining" of the bus.
7. All cables should be twisted pair type.
8. The device address should be switch selectable so that it can be easily changed later.
9. The EXT RDY line will stop the 8080 on IO operations if it is held low. Extreme care should, therefore, be exercised in its use. If possible, the equipment should be designed such that it is always ready to send or receive data. The EXT RDY line may then be left disconnected.
10. The pin assignment for the IO connector is shown in Table 7. The mating connector is ITT Cannon DBC-25s or equivalent.

Pin	Function
1	Gnd
2	Address line A_{00}
3	Address line A_{01}
4	Address line A_{02}
5	Address line A_{03}
6	Address line A_{04}

Pin	Function
7	Address line A_{05}
8	$\overline{\text{EXT WR}}$ (external write address)
9	$\overline{\text{WRO}}$ (write output)
10	$\overline{\text{EXT RD}}$ (external read address)
11	EXT RDY (external ready)
12	$\overline{\text{EXT 0}}$ (external interrupt)
13	Gnd
14	Gnd
15	D_{00} data bit 0 (lsb)
16	D_{01} data bit 1
17	D_{02} data bit 2
18	D_{03} data bit 3
19	Gnd
20	Gnd
21	D_{04} data bit 4
22	D_{05} data bit 5
23	D_{06} data bit 6
24	D_{07} data bit 7 (msb)
25	Gnd

Table 7. External IO Connector