III. THE VIO BOARD (VIO-D)

III. THE VIO BOARD (VIO-D)

A. THEORY OF OPERATION

1. HARDWARE OVERVIEW

The VIO interfaces to the computer as a 4K byte block of memory. This 4K block includes a 2K byte static video refresh memory, the 2K bytes of VIOROM firmware, and a memory-mapped command port. Figure III-1 indicates the standard VIO memory space configuration. A Block Diagram of the VIO hardware is shown in Figure III-2.

VIO Processes

The operation of the VIO can be broken down into four basic processes: Character storage, display control, refresh display and firmware access.

> Character Storage: Characters to be displayed are stored into the video refresh memory by the system CPU. The select and control signals necessary to effect this transfer are generated by the VIO's Bus Interface.

> Display Control: The system CPU will assert control over the visual attributes of the screen display by outputting a command byte to the memory-mapped control port of the VIO. This command byte will be latched by the command logic and will specify to the Video Refresh Logic the display parameters (line length, page length, and inverse video).

> Refresh Display: The Video Refresh Logic will take characters from the video refresh memory and create the necessary composite video signal to allow the characters to be displayed on an external video monitor device.

> Firmware Access: Since the VIO resides in the system memory space, the CPU will execute the VIOROM firmware as if it were a program stored in the system memory. The VIO's Bus Interface allows the CPU to access the VIOROM by providing the necessary select and control signals.

2. VIDEO REFRESH MEMORY

The 2K bytes of video refresh memory is implemented with sixteen 2102 memory chips (U11-U18 and U28-U35). Characters to be displayed are stored in this memory in order beginning with the character in the top left screen location and ending with the character which appears in the lower right corner of the display

The video refresh memory is accessible both by the system CPU as well as by the video refresh logic. Therefore, video refresh memory addressing is achieved through the use of a separate MA bus which may be driven by the S-100 address line during a CPU access, or by the video

SYSTEM MEMORY

FFFF ───→

FIRMWARE
(2K)

F7FF ───→ COMMAND PORT

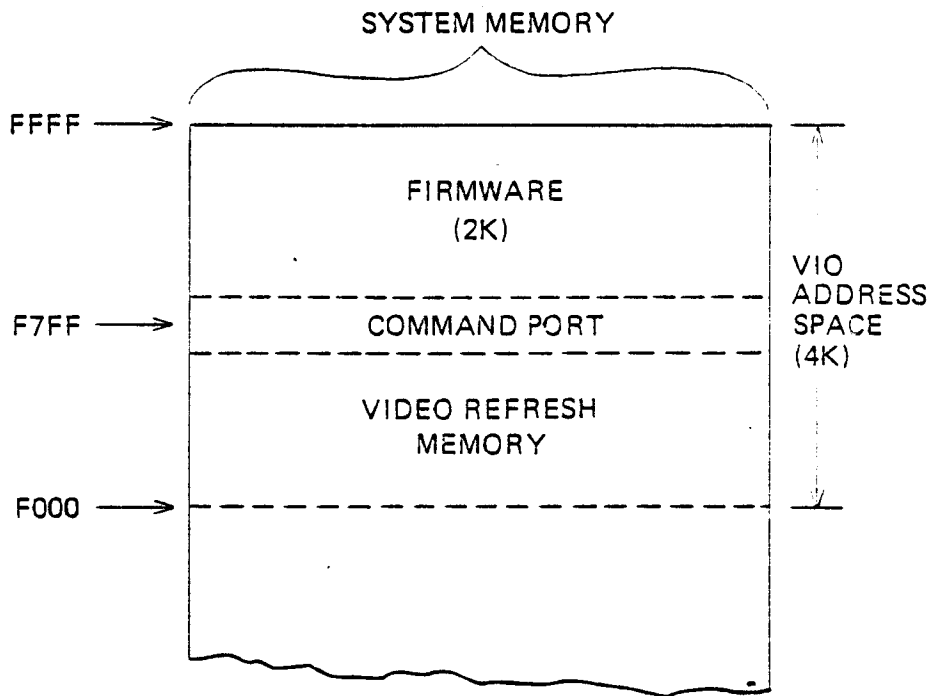VIDEO REFRESH
MEMORY

F000 ───→

VIO
ADDRESS
SPACE
(4K)

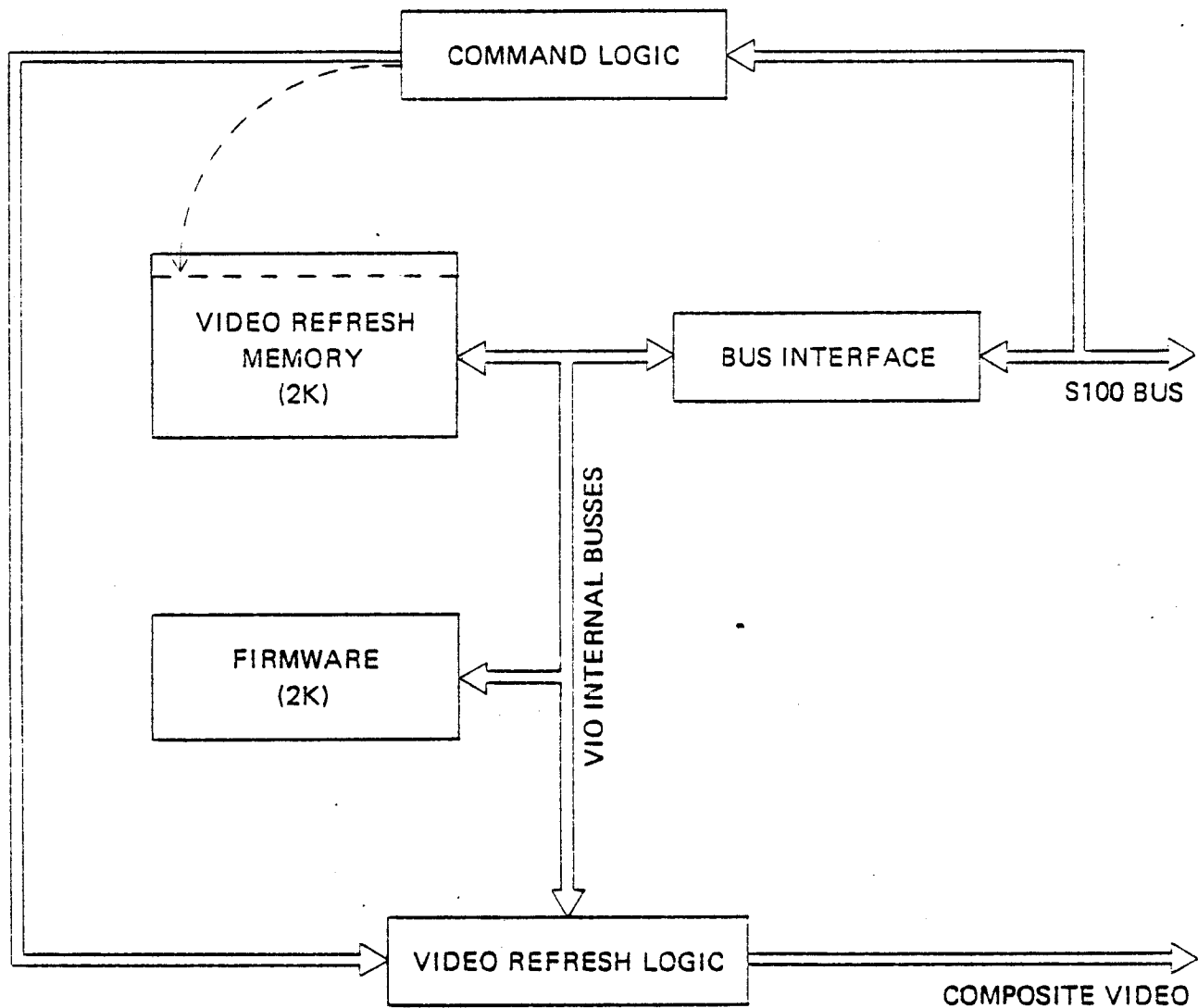FIGURE III-1 VIO MEMORY SPACE (STANDARD CONFIGURATION)

FIGURE III-2 VIO BLOCK DIAGRAM

refresh logic during refresh processes. The necessary multiplexing is achieved with tri-state drivers.

During a CPU write to the refresh memory, data from the system CPU is bussed directly to the data input pins of the memory chips via the S-100 DO bus.

Data out from the video refresh memory appears on the internal PCH bus. During a CPU read of the video refresh memory, the PCH bus drives the internal DB bus which in turn drives the S-100 Data In (DI) bus. During a video refresh, data on the PCH bus is latched by the video refresh logic.

3. VIOROM

The VIOROM firmware resides in the ROM at U46. Since the VIOROM is only accessed by the system CPU, its address lines are driven directly by the S-100 address bus.

During a CPU read of the VIOROM, the data from the PROM chip drives the VIO's DB bus which in turn drives the S-100 DI bus through tri-state drivers.

4. BUS INTERFACE

The VIO's S-100 bus interface provides the control and select signals necessary to effect a CPU transfer or access.

When the VIO is used in a megabyte address space (with IMM), the occurrence of A16-A19 (high) and /SINP, /SOUT and /SINTA indicates a CPU access in the top page of memory. The resulting signal enables a 74LS85 comparator at U50.

In a 65K address space, the occurrence of /SINP, /SOUT, and /SINTA is used to directly enable the 74LS85 comparator at U50. The comparator then compares A12-A16 to the settings of the address select switches (SW1). The comparator will drive its "=" output high only when a CPU access to the selected 4K block occurs.

This "=" output is further combined with A11 to determine if the CPU access is to the video refresh memory or to the VIOROM. The indication that the VIOROM is being accessed is used to directly drive the signal ROMSEL to enable the VIOROM at U46. The indication that the access is to the video refresh memory is latched at U40 by the user selected edge of the processor status strobe to prevent transients from interfering with the display. The /Q output of the latch U40 is the signal RAMSEL and is used to enable the refresh memory chips.

In addition to the ROMSEL and RAMSEL select signals, the VIO's bus interface must also provide the necessary control signals to effect a memory transfer.

The WRITE control signal is generated by gating /RAMSEL with /PWR at U26. The resulting signal /RAMWR is active low only during a CPU write to the video refresh memory. /RAMWR directly drives the R/W inputs of the memory chips.

If the wait state option has been selected, wait states are generated by reflecting PWAIT on

the PRDY line when the board is selected. The "not ready" signal is extended using a 74LS74 flip flop (U40) clocked by 02.

5. COMMAND LOGIC

The command port is memory mapped to the top location of the 2K byte video refresh memory space. A0-A10 is ANDed with RAMWR to detect a CPU write to the command port address. This signal then clocks DO0-DO4 into a 74LS74 latch (U44). The outputs of the four bit latch form the control signals /W80, /L24, MD0, MD1 and VIDREV; and are used by the video refresh logic.

/W80 controls line length; /L24 controls page length; MD0 and MD1 are encoded and select screen blanking, the character set and character-by-character reverse video; and VIDREV controls full screen reverse video.

6. VIDEO REFRESH LOGIC

A block diagram of the Video Refresh Logic is shown in Figure III-3. The timing and refresh address logic together generate the sequence of addresses which allow characters to be taken from the Video Refresh Memory. Data from the Refresh Memory (the encoded characters to be displayed) appear at the inputs of the character generator, where they are mapped into the specific pattern of "dots" which will form the visual representation of the characters. The video generation logic then combines the outputs of the character generator with the sync and blanking signals from the Timing Logic to produce the composite video signal required by the video monitor.

a) Timing Logic

The timing circuitry essentially consists of a 12.2304 MHz oscillator and a modulo-203840 counter chain. The counter chain consists of modulo-7, modulo-112 and modulo-260 sections cascaded to provide the 1.7472 MHz character clock, 15.600 KHz horizontal sweep synchronization and blanking and 60.00 Hz vertical sweep synchronization and blanking. This circuitry also provides outputs which specify the current scan position. A block diagram of this circuitry is shown in Figure III-4 and it is described in detail in the following paragraphs.

1) DOT CLOCK:

The dot clock (Figure III-5) is a crystal-controlled oscillator which generates the basic 12.2304 Mhz timing reference DCK1 for video generation. It consists of two sections of a 74LS04 cascaded to form a non-inverting amplifier. A series resonant crystal establishes the feed- back path to cause oscillation. A third section of the 74LS04 buffers the oscillator output. A 74LS74 D-flip-flop divides the 12.2304 Mhz signal by two to produce the 6.1152 Mhz clock signal DCK2 required for generation of double width (40 per-line) characters. The reset input of this flip-flop is connected to the composite sync signal /PCOMPSYNC to ensure that the flip-flop starts each line in the zero state (DCK2=1).
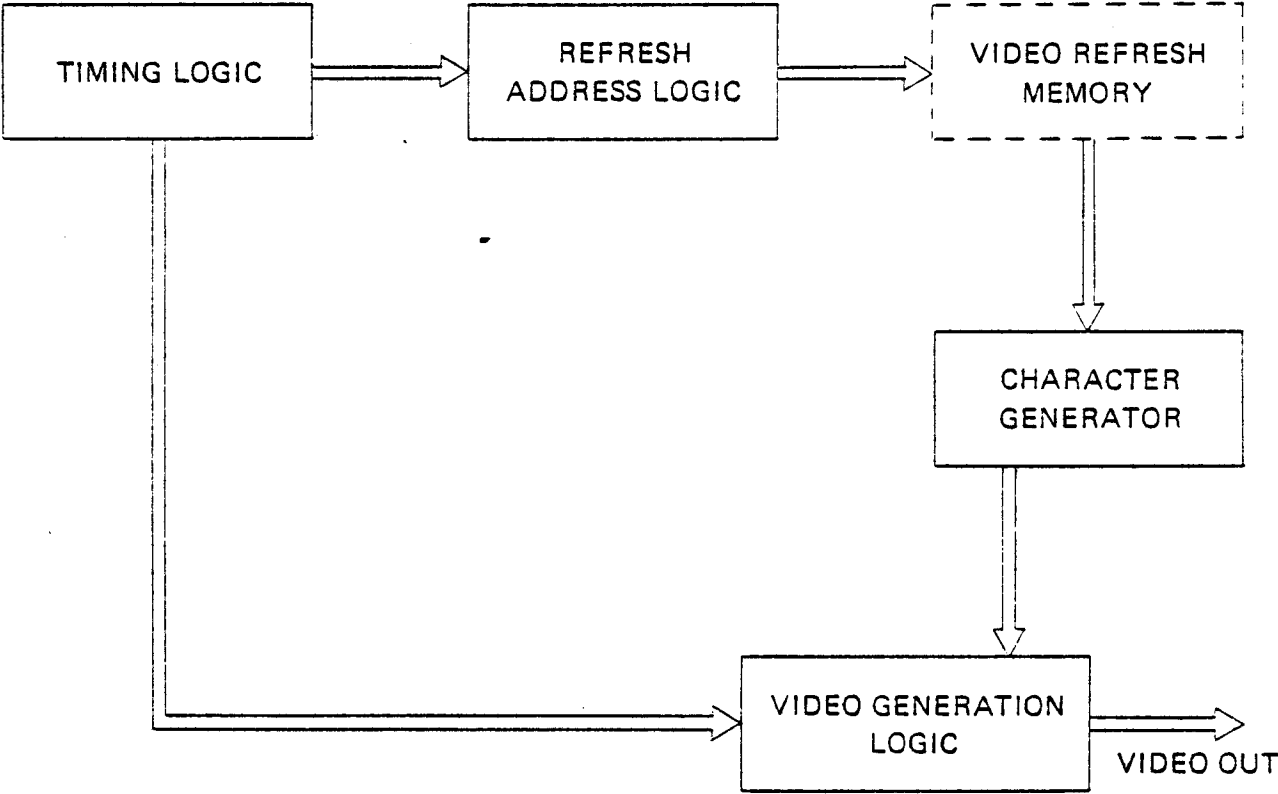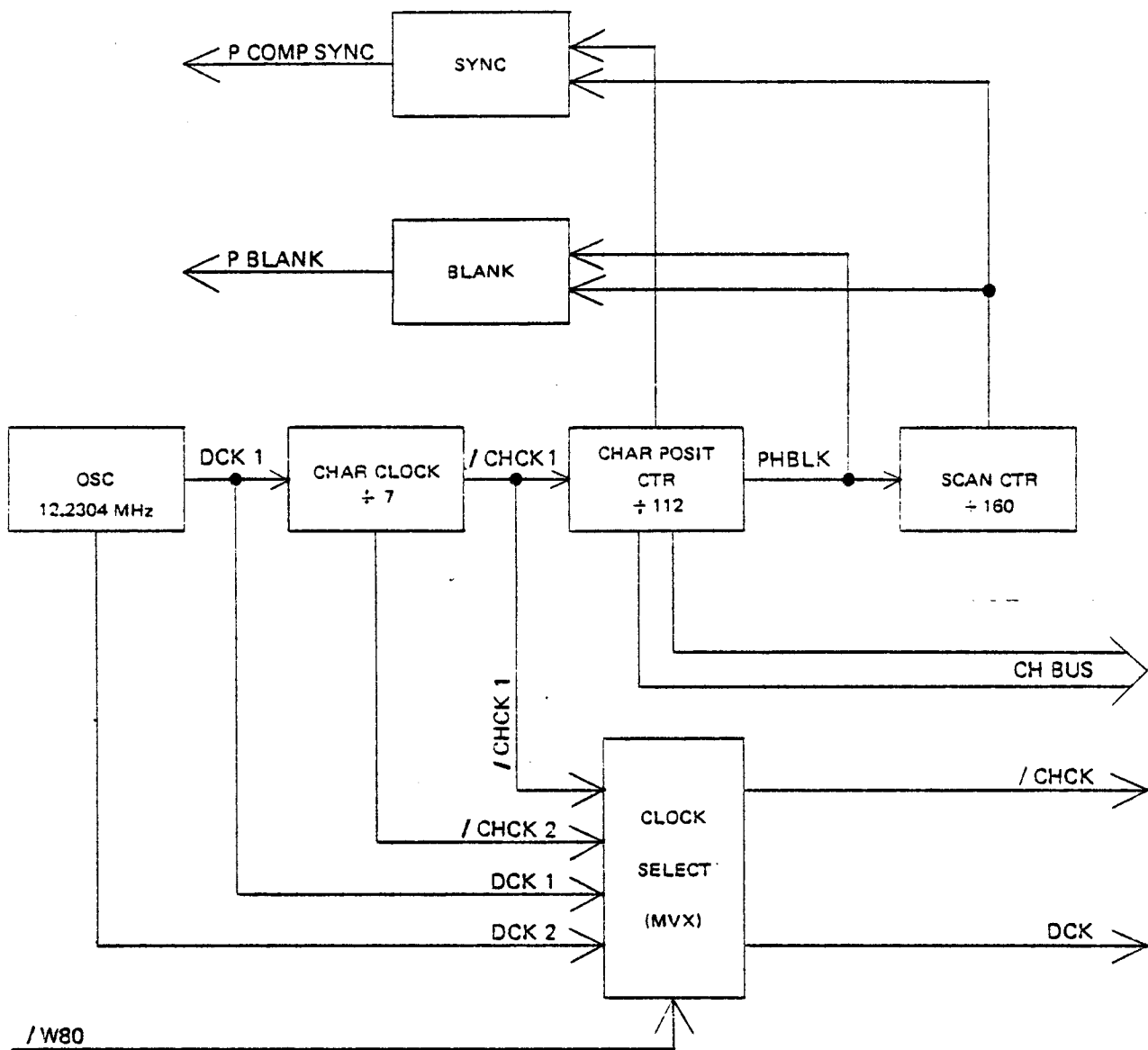
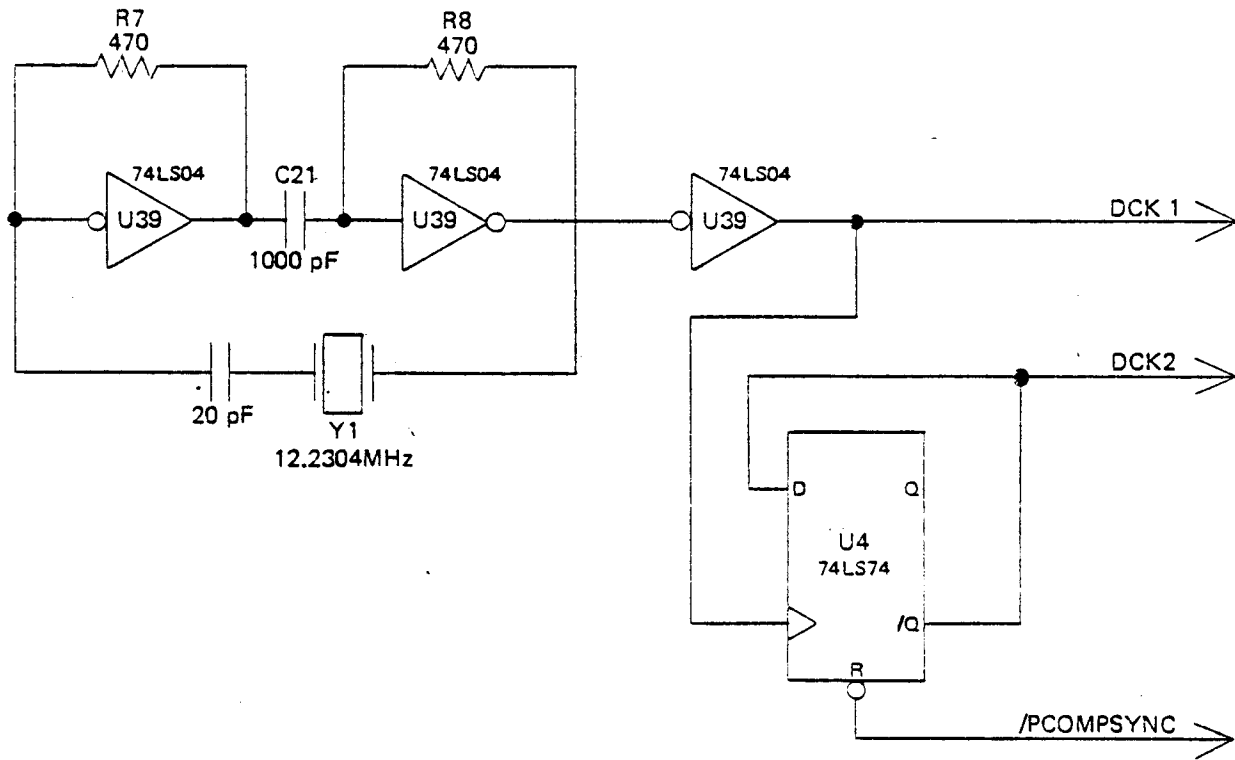FIGURE III-3 VIDEO REFRESH LOGIC

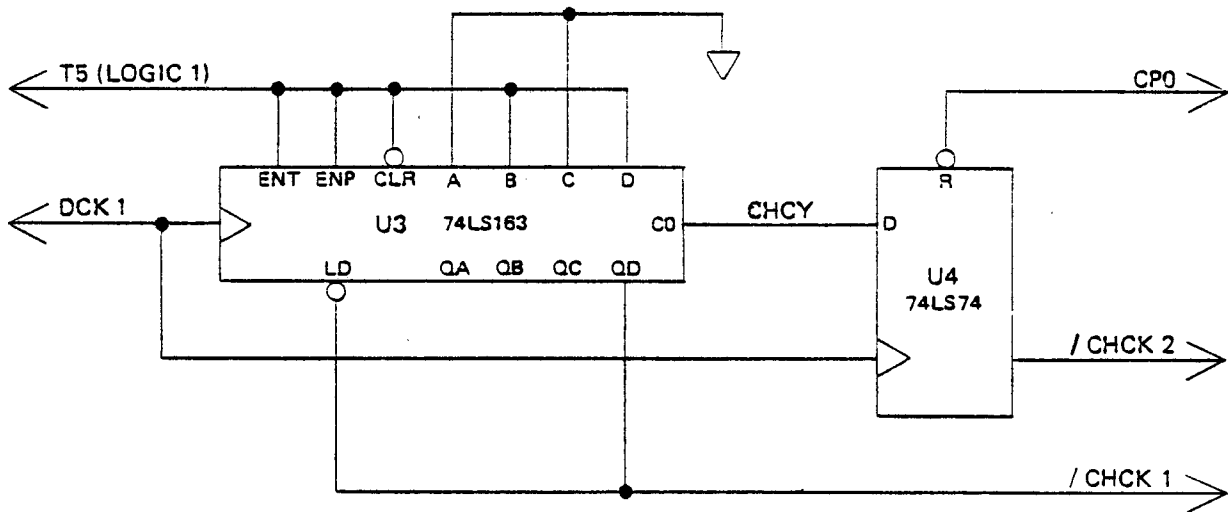FIGURE III-4 TIMING BLOCK DIAGRAM

FIGURE III-5   DOT CLOCK

FIGURE III-6   CHARACTER CLOCK

III – 10

2) CHARACTER CLOCK: The character dot matrix is seven dots wide so the character clock frequency is one-seventh that of the dot clock. It is generated by dividing DCK1 with a 74LS163 configured as a modulo-7 counter (Figure III-6). The counter is pre-loaded with a count of 10 then allowed to count through 15 overflowing to a count of zero. The QD output pulls the /LOAD input low causing a count of 10 to be again loaded. The QD output is high for the first six states and low only for the seventh (zero) state. This is used as the character clock output /CHCK1. A 74LS74 flip-flop generates a signal at half the frequency of /CHCK1, ie., the required character clock for double-wide characters /CHCK2. It is a one-bit shift register which is held in the reset state by CP0 (described in the next paragraph) during the first half of a double wide character. During the second half, the carry output of the 74LS163 CHCY is delayed one period of DCK1 by the 74LS74 to produce /CHCK2. Note that the low-to-high transitions of /CHCK2 are synchronized with the same edge of /CHCK1 and that the leading edge of /CHCK2 has the same timing relationship to DCK2 as /CHCK1 has to DCK1.

3) CHARACTER POSITION COUNTER: The character position counter (Figure III-7) counts the 80 character positions in each scan and generates the timing for blanking during horizontal retrace. It consists of two 74LS163's configured as a modulo-112 counter with input /CHCK1. The states zero through 79 correspond to the 80 character positions. State 79 is decoded by the 74LS11 and the associated inverters. This causes the /LOAD inputs of the 74LS163's to go low during state 79 and 224 to be loaded as the next state. The count continues for 32 states and over-flows back to the zero state. The states 224 through 255 correspond to the horizontal blanking interval. The counter's most significant bit is high only during these states so it's output is used as the horizontal blanking signal PHBLK. The remaining seven outputs drive the character position bus CP0-CP6.

4) SCAN COUNTER: The 240 displayed scans and 20 blanked scans are counted by the modulo-260 counter shown in Figure III-8. It consists of modulo-10 and modulo-26 cascaded counters. The modulo-10 section is a 74LS162 decade counter operating in its fundamental mode. The modulo-26 section consists of two 74LS163's. The counters are loaded with 232 and the states 232 through 255 correspond to the 240 (24 X 10) displayed scans. The counter overflows to the zero state and states zero and one correspond to the twenty blanked states. The one state is decoded by the 74LS00 and its associated inverter. The decoder causes the /LOAD inputs of the counters to be low during the one state and 232 to again be loaded as the next state. The most significant bit is low only during vertical blanking so its inverted output is used as the vertical blanking signal PVLK.

5) SYNC GENERATION: The horizontal and vertical sync pulses are generated by the circuit shown in Figure III-9. The horizontal sync signal PHSYNC is generated by using a 74LS74 flip-flop to essentially AND together CP3, /CP4 and PHBLK. The resulting signal is high only during the second quarter of PHBLK. The vertical sync signal PVSYNC is generated by using another section of a 74LS74 as a one bit shift register. The first half of PVBLK is decoded then phase-shifted by VCK. Note that PVSYNC is longer than one-half the length of PVBLK because of the phase relationship
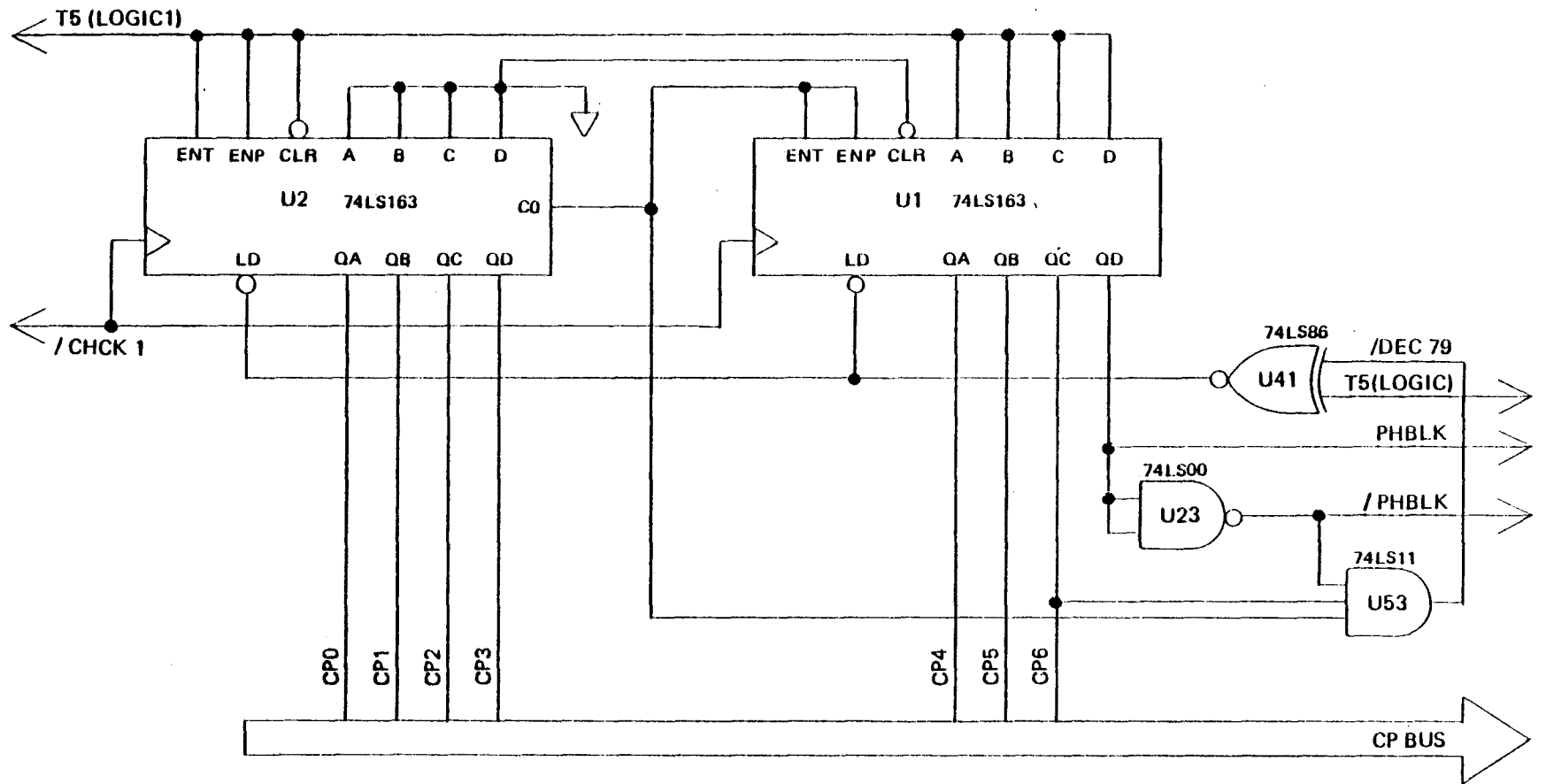
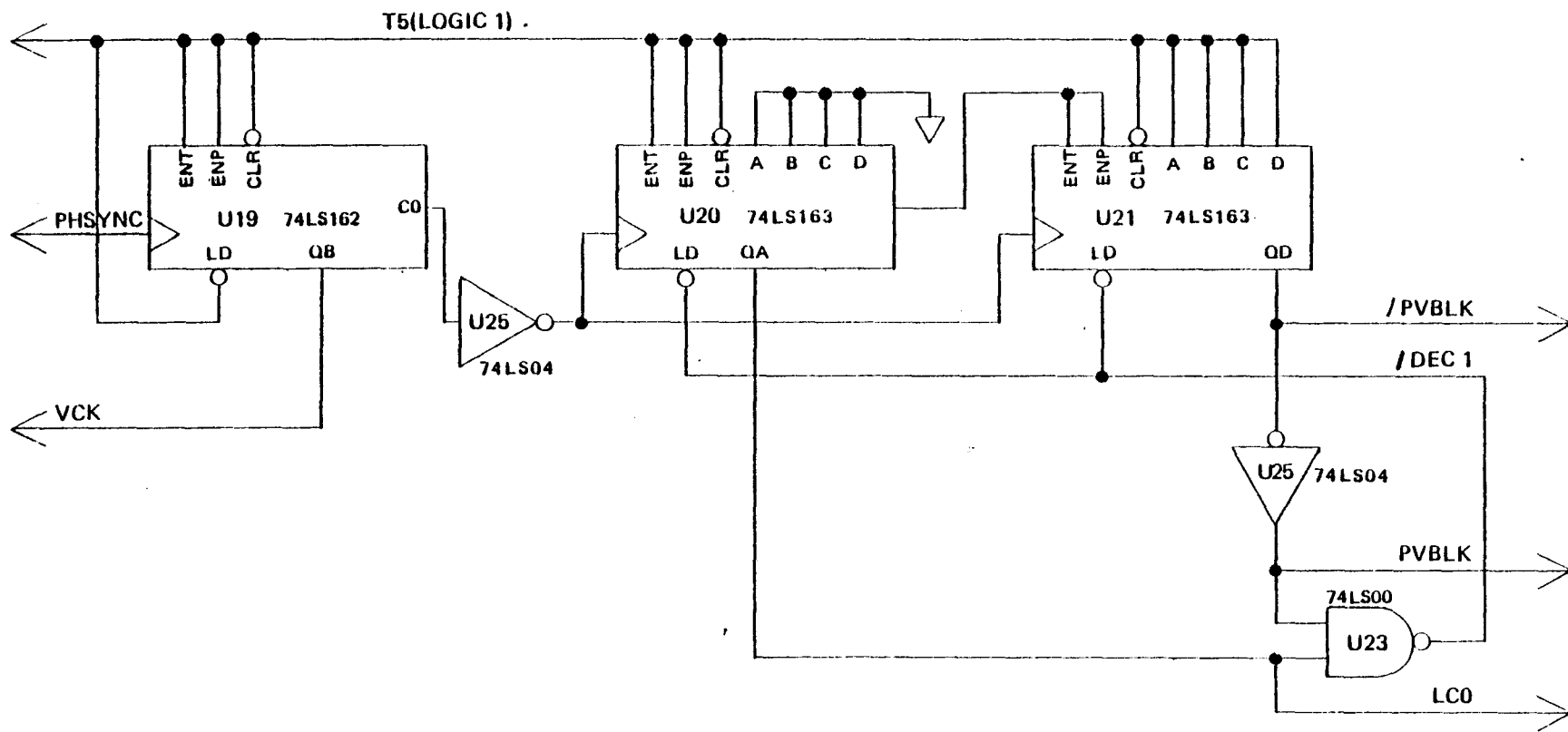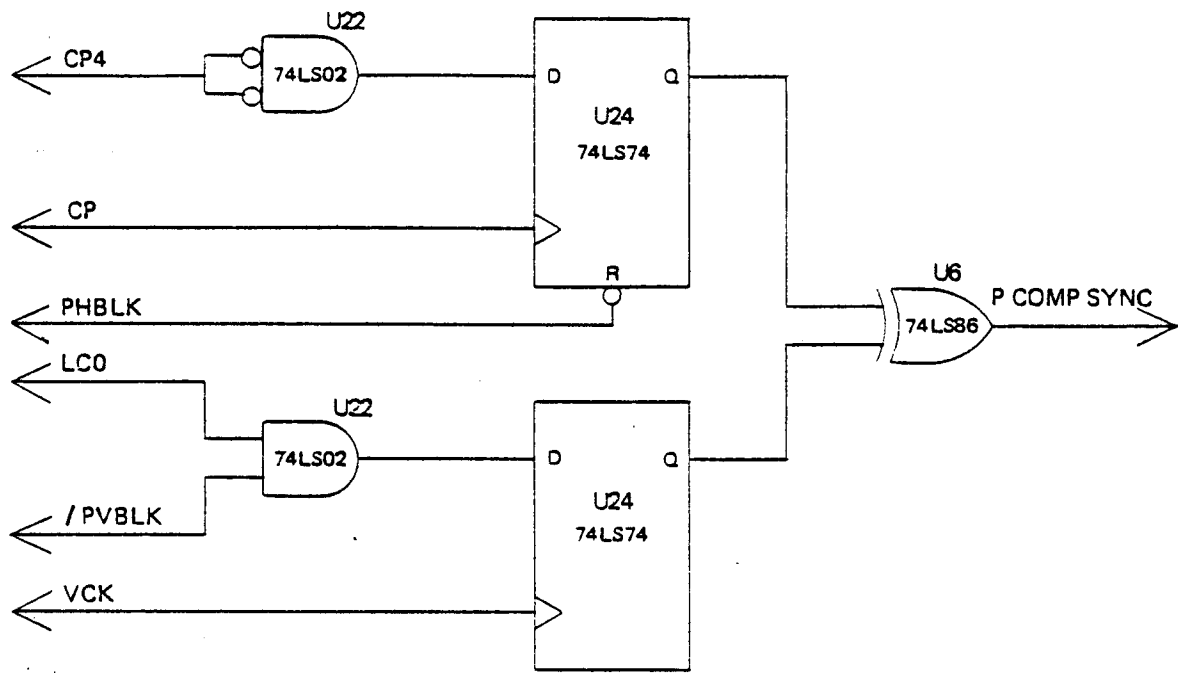FIGURE III-7 CHARACTER POSITION COUNTER

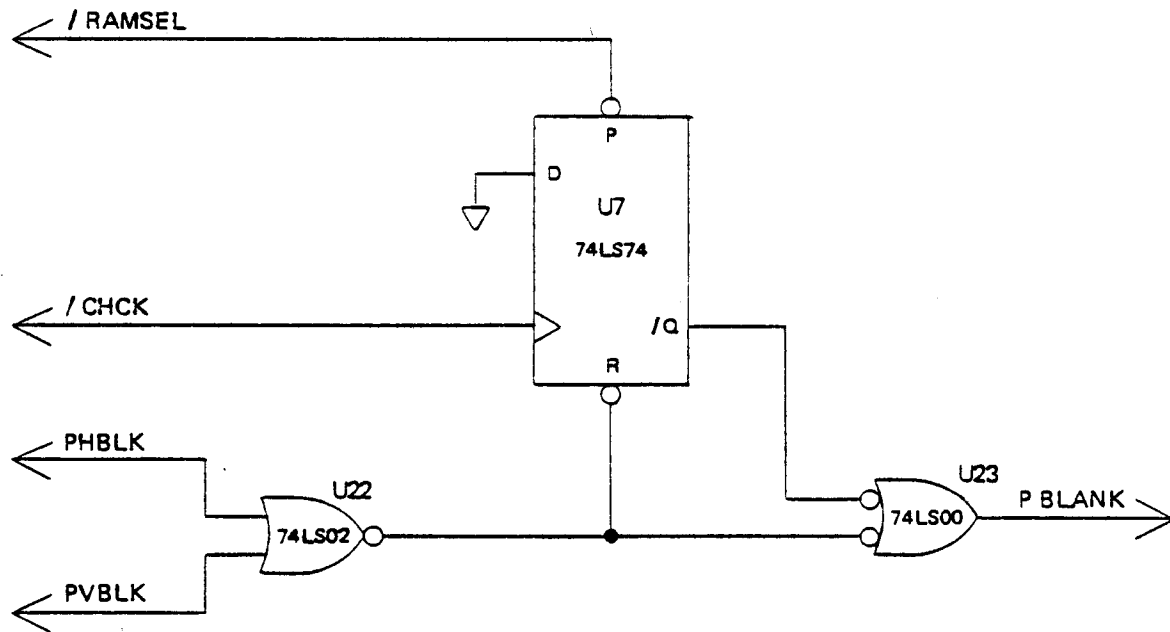FIGURE III-8   SCAN COUNTER

FIGURE III-9 SYNC GENERATION

FIGURE III-10 BLANKING GENERATION

between VCK and LCO. The sync pulses are XOR'ed to obtain the composite sync signal PCOMPSYNC. By combining the pulses in this way there is still horizontal sync information available during the vertical sync pulse.

6) BLANKING GENERATION: The horizontal blanking signals (PHBLK and PVBLK) are generated as described above. They are combined in the circuit shown in Figure III–10. In addition this circuit blanks the display when the refresh memory is accessed by the computer. PHBLK and PVBLK are OR'ed with the output of a 74LS74 flip–flop. The flip–flop is set by a computer access of the refresh memory (/RAMSEL low) and is reset by either /CHCK making a low–high transition or by either PHBLK or PVBLK being high. The result is that the circuit output PBLANK is high whenever PHBLK or PVBLK is high and during the short interval when a normally unblanked character is interrupted by a memory access.

b) Refresh Address Circuitry

A significant portion of the VIO's circuitry is associated with providing the refresh memory with appropriate address information via the MA bus. Normally the address provided is the address of the next character to be displayed. The characters are taken in order from memory regardless of screen format. Character addresses are computed from character position in a line, line number and screen format. When the VIO's refresh memory is being accessed by the system, the address on the MA bus is the same as the least significant eleven bits of the system bus. The multiplexing of this address and the character address is obtained using tri–state devices.

1) SCAN/LINE COUNTER: The circuit shown in Figure III–11 is the scan/ line counter. It counts the ten scans in each line and the 12 or 24 lines in each page. The scans are counted by a 74LS162 decade counter operating in its fundamental mode. When a 24 line format is selected the count is advanced once for each raster scan. When a 12 line format is selected the count is prescaled by a 74LS107 J–K flip flop configured as a modulo–2 divider. In this case the count is advanced once for every other raster scan. The outputs of the 74LS162 drive the SCN bus to provide scan number information to the character generator circuitry. The lines of characters on a page are counted by a five bit counter consisting of a 74LS163 and a 74LS107. The outputs of this counter drive the LN bus to provide information for computing character addresses.

2) CHARACTER ADDRESS COMPUTER: The refresh memory address of a character to be displayed is computed from its position in a line, the number of the line and the screen format. The character position counter provides position information via the CP bus. The scan/line counter provides line number information via the LN bus. The screen format is provided by the control signals /L24 and /W80 from the command latch. The four least significant bits of the character address are computed by a linear two to one scaling of the five least significant bits of the CP bus. This is accomplished using 74LS367 tri–state drivers configured as a multiplexor controlled by the line length format (80 or 40 characters). When a 40 character line is selected, bit 3 is inverted for odd numbered lines by XOR'ing this bit with LN0.
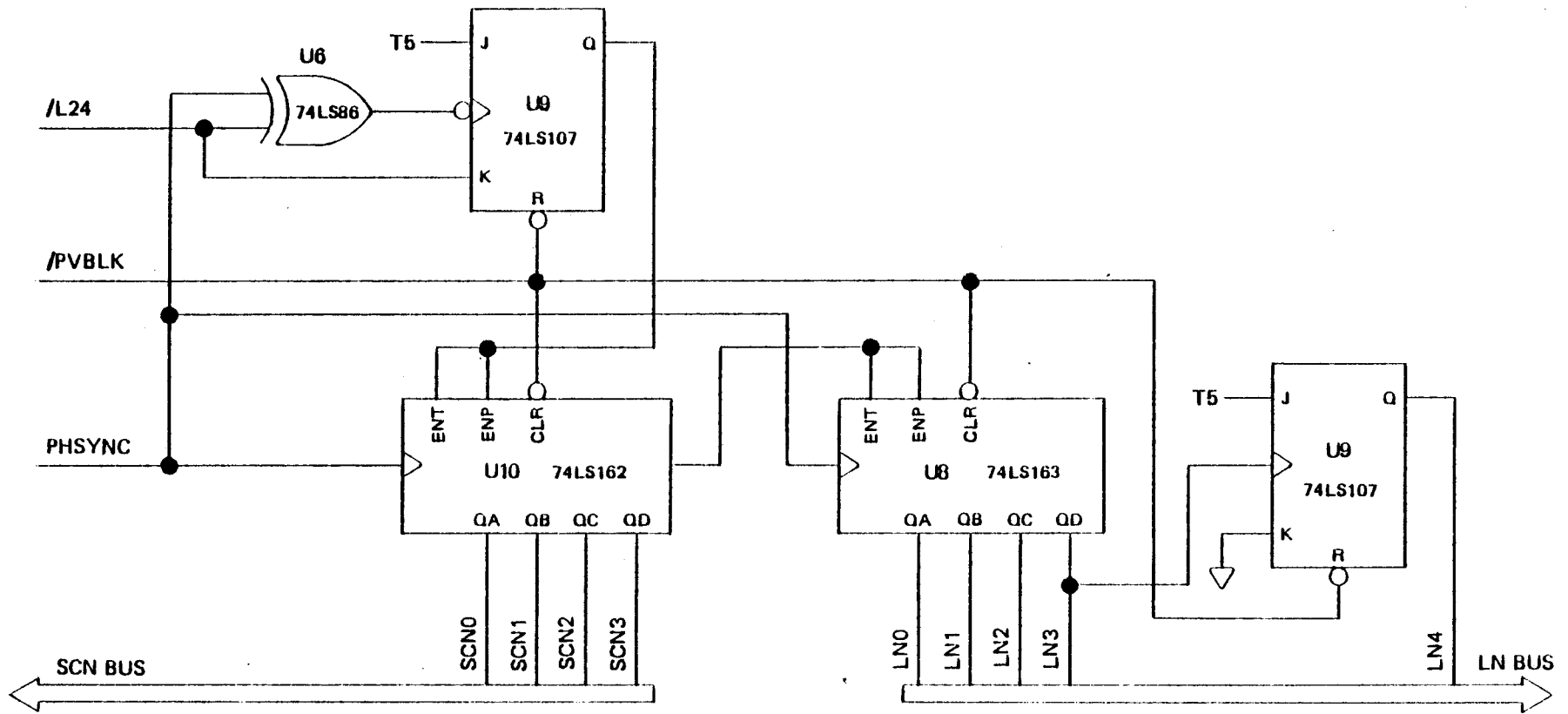
FIGURE III-11   LINE SCAN COUNTER

The high order character address bits are computed using a look-up table implimented in a 512x8 bit PROM.

All of the output structures of the character address computer are tri-state. When the refresh memory is being accessed by the system, all of these outputs are disabled and the MA bus is driven by the system address bus.

c) CHARACTER GENERATION

The character generation circuitry translates the character codes stored in the refresh memory into the dot patterns which form the desired character. The actual translation is performed using 2308/2708 ROM/EPROM's as look-up tables. The data entered in the table is the encoded character (CH bus) and the matrix row number (SCH bus). The data output is seven bits representing the seven dot positions for a row of that character. The ROM's are configured as a 7x10x256 bit memory with two 74LS32 gate sections and two 74LS04 inverter sections providing select decoding.

The output of the refresh memory (PCH bus) is first latched in 74LS174 flip-flops. The output of the latch (CH bus) is the encoded character provided to the character generator ROM's. The output of the ROM's is latched by the parallel input of a shift-register (part of video generation circuitry described below). Both latchings are clocked by /CHCK. The intermediate 74LS174 latch is necessary because the total access times of the refresh memory and the ROM's exceed one character time. A side effect of using this technique is that there is a two character time delay from the time when the address of a character is placed on the MA bus to the time when the resulting dot pattern is output by the VIO. As a result, the blanking and sync signals must also be delayed by two character times. This is accomplished using 74LS174 and 74LS74 flip-flops as two bit shift-registers.

d) VIDEO GENERATION

The video generation circuitry serializes the dot patterns output by the character generation circuitry, combines the resulting video with the blanking and sync signals and outputs the combined video signal to the monitor. The parallel to serial conversion is performed by a 74LS166 shift-register. As described above, the parallel output of the character generator ROM's is latched in the shift-register by /CHCK. The patterns are shifted out, clocked by DCK. This raw-video signal is XOR'ed with CH77, the most significant bit of the encoded character. The resulting signal is used for character-by-character reverse video. The 74LS153 mode data-selector selects either this signal or the raw video. The selected signal is XOR'ed with VIDREV for full-screen reverse video.

The video signal is gated with the blanking signal BLANK by a 74LS32 then combined with the sync signal COMPSYNC in a voltage adder circuict. A common-emmiter 2N3904 amplifier provides the impedance matching for the VIO's 72 ohm output.
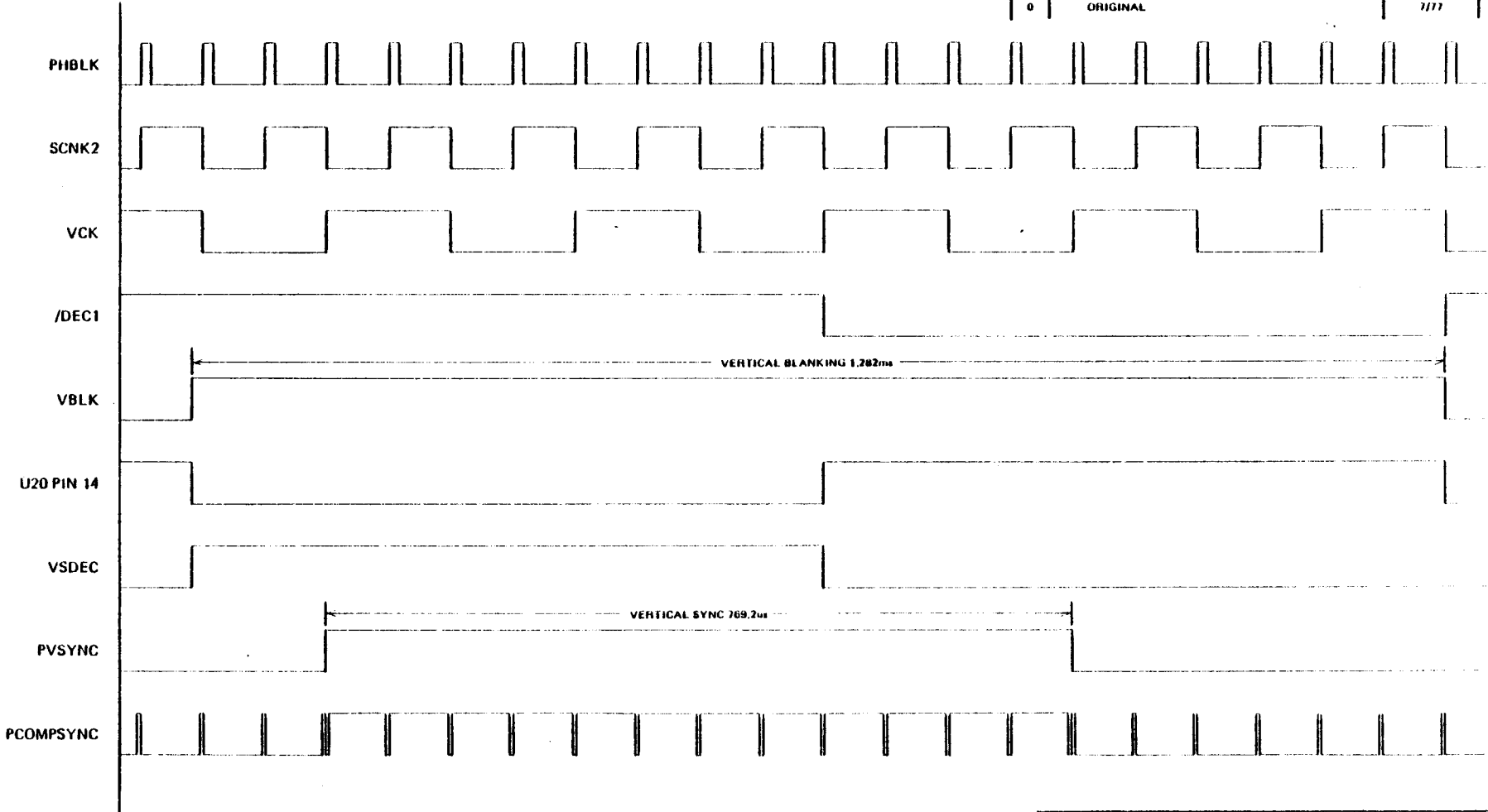
/CHCK1

CP0

CP1

CP2

CP3

CP4

/DEC79

PHBLK

PHSYNC

HORIZONTAL BLANKING 18.315us

HORIZONTAL SYNC 9.1576us

BISHOP GRAPHICS, INC.
REORDER NO. 20621

Waveform timing diagram with the following signal labels (top to bottom): PHBLK, SCNK2, VCK, /DEC1, VBLK, U20 PIN 14, VSDEC, PVSYNC, PCOMPSYNC.

Annotations on the diagram:
- VERTICAL BLANKING 1.282ms
- VERTICAL SYNC 769.2us

REVISIONS			
LTR	DESCRIPTION	DATE	APPROVED
0	ORIGINAL	1/77	

DCK1

COMPSYNC

DCK2

CHCY

/CHCK1

CPO

/CHCK2

SINGLE DOT TIME (96.39ns)

DOUBLE DOT TIME (190.78ns)

SINGLE CHAR WIDTH

DOUBLE CHAR WIDTH

TOLERANCES UNLESS OTHERWISE SPECIFIED
FRACTIONS   DEC.   ANGLES

APPROVALS | DATE
DRAWN
CHECKED

VIO
DOT AND CHARACTER TIMING

SCALE | SIZE **B** | DRAWING NO

DO NOT SCALE DRAWING | SHEET

B. USER GUIDE

1. BOARD CONFIGURATION

To properly set-up the VIO board, the board jumpers and switches must be configured for the particular USER application. The board configuration instructions must be followed to insure a properly functioning unit.

SWITCHES 1-8 (SW1)

Switches one through eight are located in position SW1 to the left of U39. Each switch must be set according to the instructions detailed in Steps 1-5 below.

1) Set switches (SW1) 1, 2, 3, and 4 to the OFF position. The VIO board will now occupy the 4K block of memory beginning at address F000 hex.

   NOTE: Switches 1-4 correspond to address bits A12 to A15 respectively. Each switch should be in the OFF position to select a "1", or in the ON position to select a "0" for the appropriate address bit.

2) Set switch 5 (SW1) to the OFF position. The VIOROM firmware is now set to reside in the upper half of the associated 4K block of memory (F800-FFFF).

3) Set SWITCH 6 (SW1) to the ON position. The VIO board will now be selected when either the VIOROM or VIO RAM is being accessed.

TABLE III-1
Address Switches 1-4

ADDRESS	S1	S2	S3	S4
0000-0FFF	ON	ON	ON	ON
1000-1FFF	OFF	ON	ON	ON
2000-2FFF	ON	OFF	ON	ON
3000-3FFF	OFF	OFF	ON	ON
4000-4FFF	ON	ON	OFF	ON
5000-5FFF	OFF	ON	OFF	ON
6000-6FFF	ON	OFF	OFF	ON
7000-7FFF	OFF	OFF	OFF	ON
8000-8FFF	ON	ON	ON	OFF
9000-9FFF	OFF	ON	ON	OFF
A000-AFFF	ON	OFF	ON	OFF
B000-BFFF	OFF	OFF	ON	OFF
C000-CFFF	ON	ON	OFF	OFF
D000-DFFF	OFF	ON	OFF	OFF
E000-EFFF	ON	OFF	OFF	OFF
F000-FFFF	OFF	OFF	OFF	OFF
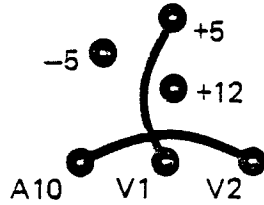
FIGURE III-12 POWER JUMPERS FOR 2316E ROM'S AND 2716/8716 EPROMS
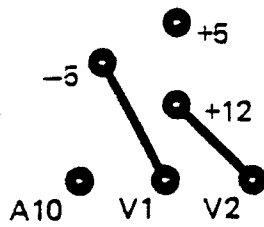

FIGURE III-13 POWER JUMPERS FOR 2308/8308 ROM'S AND 2708/8708 EPROMS
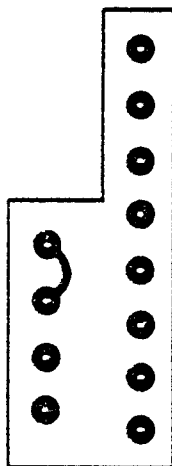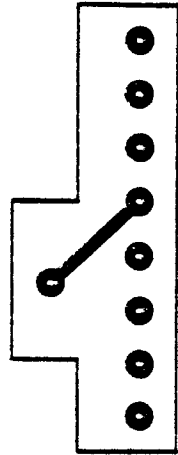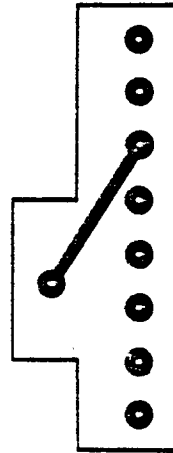
A16/A32/A65

FIGURE III-14 DYNAMIC RAM BOARDS. REMOVE JUMPER FROM A16 TO GROUND
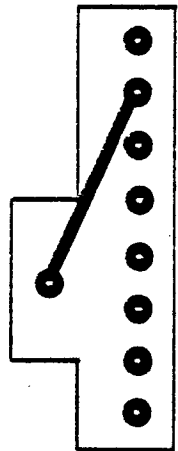ONLY IF USING A16 AS PHANTOM LINE.
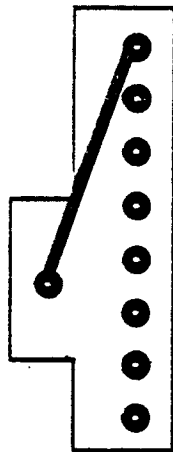
FIGURE III-15

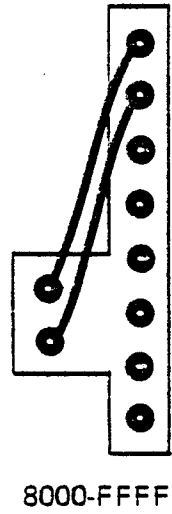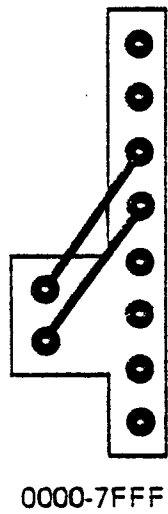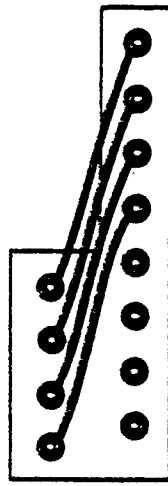## RAM16 B16 JUMPERS

0000-3FFF

4000-7FFF

8000-BFFF

C000-FFFF

NOTE: VIO USING A16 AS PHANTOM LINE MUST BE INSTALLED.

III – 28

FIGURE III-16

RAM 32 B32 JUMPERS

0000-7FFF   8000-FFFF

NOTE: VIO USING A16 AS PHANTOM LINE MUST BE INSTALLED.

FIGURE III-17
RAM 65 B65 JUMPERS

0000-FFFF

NOTE: VIO USING A16 AS PHANTOM LINE MUST BE INSTALLED.

4) Set SWITCH 7 (SW1) to the OFF position. The VIO memory will now operate with no WAIT states.

If it is desired to insert WAIT states into every VIO memory access, set SWITCH 7 to the ON position. Note that the memory devices supplied by IMSAI do not require WAIT states.

5) SWITCH 8 is not used and should be disregarded.

JUMPERS

The board jumper areas are clearly marked both on the board and on the Assembly Diagram. Each jumper area must be configured as required in Steps 6-9 below. Note that assembled VIO's are shipped with the jumpers described in Steps 6 and 7 installed.

6) Connect a jumper between pads "PH" and "B". This will allow the VIO memory to be accessed on the trailing edge of the processor status strobe (as required by 8085 type CPU's).

7) Connect jumpers V1 and V2 as shown in Figures III-12 AND III-13 for the particular type of ROM device used at U46.

CAUTION!! Incorrect connection of jumpers V1 and V2 may cause damage to the ROM. Make certain that these jumpers are correct for the type of ROM used before applying power to the board. Only the following types of EROM/ROM's may be used at location U46: 2708, 2308, 2716 or 2316E. Do not attempt to use any other types of ROM/EROM devices or damage to the device will result. Ensure that the jumpers do not short to each other, to any board traces, or to any IC pins.

8) If IMSAI RAM-16, RAM-32 or RAM-65 memory boards are not used in the system, leave jumper area "F" open.

If it is desired to allow the VIO Board to coexist with a full 65K of memory, consisting entirely of RAM-16's, RAM-32's, or a RAM-65 Board, install a jumper in Jumper Area "F", from the pad labelled "F" to the pad labelled "A16". On each memory board, install the jumper shown in Figure III-14, in the A16, A32, or A65 Jumper Area. Re-address each memory board to reside in the second 65K using the jumpers shown in Figures III-15, III-16, and III-17 (B16, B32, or B65 Jumper Areas).

NOTE: The memory boards are now configured to reside in the second 65K memory space. The VIO will disable the memory boards using address line A16 whenever the VIO is accessed. (A19 may be used in place of A16 if it is so desired). Note that in this configuration, if the VIO is removed from the system, the memory boards will not be accessible unless provision is made to externally drive A16.

9) If an IMSAI IMM board is not being used in the system, Jumper Areas C and D should be disregarded.

If an IMM board is being used, the VIO board may be addressed to reside in the top 65K of the Megabyte Address Space. To do this, cut the two traces connecting the pads labelled "C" and "D" to the pads labelled A15 and A14 respectively. Then install a jumper between the pad labelled "AA15" and the pad labelled "C". Also install a jumper between the pad labelled "AA14" and the pad labelled "D". Install a jumper at location "M".

Refer to the IMSAI IMM Documentation for further details.

2. VIO OPERATION WITH VIOROM

The VIOROM contains two separate and distinct programs. The first is a firmware driver which will allow a user program to easily control all VIO functions and screen displays, (new releases of IMSAI Software will include provisions for using the VIOROM Firmware). The second is a complete system monitor program, IMSAI VDP–80, which gives the user a number of extremely useful memory and input/output functions when the VIO and monitor are used with a keyboard or other input device.

Since operation with the VIOROM depends upon the Users application, the User may proceed in one of three ways. If it is desired to use the display functions of the VIO with existing User Software, proceed to "VIO Firmware Driver". If it is desired to exercise the display functions of the VIO and VIOROM Firmware Driver, proceed to "VIO Test Routine". If it is desired to operate the VIO as an Intelligent Video Terminal, proceed to VDP–80 Monitor.

a) VIO Firmware Driver

The VIO firmware driver can be called from the user program whenever it is desired to initialize the VIO, place a character on the screen display, or execute a control sequence.

1) SCREEN INITIALIZATION: To initialize the VIO, the user program should CALL the INITIALIZATION ENTRY POINT at F800 H. The firmware will then

    clear the screen
    put the cursor in the upper left corner (HOME)
    select an 80x24 screen format
    select UPPER CASE display (ASCII text mode)
    activate the screen scroll mode

Following the initialization routine, control will return to the user's calling program.

2) CHARACTER DISPLAY: To display a character on the screen, the user program must CALL the CHARACTER OUT entry point, F803H, with the character code (for the character to be displayed), in the Accumulator. Prior to the CALL, the stack must contain at least 30 bytes of available STACK SPACE. Once the character has been displayed, control will return to the user's calling program with none of the registers or status bits affected.

A list of character codes appears in Appendix 1.

The characters which may be displayed on the screen will depend entirely on the mode of operation selected by the user: ASCII text mode, extended text mode, or graphic

mode. These modes are selected with the ESCAPE SEQUENCES, described in the next section.

Note: characters may also be displayed by directly accessing the refresh memory (as described in section III-A,6-b).

ASCII TEXT MODE: When the ASCII Text Mode is selected, any of the 96 characters represented by the character codes 20-7FH (see Appendix 1) may be displayed through the firmware. This includes the standard 96 character ASCII set (upper and lower case plus punctuation). Character-by-character reverse video is available and both CONTROL CHARACTER COMMANDS and ESCAPE SEQUENCES will be accepted.

To display a character in the ASCII Text Mode, the User program must CALL the Character Out Entry Point, F803 H. Prior to the CALL, the lower 7 bits of the Accumulator must contain a valid ASCII Text Character Code. Bit 7 of the Accumulator will act as the character-by-character reverse video flag (0=positive video, 1=reverse video).

Displayable ASCII Text Character Codes consist of the lower 7 bits of CODES 20-7F H listed in Appendix 1. Note that these codes correspond to the standard 7 bit ASCII Codes 20-7F H.

For example, if we wanted to display an ASCII "1" in positive video, we would CALL the Character Out Entry Point, F803 H, with the code 31 H in the Accumulator. The same code, with bit 7 "on" would cause the ASCII "1" to be displayed in reverse video. Thus if the Accumulator contained a B1 H, the ASCII "1" would be displayed as a dark character on a light background (reverse video).

Note that character codes 00-1F H are not displayable through the firmware (in the ASCII Text Mode). However, they may be displayed by directly accessing the VIO's refresh memory (III-A,6-b).

In most cases, existing User Software may easily be modified to be used with the VIO.

1.   Locate the output driver routine.
2.   Substitute a CALL to the Character Out Entry Point (CALL F803 H), for the status port input instruction (directly preceding the output instruction to the terminal device).
3.   Ensure that the character to be output, is in the Accumulator prior to the CALL.

4.   "NOP" out the rest of the driver up to but not including any RET instruction.

For example, suppose the existing output driver appears as follows:

```
1950  DB 03     TOUT     IN   TTY+1        ;GET STATUS
1952  1F                 RAR               ;TEST IF READY
1953  D2 50 19           JNC  TOUT         ;LOOP IF NOT RDY
1956  F1                 POP  PSW          ;GET CHAR
1957  D3 02              OUT  TTY          ;OUT TO TERMINAL
1959  C9                 RET
```

This driver may be modified to be used with the VIO firmware as follows:

```
1950  F1           POP  PSW        .      ;GET CHAR
1951  CA 03 F8     CALL VIO               ;DISPLAY CHAR
1954  00           NOP
1955  00           NOP
1956  00           NOP
1957  00 00        NOP!NOP
1959  C9           RET
```

EXTENDED TEXT MODE: When the Extended Text Mode is selected, any of the 96 graphic characters, represented by the character codes C0–FFH (see Appendix 1) may be displayed through the firmware. Character–by–character reverse video is available and both CONTROL CHARACTER COMMANDS and ESCAPE SEQUENCES will be accepted.

To display a character in the Extended Text Mode, the User program must CALL the Character Out Entry Point (CALL F803 H). Prior to the CALL, the lower 7 bits of the Accumulator must contain a valid Extended Text Character Code. Bit 7 of the Accumulator will act as the character–by–character reverse video flag (0=positive video, 1=reverse video).

Displayable Extended Text Character Codes consist of the lower 7 bits of Codes A0–FF H listed in Appendix 1. Note that Codes A0–FF H correspond to a partial block drawing and a complete line drawing graphic character set. This set is a subset of the one available in the Graphic Mode, and it is fully explained in the Graphic Mode section (below).

For example, if we wanted to display in positive video, the vertical line segment represented by the Code C5 H in Appendix 1, we would CALL the Character Out Entry Point F803 H, with a 45 H in the Accumulator. The same Code with bit 7 "on" would cause the character to be displayed in reverse video. Thus if the Accumulator contained a C5 H, the vertical line segment would be displayed as a dark character on a light background (reverse video).

Note that Codes 80–9F H are not displayable through the firmware (in the Extended Text Mode). However, they may be, displayed by directly accessing the VIO's refresh memory (Section III–b,3).

While the Extended Text Mode currently allows graphic characters to be displayed, it is ideally suited for the display of foreign language fonts. This would require the upper half of the character generator PROM's to be appropriately reprogrammed.

GRAPHIC MODE: When the Graphic Mode is selected, any of 255 ASCII and graphic characters may be displayed (codes 00–1A H and codes 1C–FF H). The character codes appear in Appendix 1.

While this mode allows both ASCII and graphic characters to be displayed, it does not allow the use of character–by–character reverse video. Any new CONTROL CHARACTER COMMANDS will not be accepted. Any CONTROL CHARACTER COMMANDS in effect prior to the entry into the Graphic Mode will remain valid and will still affect character display. ESCAPE sequences will be accepted and provide the only means of control when in the Graphic Mode.

To display a character in the Graphic Mode, the User program must CALL the Character Out Entry Point F 803 H. Prior to the CALL, the Accumulator must contain a valid 8 bit Graphic Character Code.

All Codes 00–FF H (listed in Appendix 1) are displayable in the Graphic Mode, with the exception of Code 1B H. The character codes 00–1F H correspond to 32 special characters. The character codes 20–7F H, correspond to the standard ASCII codes 20–7F H. The characters represented by the codes 80–FF H, comprise a complete line drawing and block drawing graphics set.

For example, if we wanted to display an ASCII "A", we would CALL the Character Out Entry Point, F 803 H, with a 41 H in the Accumulator. Similarly, if we wanted to display the line segment represented by the Code C1 in Appendix 1, we would CALL the Character Out Entry Point F 803 H, with a C1 H in the Accumulator.

Note that Character Code 1B H is not displayable through the firmware (in the Graphic Mode). However, it may be displayed by directly accessing the refresh memory (Section III–b,3).

Codes 80–BF H correspond to the Block Drawing Graphic Set. The display grid for this set uses a 7x10 dot matrix font, equally divided into 6 cells or display areas as shown in Figure III–18. The characters in the Block Drawing Set consist of the patterns formed by using these six cells in various combinations. Bits 0–5 of the character codes 80–BF H correspond to the six display cells as shown in Figure III–18. A "one" in any of these bit positions will cause the corresponding cell to be used in the display character. For example, if it is desired to display the character formed by using cell 0 and cell 2, the character code will contain a "1" in bits 0 and 2. Bits 1, 3, 4, and 5 will be "0", since cells 1,3,4, and 5 are not used in the display character. Since bit 6 is a "0" and bit 7 is a "1" for all Block Drawing Characters, the resultant character code is B5 H (10000101).

Codes C0–FF H correspond to the Line Drawing Graphic Set. The display grid for this set uses the 7x10 dot matrix font to create the six line segments shown in Figure
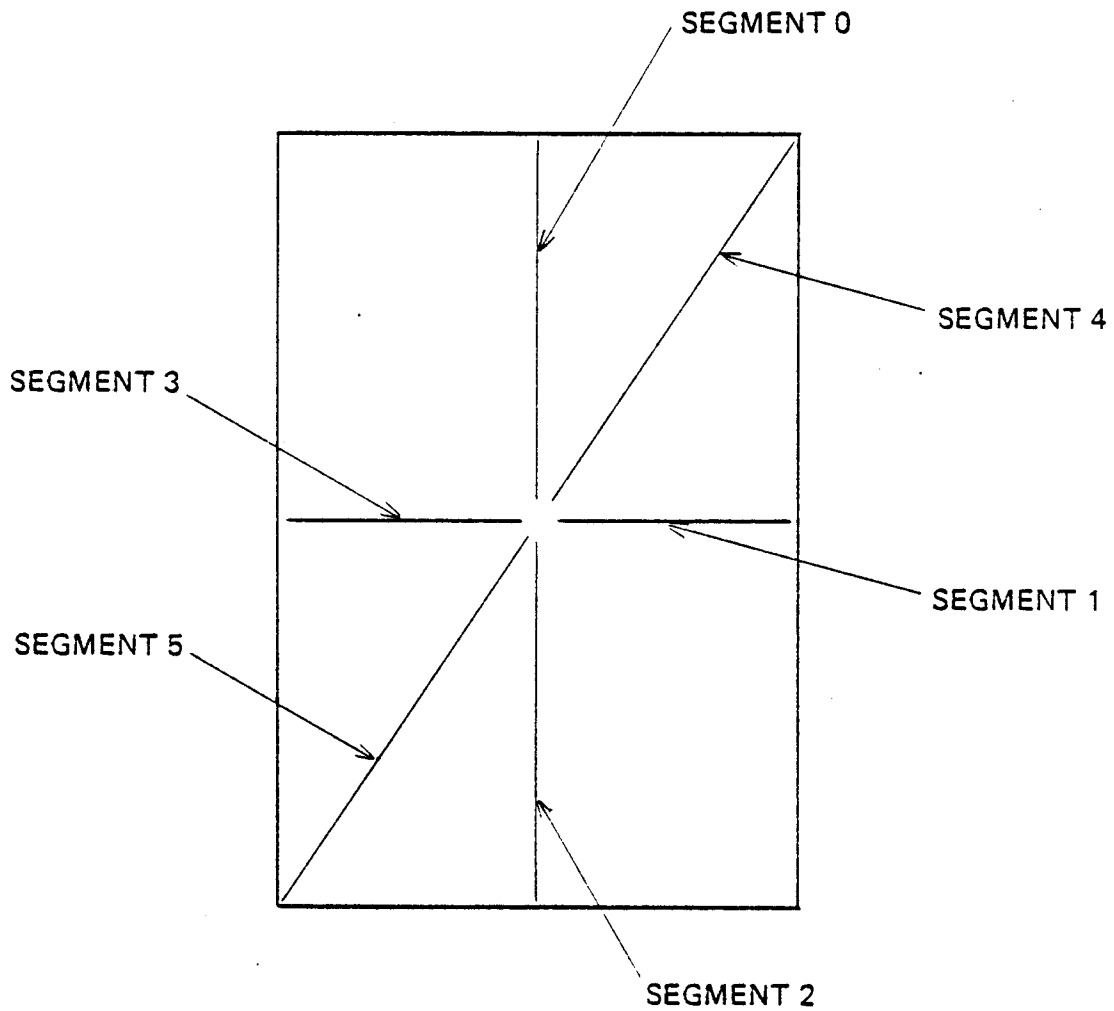
FIGURE III-18 THE CHARACTER CODE — DISPLAY RELATIONSHIP
FOR THE BLOCK DRAWING GRAPHIC SET

CELL 5	CELL 2
CELL 4	CELL 1
CELL 3	CELL 0

1	0	CELL 5	CELL 4	CELL 3	CELL 2	CELL 1	CELL 0

CHARACTER CODES 80—BF H.

FIGURE III-19   THE CHARACTER CODE — DISPLAY RELATIONSHIP
FOR THE LINE DRAWING GRAPHIC SET

1	1	SEGMENT 5	SEGMENT 4	SEGMENT 3	SEGMENT 2	SEGMENT 1	SEGMENT 0

CHARACTER CODES C0—FF H.

III-19. The characters in the line drawing set consist of the patterns formed by using these six line segments in various combinations. Bits 0–5 of the character codes C1–FF H, correspond to the six display segments as shown in Figure III–19. A "one" in any of these bit positions will cause the corresponding segment to be used in the display character. For example, if it is desired to display the character formed by using segments 1 and 2, the character code will contain a "1" in bits 1 and 2. Bits 0, 3, 4, and 5 will be "0", since segments 0, 3, 4, and 5 are not used in the display character. Since bit 6 is "1" and bit 7 is "1", for all Line Drawing Characters, the resulting character code is C6 H (11000110).

3) CONTROL SEQUENCES: There are two basic methods of controlling or modifying VIO operations: CONTROL CHARACTER COMMANDS and ESCAPE SEQUENCES.

CONTROL CHARACTER COMMANDS ———————————————

To execute a CONTROL CHARACTER COMMAND, the user program should CALL the CHARACTER OUT entry point at F803 H, with the desired control character in the Accumulator. Prior to the CALL, the stack must contain at least 30 bytes of available stack space. Following the execution of the command, control will return to the user's calling program with none of the registers or status bits affected. Note that CONTROL CHARACTER COMMANDS will not be accepted when operating in the Graphics Mode.

Carriage Return (0DH)	causes the cursor to return to position 1 of the current line and forces the overwrite mode.
Line Feed (0AH)	causes the cursor to move one line down in the same column position
Up Cursor (0BH or CTRL-K)	causes the cursor to move one line up in the same column position
Forward Cursor (0CH or CTRL-L)	causes the cursor to move non—destructively to the right one position. An auto line feed/ carriage return occurs at the end of the line.
Back Cursor (08H or CTRL-H)	causes the cursor to move non—destructively to the left one position. The cursor will not move past position 1.
Home Cursor (1EH or CTRL-)	Moves cursor to column 1, row 1
Erase Screen (1AH or CTRL-Z)	Fills screen with blanks and homes cursor.
Clear to end of Field (15H or CTRL-U)	Clears the characters from the cursor to the end of the line or until the existence of the first protected character, whichever comes first.

Tab (09H or CTRL–I)	Causes the cursor to move to the next tab stop that is not within a protected field or the first unprotected character following the next protected field or the home position, whichever comes first.
Delete Character (7FH or rubout)	Removes the character under the the cursor by shifting the remainder of the line one place to the left. The right–most character is filled with a blank.
Insert Mode (14H or CTRL–T)	Toggles the system in and out of the insert mode. In the insert mode, any character entered at the cursor causes the remaining characters of the line to be shifted to the right one place. The last character on the line is lost to the bit bucket. When not in the insert mode, the system is in the overwrite mode. In the overwrite mode any new character output just overwrites any previous character in that position.
Delete Line (04H or CTRL–D)	Deletes the current line indicated by the cursor by moving all lines below the current line up one line and filling in the last line with blanks. The cursor is returned to the beginning of the line.
Enter Line (05H or CTRL–E)	All lines from the current line to the end of the screen are moved down one line with the last line being lost to the bit bucket. The former current line is then blanked and the cursor returned to position 1.
Protect Fields/Unprotect Fields (10H or CTRL–P)	All inverted characters currently existing or subsequently entered are treated as protected fields when the protect mode is active. When inactive, there is no significance to the inverted video other that visual. When protected, these fields will not allow entry of data in most cases and affect such things as Tabs, and cursor control.

ESCAPE SEQUENCES————————————

To execute an ESCAPE SEQUENCE, the User program should successively place each Escape Sequence character in the Accumulator and CALL the CHARACTER OUT entry point at F803 H. All Escape Sequences begin with the ESCAPE character 1B H, followed by the appropriate character or characters listed below. Following the execution of the Escape Sequence, control will return to the User program, with none of the registers or status bits affected.

Set/Clear Tab (49H or "I")	Set a Tab stop at the current cursor position or clear an existing tab stop at current cursor position.
Clear All Tabs (09H or CTRL–I)	Unconditionally clear all all Tab positions
ASCII Text Mode (54H or "T")	Standard ASCII Text Mode allows for the display of the 96 character ASCII set (upper and lower case with punctuation). Character codes 20–7FH may be displayed (see Appendix 1). Character–by–character reverse video is available in this mode.
Extended Text Mode (45H or "E")	Extended Text mode currently allows for the display of 96 graphic characters but could be used to display foreign language fonts (when the upper half of the character generator ROMs are appropriately programmed). Character codes A0–FFH may be displayed (see Appendix 1). Character–by–character reverse video is available in this mode.
Graphic Mode (47H or "G")	Graphic Mode allows 255 ASCII and graphic characters to be displayed. Codes 00–1AH and Codes 1C–FFH may be displayed (see Appendix 1). Character–by–character reverse is NOT available in this mode. Note that new Control Character Commands will not be accepted, however, escape sequences will be accepted.

Scroll Mode Toggle (53H or "S")	Alternately places the screen mode scroll or wrap-around mode. The Text or Extended Text Modes default to the scroll option while the Graphic Mode defaults to the Wrap-around Option. Note that the scroll option may not be used in the Graphic Mode.
Upper/Lower Case Toggle . (55H or "U")	Alternately selects the Upper case only or Upper/Lower case display. Defaults to Upper Case only.
Inverse Video Toggle (56H or "V")	Alternately selects positive video (white characters on a black background) or inverse video (black characters on a white background). Default is positive video (white on black).
Lines Per Page Toggle (4CH or "L")	Alternately selects 12 line pages or 24 line pages. Default is 24 line pages.
Columns per Line Toggle (43H or "C")	Alternately selects 40 column lines or 80 column lines. Default is 80 column lines.
Addressable Cursor (=YX) ´	Places the cursor at the position defined by y-axis (line number) and X-axis (column number). Y-axis and X-axis position codes may be determined from Appendix 2.

b) VDP–80 Monitor

The monitor program allows the VIO to operate as an intelligent terminal when used with an ASCII encoded keyboard or other suitable ASCII encoded input device. It allows for keyboard control of all VIO firmware and hardware features and provides a number of extremely useful memory and I/O commands.

1) INPUT DEVICES

The VDP–80 monitor commands allow for the use of a keyboard, cassette recorder and a paper tape reader. It is ideally suited for use with the IMSAI MIO board.

1) To use the monitor, an ASCII encoded keyboard or other suitable ASCII encoded input device must be used. Configure the input interface (MIO) so that input data appears on Port 02H. The input interface (MIO) must also be configured to drive Bit 1 of Port 03H with an input data ready or input data available signal.

2) If the cassette commands are to be used with an IMSAI MIO cassette port, configure the MIO cassette port so that cassette data appears on Port 00H. Configure the cassette status so that Bit 2 of Port 03H is driven by the cassette data ready or cassette data available signal.

3) If the paper tape command is to be used with a Teletype device, configure the input interface (MIO) so that the teletype input data appears on Port 02H. Configure teletype status so that Bit 1 of Port 03H is driven by the input data ready or input data available signal. .

2) MONITOR OPERATION

1) The monitor ENTRY POINT is located at F806H. The display will come–up in the default mode:

   40x12 screen format
   Upper case only
   ASCII Text Mode
   Screen Scroll Mode

2) Any time a Hex parameter is requested from the user, any number of hex digits will be accepted, but only the last 4 (if a 16 bit value is requested) or the last 2 (if an 8 bit value is requested) will be used. The user may use any type of field delimiter .i.e., ",", "b", ";", etc.).

3) MONITOR COMMANDS

The following is a list of valid VDP–80 commands. In all cases the symbol @ represents CARRIAGE RETURN.

DISPLAY MEMORY

D@
D<start>@
D<start>,<end>@

> Display the contents of the memory locations beginning at the start address (or 0 if not specified) and continuing until the end address if specified. If no end address is specified only one single location will be displayed.

EXAMINE AND MODIFY MEMORY

E<addr>@ [XX]@

> Display the contents of memory location addr. The user then has four options:
>
> a. hitting the space bar will display the next successive location;
>
> b. hitting the minus key (–) will display the previous location;
>
> c. hitting carriage return will terminate the operation; or
>
> d. entering 2 or more valid Hex characters (followed by 1, 2, or 3 above) will modify memory as entered and perform 1, 2, or 3 above.

FILL MEMORY

F<start>,<end>, XX@

> Fill memory from starting address through ending address inclusive with specified XX byte.
>
> To Zero all of memory the following could be used (do not zero refresh memory on VIO): F0,EFFF,0@

MOVE MEMORY

M<start>,<end>,<dest>@

Move the block of memory starting at start address through end address to the destination address. The move occurs byte by byte from low memory to high memory. Hence, the following command will fill memory locations 100-1FF with the same byte that is at location 100:

M100,1FE,101@

SEARCH MEMORY

S<start>,<end>,<16 bit value>,<16 bit mask>@

Search memory from starting address through ending address for all instances of 16 bit values which, when ANDed with the specified mask yield the 16 bit specified value. If no mask is specified, FFFF is assumed which implies a full 16 bit compare.

EXAMPLE: You have a version of BASIC in memory locations 0-1FFF and wish to locate the status input instructions so you can locate and modify the output driver for use with the VIO.

S0,1FFF,DB03@ will print the address and contents of all memory locations which contain DB03.

It should be noted that specifying less than 4 digits in the value, or mask, might produce unexpected results since all inputted hex values are left padded with zeros.

VERIFY MEMORY

V<start>,<end>,<dest>@

Do a byte by byte compare from start through end to the destination. Print each pair of bytes which are not identical in the 2 strings.

PROTECT/UNPROTECT RAM 4A MEMORY

P<start>,<end>@
U<start>,<end>@

Protects all 1K blocks of memory (RAM 4A-4 only) which fall within the specified memory addresses. Any 1K blocks which overlap the boundary will also be protected.

JUMP TO MEMORY

J<start>@

Jump to memory at location start.

CALL MEMORY

C<start>@

Push the start address of the monitor on the stack and jump to memory address start. If the program entered does not alter the stack and terminates with a Return instruction, the monitor will be properly re—entered.

PERFORM DIRECT I/O

I<port #>@
O<port #>,XX@

Input or output from/to the specified port #. If input then display the resulting 8 bit value on the VIO. If output then output the specified value.

This instruction causes an IN (DB) or OUT (D3) instruction with the appropriate port # to be stuffed into RAM in the MPU–B memory and then called.

Loc	Inst	
10FA	DB or D3	IN or OUT
10FB	XX	Port #
10FC	C9	RET

The contents of memory locations 0, 1, and 2 will therefore be destroyed. The following command will change the RAM direct I/O ptr:

Y<addr>@

The address specified will now be used along with addr+1 and addr+2.

MEMORY DIAGNOSTIC

T@
T<start>,<end>@

Test memory from start address to end address. Every memory cell is cycled through all 256 bit patterns and tested to see if it contains the correct pattern. If all memory within the specified range is good, a prompt "?" will appear. If a bad cell or non-existent cell is located, the following display will occur:

AAAA VV SS

where: AAAA is the bad address
          VV is the value in the memory
          SS is what the value should be

Using the form T@ begins the memory test at location 0 and continues until the first bad cell, non-existent cell, or ROM memory is found.

The memory diagnostic DOES NOT destroy the contents of the memory tested.

GENERATE SYNC STREAM

G@

ADJUST AND ALIGN CASSETTE RECORDER

A@

EXEC FROM CASSETTE FILE

X@
X<start>,<end>,<exec>@

This command is identical to the Load command except that after the file is loaded it is CALLED at its execution address. If the program called does not change the stack ptr and returns properly, the monitor will be re-entered upon return.

LOAD INTEL HEX PAPER TAPE

H@

Reads characters from port 2 expecting to find a file in the Intel HEX format. If found, it is loaded at the specified address contained in the file itself. If a checksum error is detected in loading, a "C" will appear. If an

invalid character is detected, a "T" will appear and the operation will be terminated.

The input drivers are set up to read an MIO or SIO jumpered so the status appears on port 3 with bit 1 going high when data is available. The data should appear on port 2. This is the configuration which would be required if a teletype were being used for input.

LOAD CASSETTE FILE

L@
L<START>,<END>,<EXEC>@

This routine allows the user to load an object file (Type 81H) from cassette. Cassette files generated by IMSAI or by TCOS (Tape Cassette Operating System) may be loaded with this command.

All Type 81H files created by IMSAI or TCOS have a header record in front containing the file name (up to 5 characters), the starting memory address, the ending memory address, and an execution address which is often the same as the starting address. By typing in the L@ command, the user is effectively saying, "Load the next headered file into the memory address specified in the header."

By typing in the "L start, end, exec@" command, the user is saying, "load the next type 81 file whether headered or not using the start, end addresses specified in the command itself.

After the command is entered, the cassette recorder should be started. If a header is encountered, the name of the file to be loaded will be displayed. If no starting and ending addresses were specified, they will be read from the header. If an error is discovered in the header, an "I" initialization error will appear and the operation terminated.

All files created by IMSAI or TCOS block their records into 128 (80H) byte records. As the loader loads the object file, an indicator, "*", is displayed after each record is loaded to indicate a good load or a checksum error, "C", for that particular record.

After the entire file has been loaded, the starting address, ending address, and execution address of the file just loaded will be displayed. If at any point, an incorrect file type is encountered, a "T" for type code error will be displayed and the operation terminated.

CAUTION: DUE TO TIMING CONSIDERATIONS, THE VIO MUST NOT BE ALLOWED TO SCROLL DURING A LOADING FROM CASSETTE. THEREFORE THE SCREEN SHOULD BE ERASED (CTRL–Z) BEFORE KEYING IN THE LOAD COMMAND.

BOOT FROM FLOPPY DISK

   B@

READ FROM DISKETTE

   R<tt,ss,bbbb,u>@

      where:

           tt = track
           ss = sector
           bbbb = buffer address
           u = unit

      Track and sector must be specified; buffer and unit specifications are optional.

      If the information contained within the bracket is not specified, the following values are assumed:

           track = 00
           sector = 01
           buffer address = 0000
           unit #(master) = 0

WRITE TO DISKETTE

   W@
   W<tt,ss,bbbb,u>

JUMP AND SWITCH-OUT MPU-B

   K<x>@

      Jump to locattion x and switch off MPU-B ROM and system memory

      MPU-B ROM residing at D800 - DBFF
      MPU-B RAM residing at D001 - D0FF

SWITCH OUT MPU-B AND JUMP TO VIO MONITOR
(INITIALIZE SCREEN)

Q@

SET BAUD RATE

Z@ (requires two terminals)
Zxxxx (where xxxx is baud rate)

Sets baud rate for MPU-B serial port as 12 and 13 and 4 and S (the system
terminal is tied to 2 and 3).

When Z@ is typed, the system will print:

HIT SPACE BAR

The user then hits the space bar on the terminal tied to 12 and 13. the system
will print out:

xxxx BAUD SERIAL PORT

3. VIO OPERATION WITHOUT VIOROM

When the VIOROM is not used, the VIO functions and modes of operation may be controlled directly from user programs.

It should be noted that operation without the VIOROM is intended primarily for those Users who wish to write a display driver for a particular application, or for those Users who wish to use only very minimal display functions. For most video display applications, it is recommended that the VIOROM be used.

a) VIO Initialization

On power–up or reset, the VIO will default to the following mode of operation:

> screen blanked (mode 0)
> 80x24 screen format

It is then up to the user's program to 1) initialize the VIO's refresh memory for the initial display desired (Part b); and 2) to select the desired VIO mode of operation using the appropriate command byte (Part c).

b) Character Display

To display a character on the screen, the user's program must store the character code for the character to be displayed into the VIO's refresh memory.

Since the VIO's refresh memory consists of 1K or 2K of RAM residing in the system memory space, the user's program may store a character code into the refresh memory using any memory store instruction (e.g., MOV M,R). It is not possible to write into the refresh memory using a front panel Deposit or DMA.

Note that the address of the VIO's refresh memory was set by the user in Section III–B,1, Steps 1 and 2. The position of the displayed character, on the screen, is a function of the location (in the refresh memory) where the character code is stored.

The first memory location in the VIO's refresh memory corresponds to the first display position in the upper left corner of the screen. Each succeeding memory location will correspond to the next display position. Characters are displayed by row, from left to right, with the last display position in the lower right corner of the screen. The relationship between memory locations is shown in Figure III–20.

For example, if the refresh memory were jumpered to reside at F000H (standard address), storing a 31H (ASCII "1") in memory location F000H would cause the character "1" to appear in the upper left hand corner of the screen.

Similarly, a 31H stored in location F028H would cause the character "1" to appear as the 41st character of the first line (if the 80 column option is selected) or as the first character of the second line (if the 40 column option is selected).

A list of the valid character codes appears in Appendix 1. The characters which may be

displayed on the screen depend entirely upon the mode of operation selected by the user: Mode 0, Mode 1, Mode 2, or Mode 3. These modes are selected with the appropriate command byte described in Part c.

MODE 0

When MODE 0 is selected, the screen will be blanked and no characters will be displayed. Display of the characters stored in the VIO's refresh memory will not begin until MODE 0 is terminated.

MODE 1

When MODE 1 is selected, any of the 128 graphic characters represented by character codes 80–FF H (Appendix 1) may be displayed. In MODE 1 bit 7 of the character code acts as the reverse video flag (0=positive video, 1=reverse video).

Note that the character codes listed in Appendix 1 are 8 bits wide. In MODE 1, only the lower 7 bits will be used as the actual character code. The high order bit is used as the reverse video flag.

MODE 2

When MODE 2 is selected, any of the 128 characters represented by the character codes 00–7F H (Appendix 1) may be displayed. This includes the standard 96 character ASCII set and 32 graphic characters. In MODE 2, bit 7 of the character code will act as the reverse video flag (0 = positive video, 1 = reverse video).

For example, a 31H (ASCII "1") stored into the refresh memory would cause the character "1" to be displayed in positive video (white character on dark background). The same code, with bit 7 on, would cause the character to be displayed in reverse video. Thus a B1H would cause the character "1" to be displayed as a dark character on a light background.

Note that the character codes listed in Appendix 1 are 8 bits wide. In MODE 2, only the lower 7 bits will be used as the actual character code. The high order bit is used as the reverse video flag.

MODE 3

When MODE 3 is selected, any of the 256 characters represented by character codes 00–FFH (Appendix 1) may be displayed. Note that in MODE 3, character–by–character video is not available. All 8 bits of the character codes listed in Appendix 1 will be used as the actual character codes.

```
┌─────────────────────────────────────────────────────────────┐
│  B        B+1      B+2 ......................... B+C−1        │
│  B+C      B+C+1    B+C+2 ....................... B+2C−1       │
│  B+2C     B+2C+1   B+2C+2 ...................... B+3C−1       │
│  :                                               :           │
│  :                                               :           │
│  :                                               :           │
│  :                                               :           │
│  :                                               :           │
│  :                                               :           │
│  :                                               :           │
│  :                                               :           │
│  :                                               :           │
│  B+(L−1)C ...................................... B+LC−1       │
└─────────────────────────────────────────────────────────────┘
```
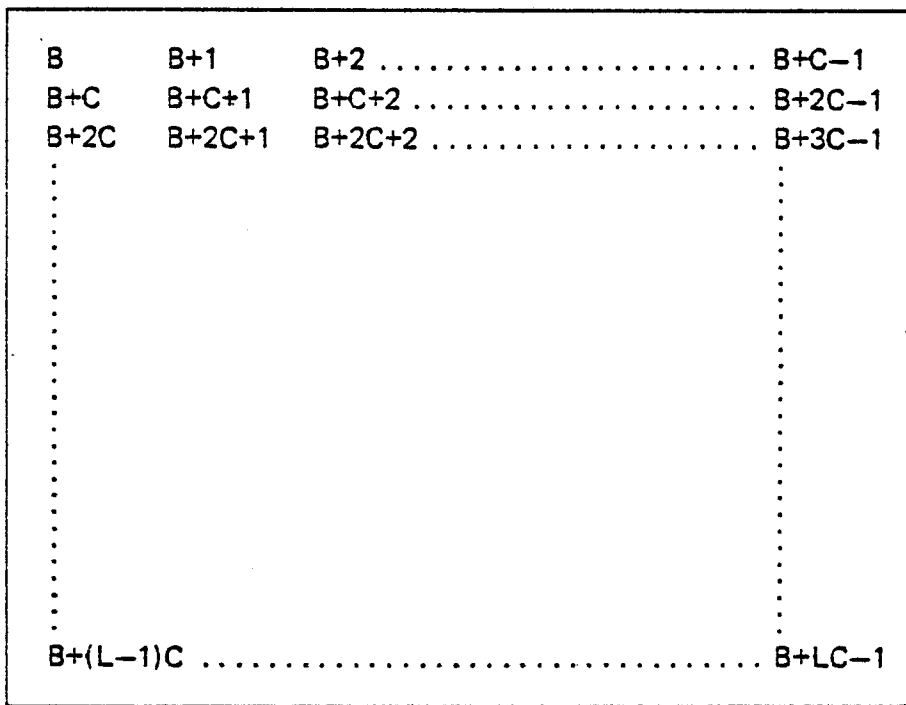
B=BASE ADDRESS OF REFRESH MEMORY
C=NUMBER OF COLUMNS PER LINE
L=NUMBER OF LINES PER PAGE

FIGURE III-20   RELATIONSHIP BETWEEN DISPLAY POSITION
                 AND REFRESH MEMORY LOCATION

c) Command Byte

The top location in the VIO's refresh memory serves as a memory mapped command port. Note that the address of this port is dependent upon the location of the refresh memory (set in Section III—B,1, Steps 1 and 2).

If the refresh memory was set to reside in the top half of the VIO's 4K address space, (Section III—B,1, Step 2), the command port address will be of the form XFFF, where X is equal to the high order address determined in Section III—B,1, Step 1. If the refresh memory was set to reside in the lower half of the VIO's 4K address space, the command port address will be of the form X7FF, where X is equal to the high order address determined in Section III—B,1, Step 1.

The command port address is not affected by the amount of refresh memory (1K or 2K) installed.

The command byte which is stored in the command port location will determine line length, page length, full screen reverse video, and mode of operation, according to the following format:

Bit 0   when equal to 0, produces 80 character lines
            when equal to 1, produces 40 character lines

Bit 1   when equal to 0, produces 24 line pages;
            when equal to 1, produces 12 line pages.

Bits 2—3   are encoded and control the VIO mode of operation
            When equal to 00, activates Mode 0,
            video off, screen blanked.

            When equal to 01, activates Mode 1. Character
            codes 80—FF (Appendix 1) may be displayed. Bits
            0—6 of codes 80—FF are used as the character
            code. Bit 7 is used as the character—by—character
            reverse video flag (0 = positive video, 1 = reverse video).

            When equal to 10, activates Mode 2. Character codes
            00—7F H (Appendix 1) may be displayed.
            Bits 0—6 of codes 00—7F H are used as the character
            code. Bit 7 is used as the character—by—character
            reverse video flag (0=positive video, 1=reverse
            video).

            When equal to 11, activates Mode 3. Character codes
            00—FF (Appendix 1) may be displayed. Character—
            by—character reverse video is not available in
            this mode, and all 8 bits of codes 00—FF are
            used as character code.

COMMAND BYTE

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0

0=80 CHAR/LINE
1=40 CHAR/LINE

NOT USED

0=24 LINES/PAGE
1=12 LINES/PAGE

00=MODE 0
01=MODE 1
10=MODE 2
11=MODE 3

0=POSITIVE VIDEO
1=REVERSE VIDEO
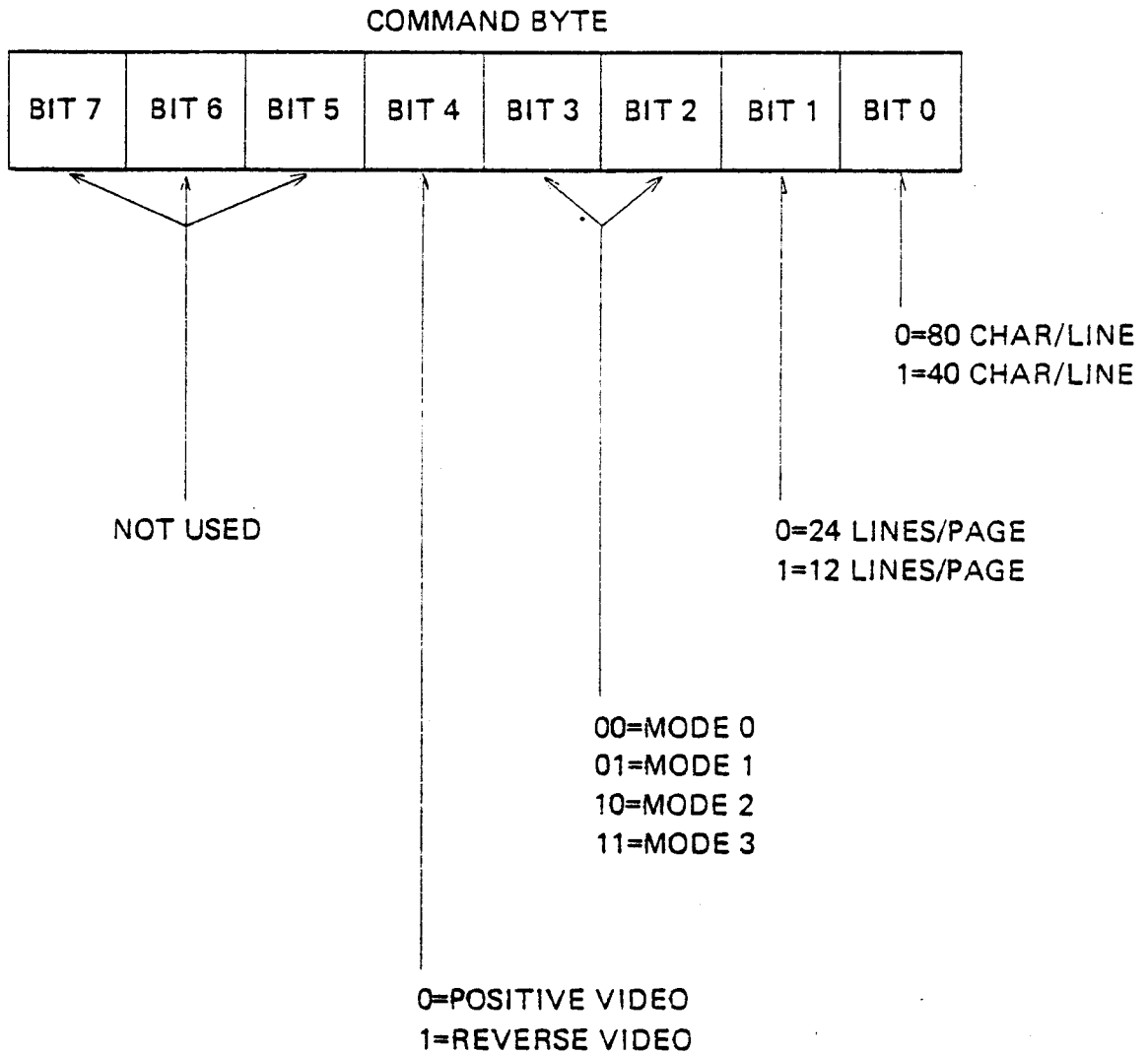
FIGURE III-21   COMMAND BYTE FORMAT

Bit 4   when equal to 0, causes the screen display to appear as
          positive video (light characters on a dark
          background).

          When equal to 1, causes the screen display
          to appear as reverse video (dark characters
          on a light background).

Bits 5–7 are not used.

The command byte format is shown in Figure III–21. On power–up or reset, the command byte
will default to 00H (Mode 0) with the screen blanked. The command byte may then be changed
using any memory store instruction (e.g., MOV M,R). If the VIO contains a full 2K bytes of
refresh memory, the command byte may also be read using any memory read instruction (e.g.,
MOV R,M). It is not possible to write into the Command Port location using a front panel
Deposit or DMA.

NOTE: Reading the command port location after a reset, but prior to writing, will not reflect
the default command (00).

For example, if the VIO's refresh memory is set to reside in location F000–F7FF (bottom half
of 4K block beginning at F000), we may store the command byte 07H in location F7FF to
produce

          positive video display
          Mode 1
          12 lines per page
          40 characters per line

Similarly, if the VIO's refresh memory is set to reside in location F800–FFFF (top half of 4K
block beginning at F000H), we may store the command byte 08H in location FFFF to produce

          positive video display
          Mode 2
          24 lines per page
          80 characters per line

C. APPENDICES

APPENDIX 1 ——————————————————

The characters represented by the character codes 00–FF H, comprise the complete 256 character set of the VIO.

Characters which may be displayed depend upon the display mode selected. Codes 20–7F H may be displayed through the firmware, in the ASCII Text Mode. Codes A0–FF H may be displayed through the firmware in the Extended Text Mode. Codes 00–FF H may be displayed through the firmware in the Graphic Mode (with the exception of the character 16 H).

Codes 00–FF H may also be displayed by directly accessing the VIO's refresh memory.

The valid character codes 00–FF are listed in the following pages.

CODE: 00 H
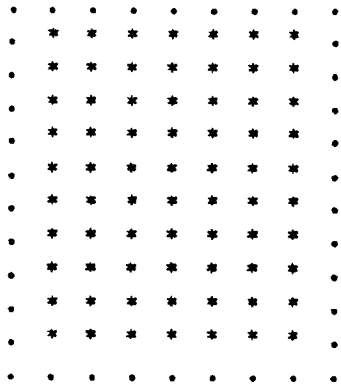
CODE 03 H
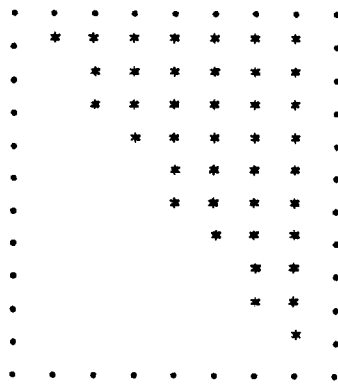
CODE 06 H
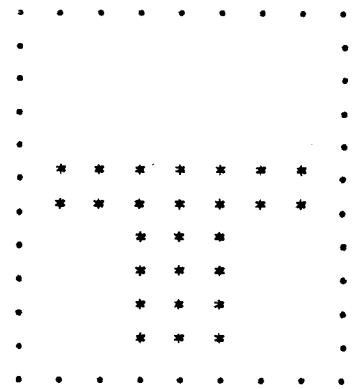
CODE 01 H
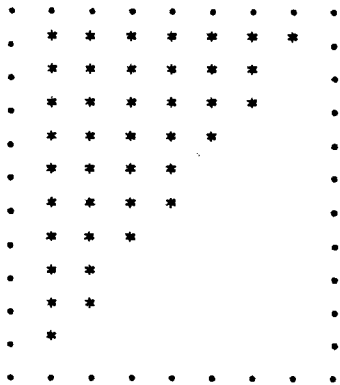
CODE 04 H
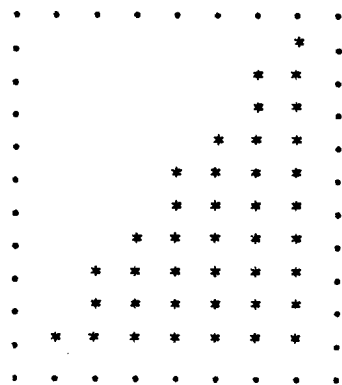
CODE 07 H
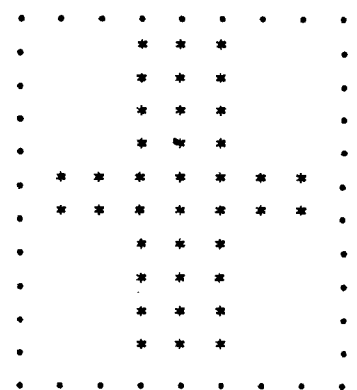
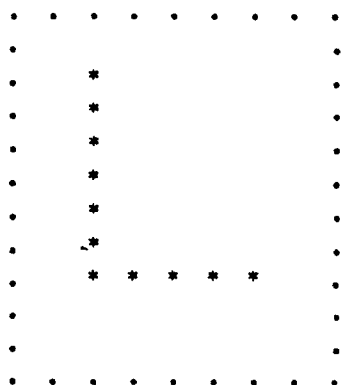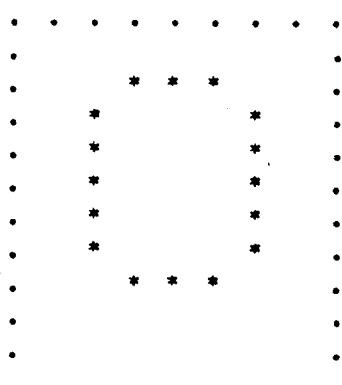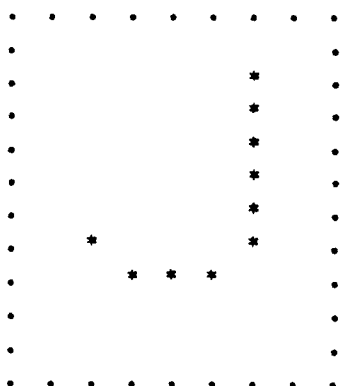CODE 02 H
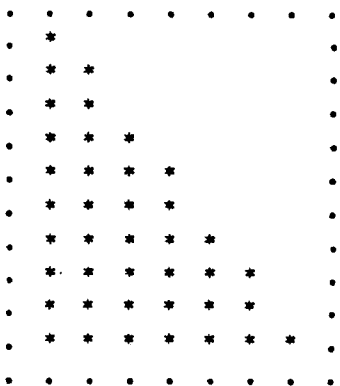
CODE 05 H
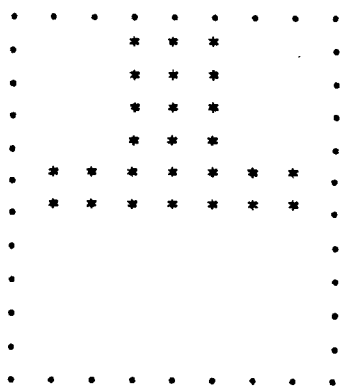
CODE 08 H
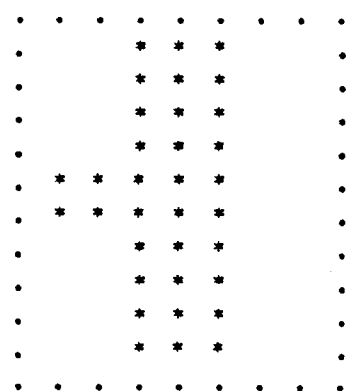
CODE 09 H
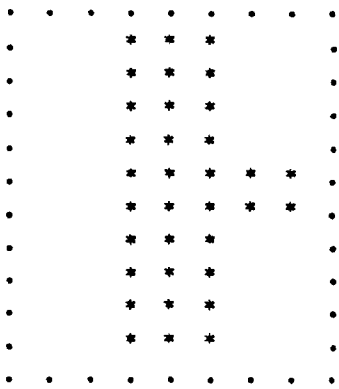
CODE 0C H
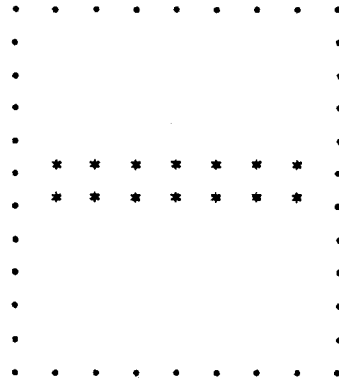
CODE 0F H
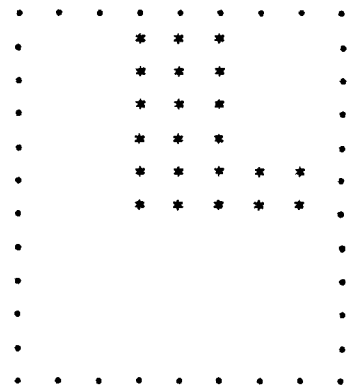
CODE 0A H
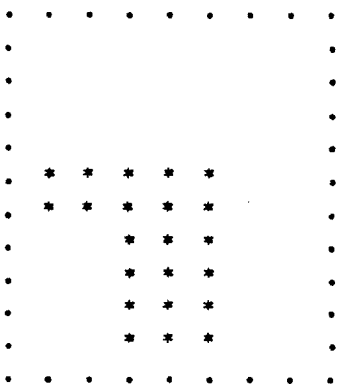
CODE 0D H
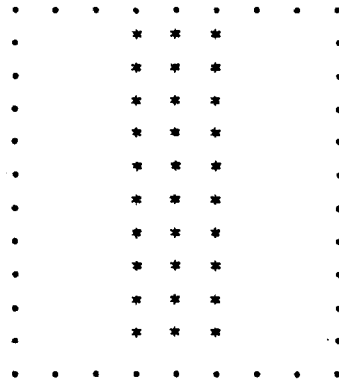
CODE 10 H
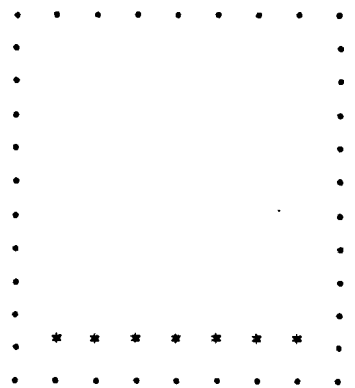
CODE 0B H
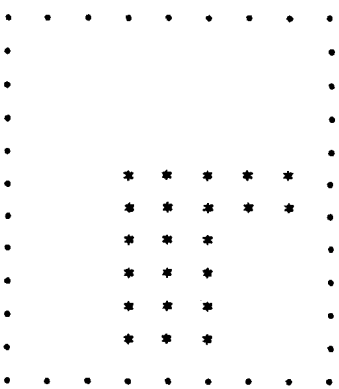
CODE 0E H
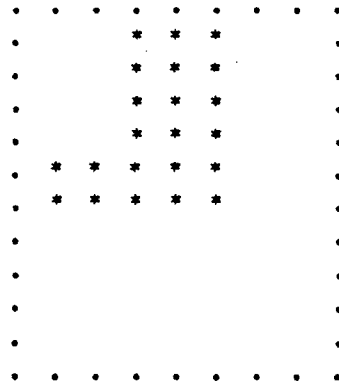
CODE 11 H
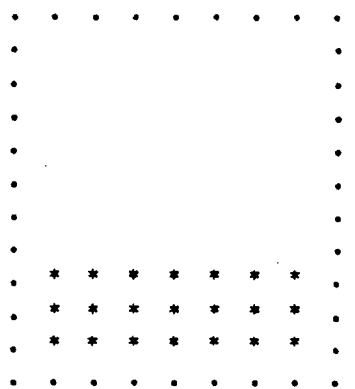
CODE 12 H
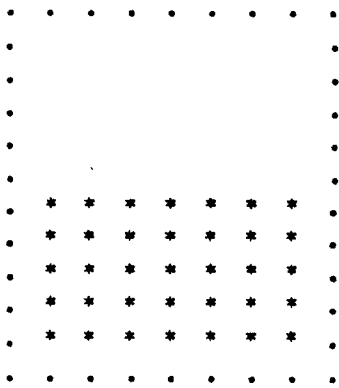
CODE 15 H
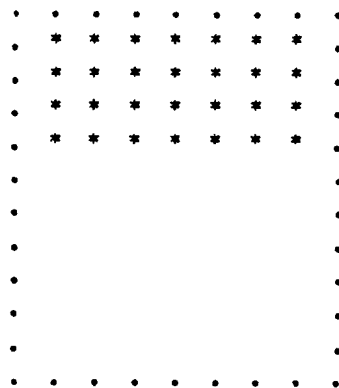
CODE 18 H
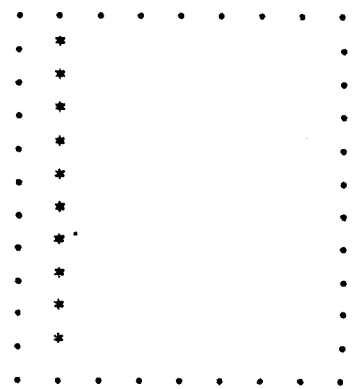
CODE 13 H
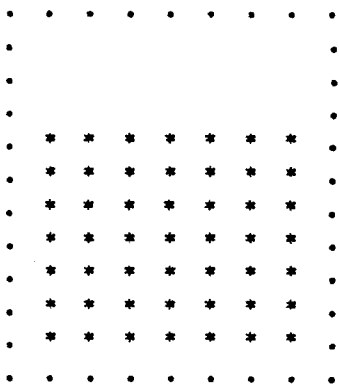
CODE 16 H
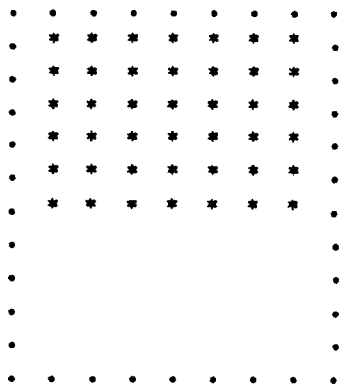
CODE 19 H
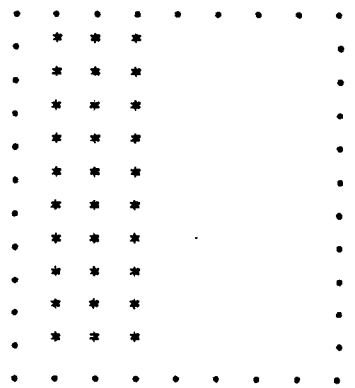
CODE 14 H
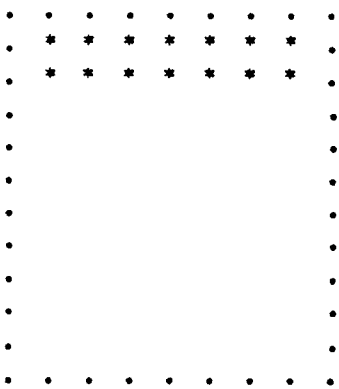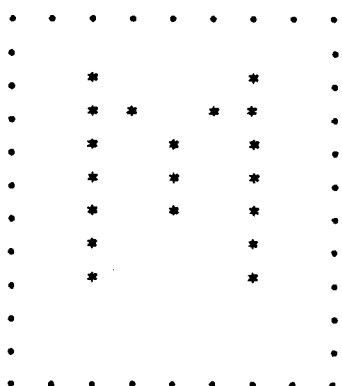
CODE 17 H
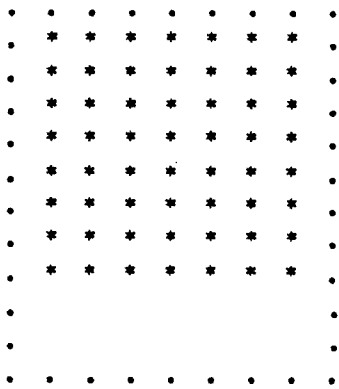
CODE 1A H
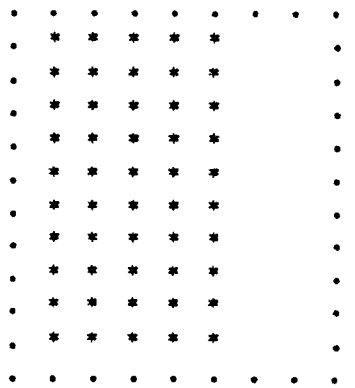
CODE 1B H
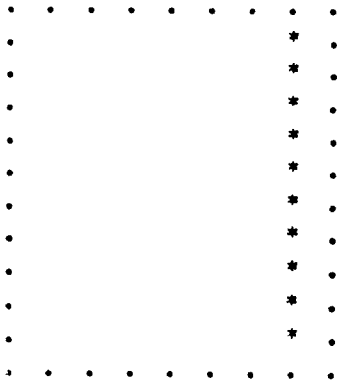
CODE 1E H
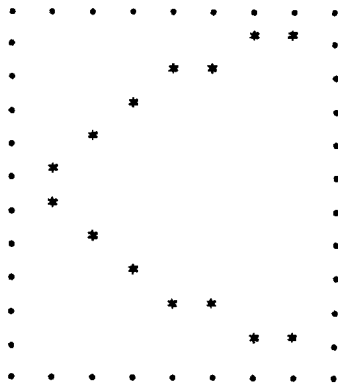
CODE 21 H
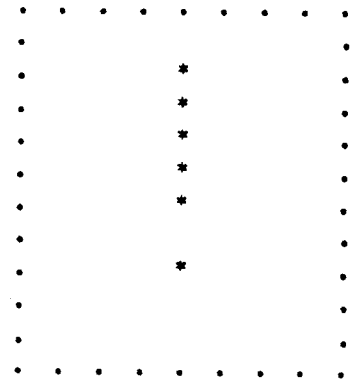
CODE 1C H
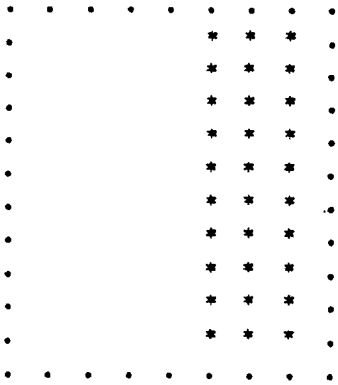
CODE 1F H
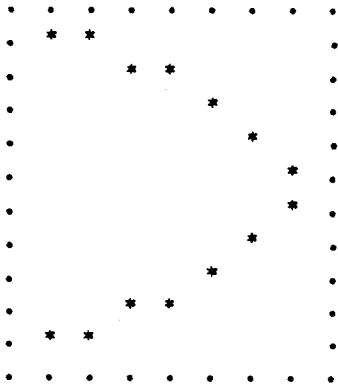
CODE 22 H
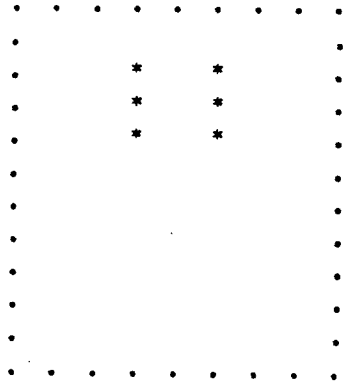
CODE 1D H

CODE 20 H

CODE 23 H

CODE 24 H

CODE 27 H

CODE 2A H

CODE 25 H

CODE 28 H

CODE 2B H

CODE 26 H

CODE 29 H
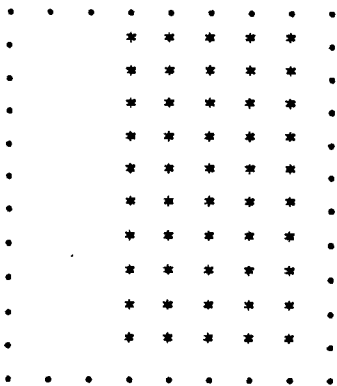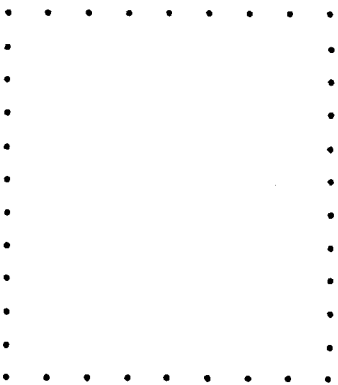
CODE 2C H
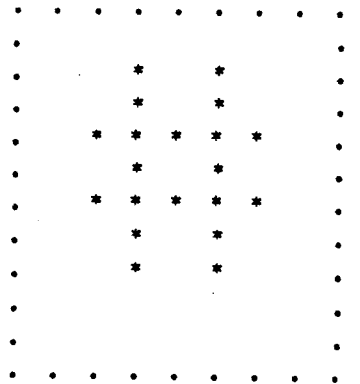
CODE 2D H
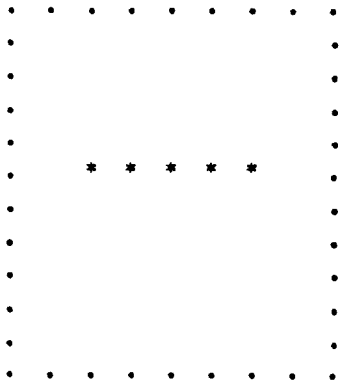
CODE 30 H
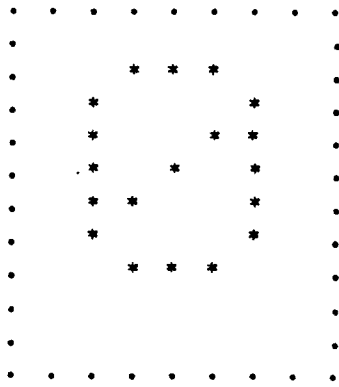
CODE 33 H
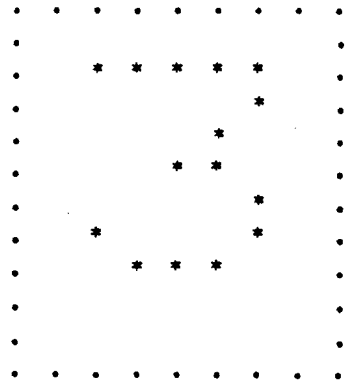
CODE 2E H
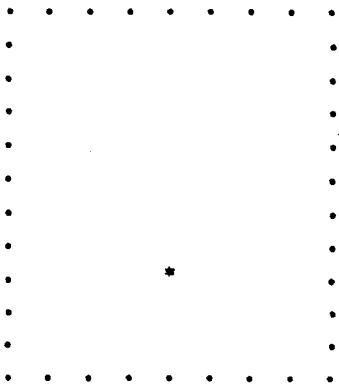
CODE 31 H
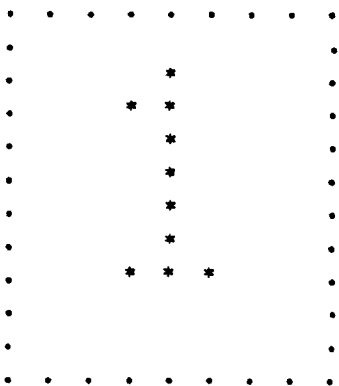
CODE 34 H
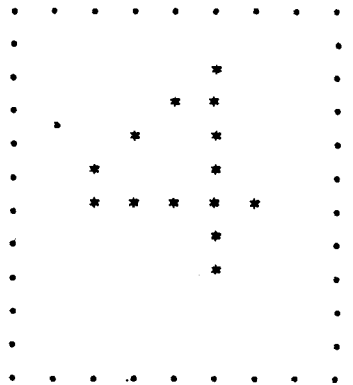
CODE 2F H
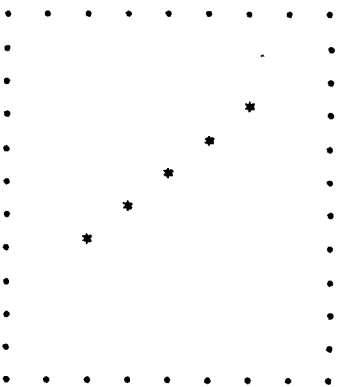
CODE 32 H
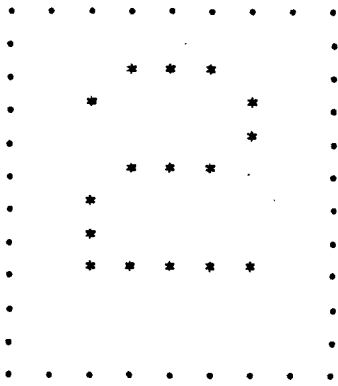
CODE 35 H
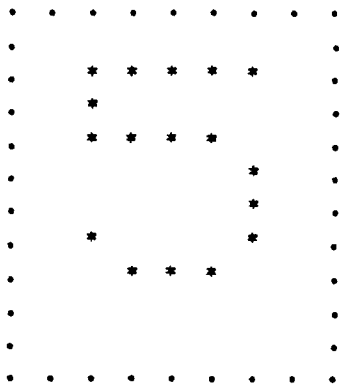
CODE 36 H
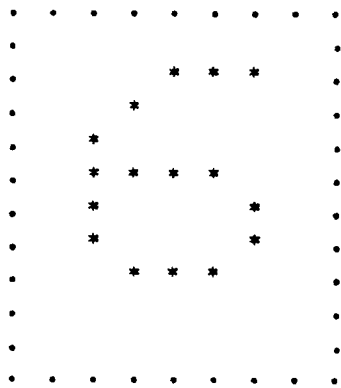
CODE 39 H
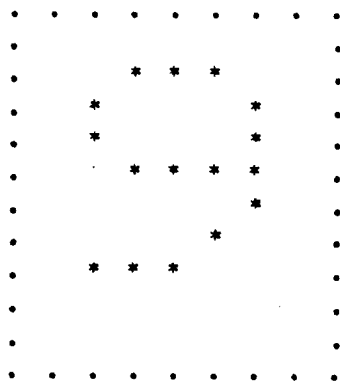
CODE 3C H
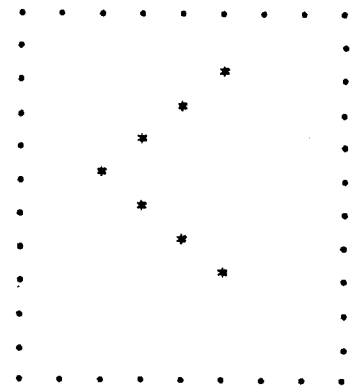
CODE 37 H
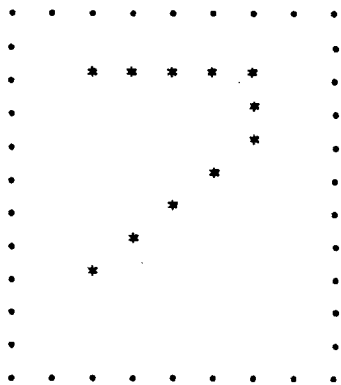
CODE 3A H
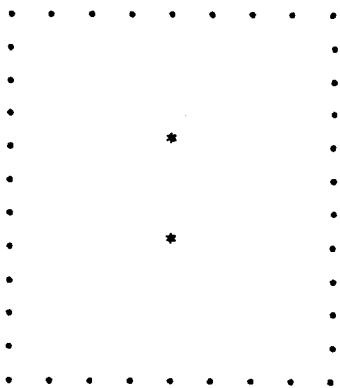
CODE 3D H
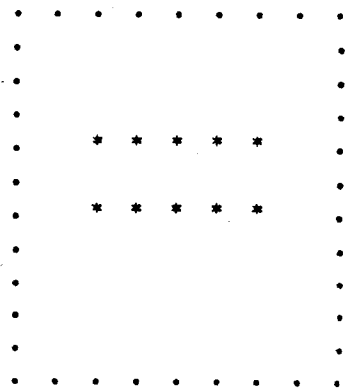
CODE 38 H

CODE 3B H

CODE 3E H

CODE 3F H

CODE 42 H

CODE 45 H

CODE 40 H

CODE 43 H

CODE 46 H

CODE 41 H

CODE 44 H
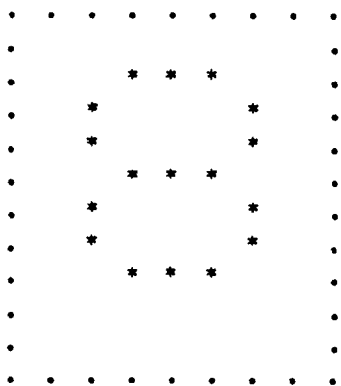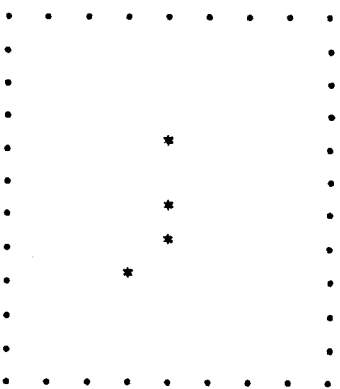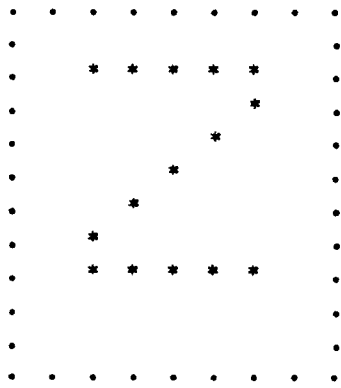
CODE 47 H
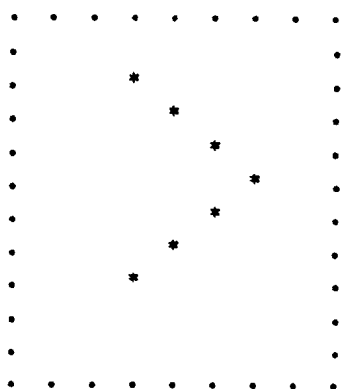
CODE 48 H
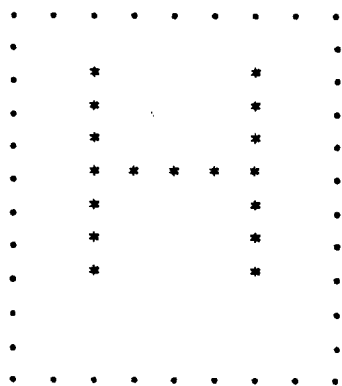
CODE 4B H
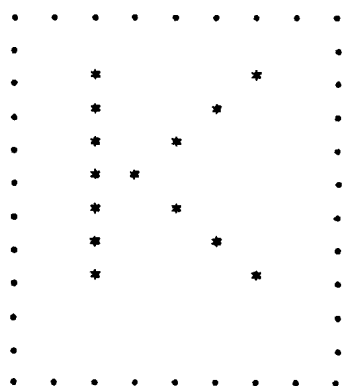
CODE 4E H
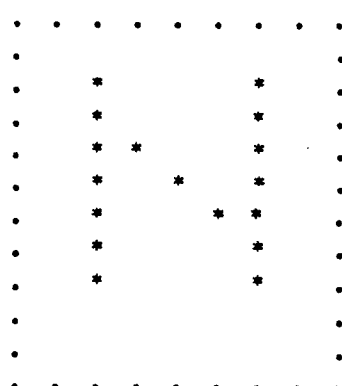
CODE 49 H
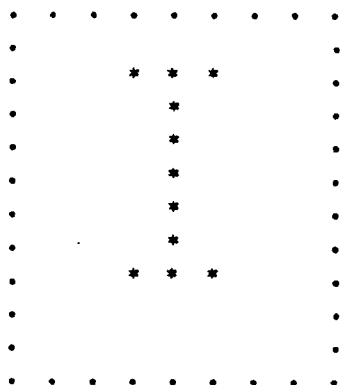
CODE 4C H

CODE 4F H

CODE 4A H

CODE 4D H

CODE 50 H
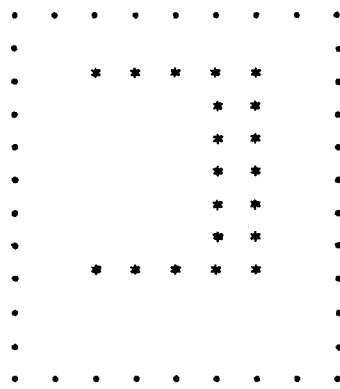
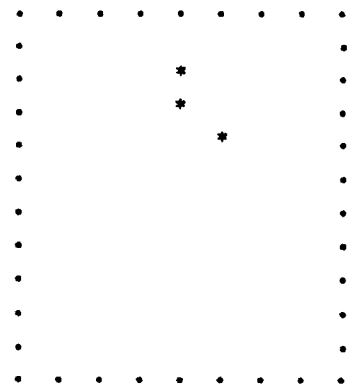CODE 51 H

CODE 54 H
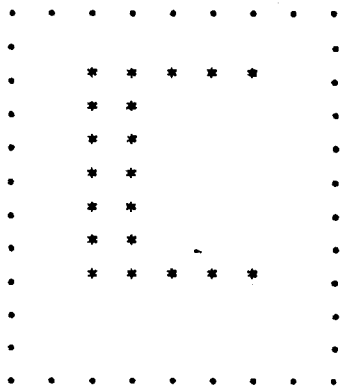
CODE 57 H

CODE 52 H
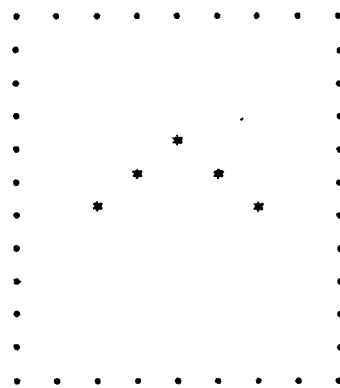
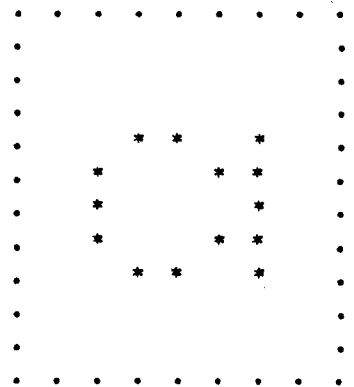CODE 55 H

CODE 58 H

CODE 53 H

CODE 56 H

CODE 59 H
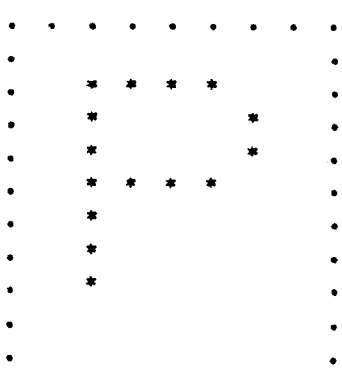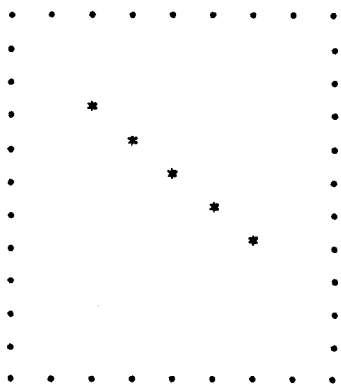
CODE 5A H

CODE 5D H

CODE 60 H

CODE 5B H

CODE 5E H

CODE 61 H

CODE 5C H

CODE 5F H

CODE 62 H

CODE 63 H
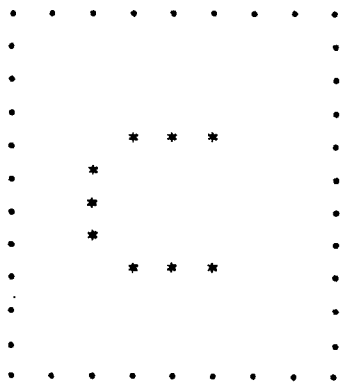
CODE 66 H
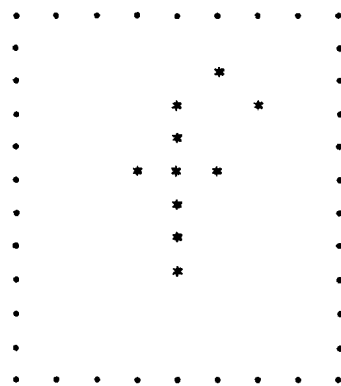
CODE 69 H
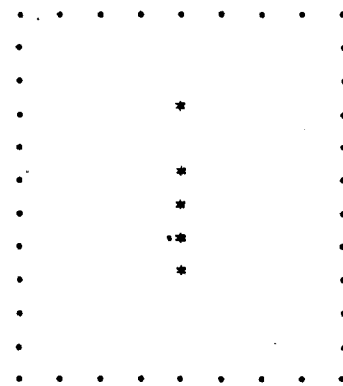
CODE 64 H
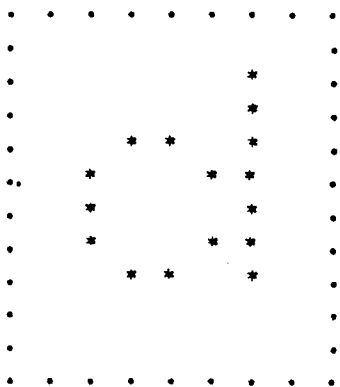
CODE 67 H
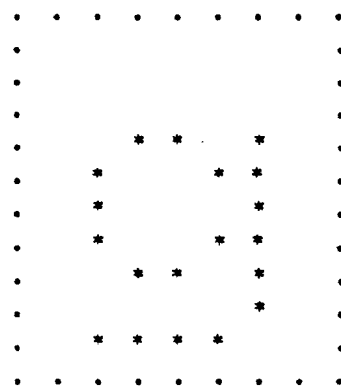
CODE 6A H
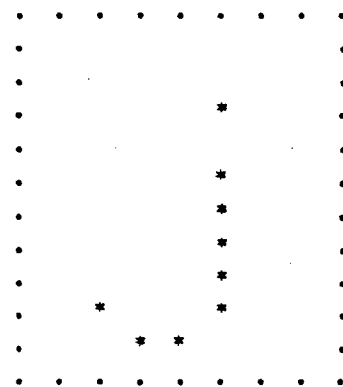
CODE 65 H
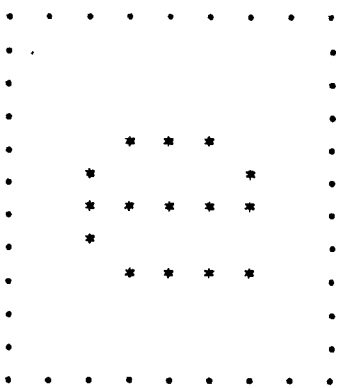
CODE 68 H
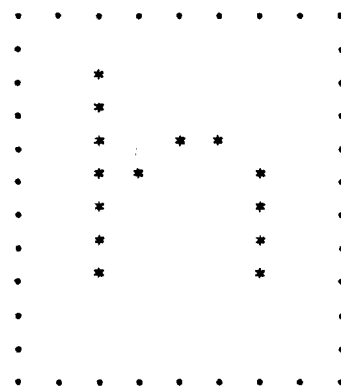
CODE 6B H
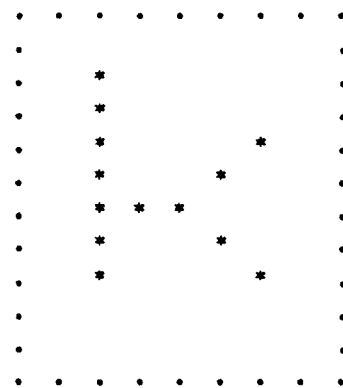
CODE 6C H
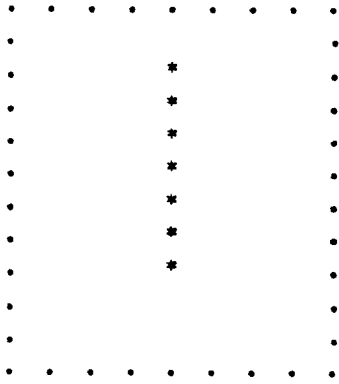
CODE 6F H
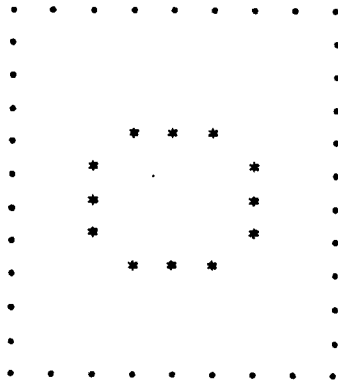
CODE 72 H
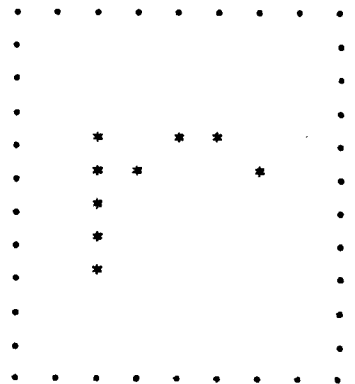
CODE 6D H
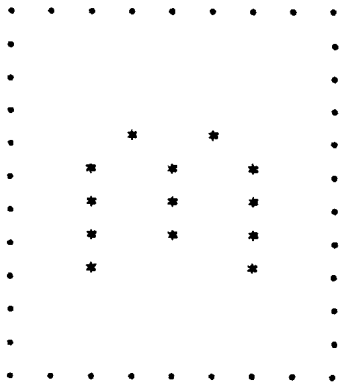
CODE 70 H
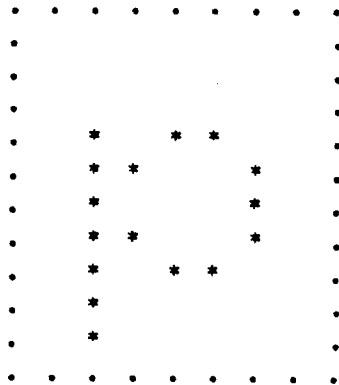
CODE 73 H
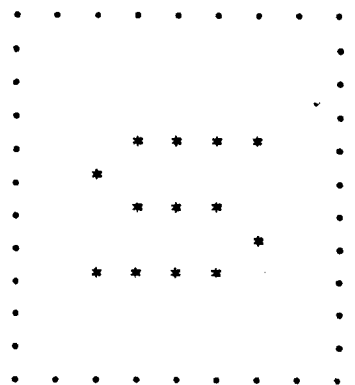
CODE 6E H
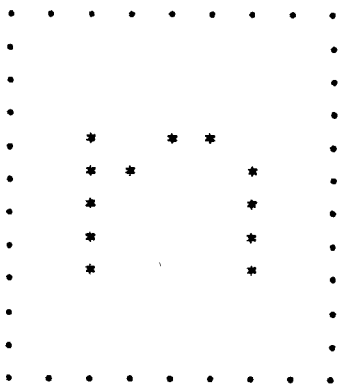
CODE 71 H
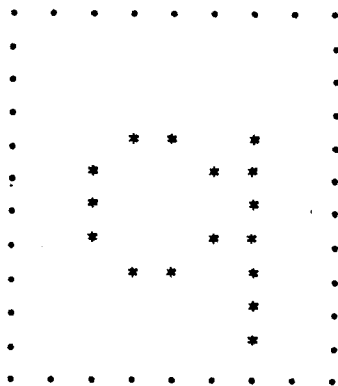
CODE 74 H
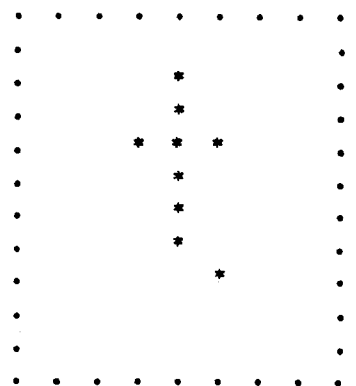
CODE 75 H
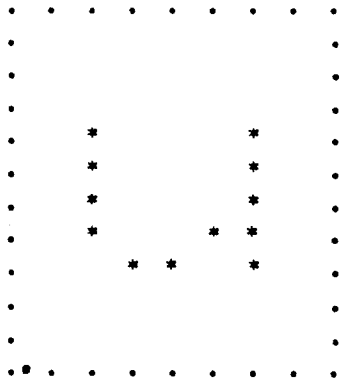
CODE 78 H
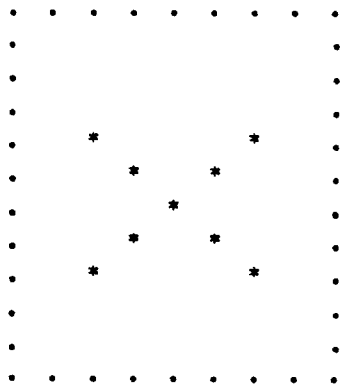
CODE 7B H
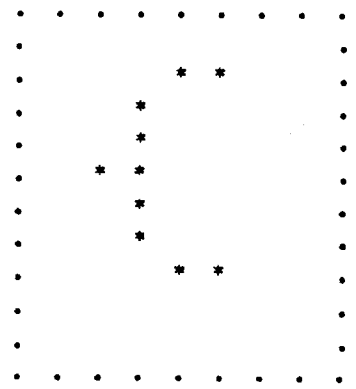
CODE 76 H
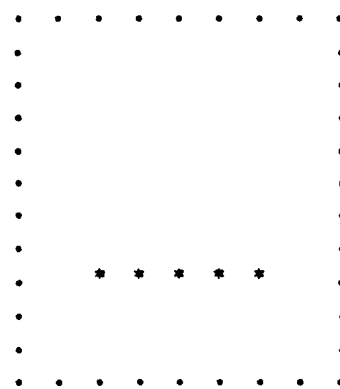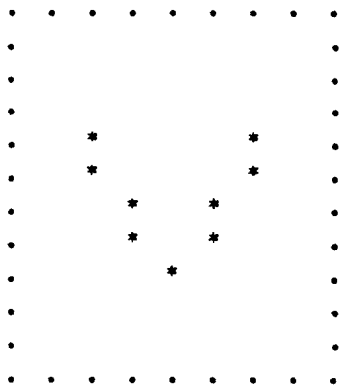
CODE 79 H
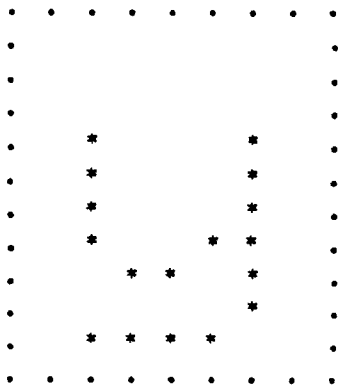
CODE 7C H
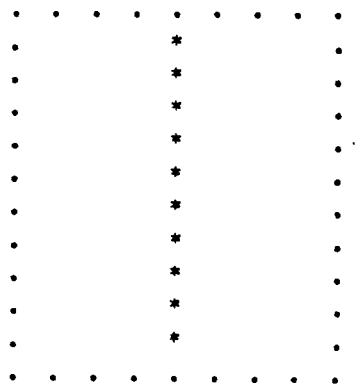
CODE 77 H
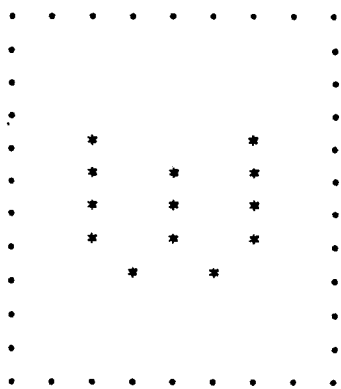
CODE 7A H
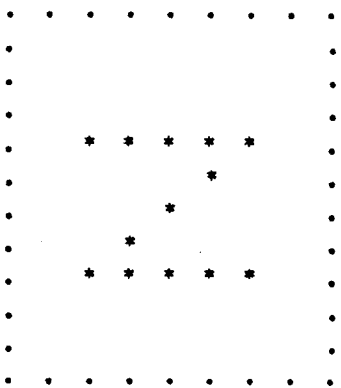
CODE 7D H
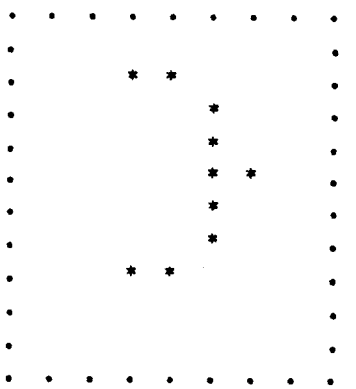
CODE 7E H
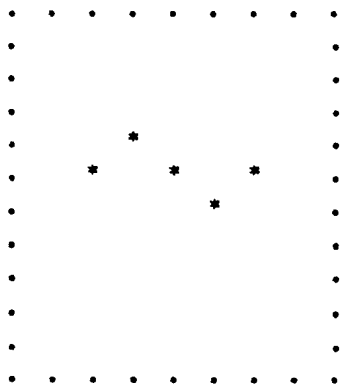
CODE 81 H
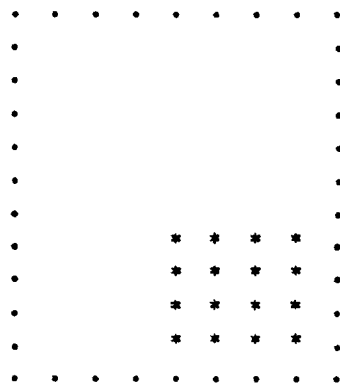
CODE 84 H
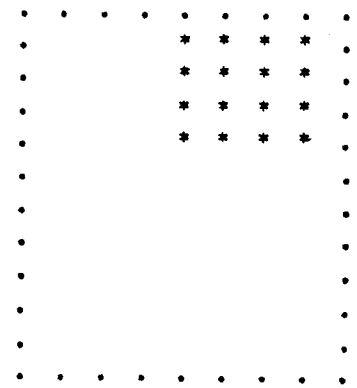
CODE 7F H
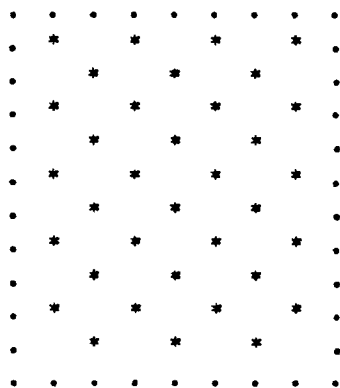
CODE 82 H
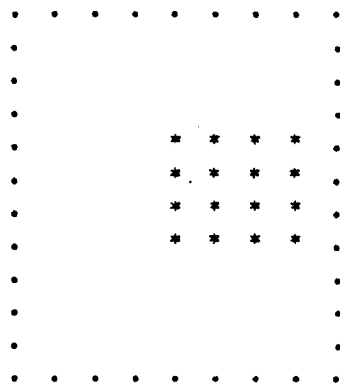
CODE 85 H
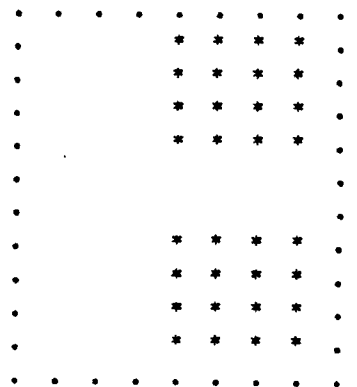
CODE 80 H
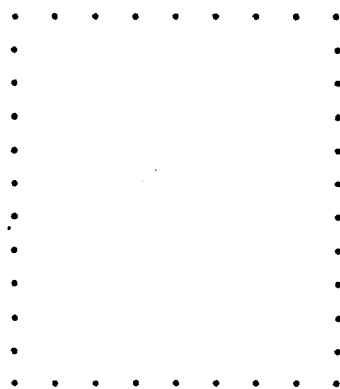
CODE 83 H
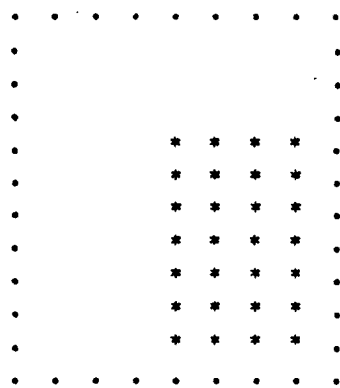
CODE 86 H
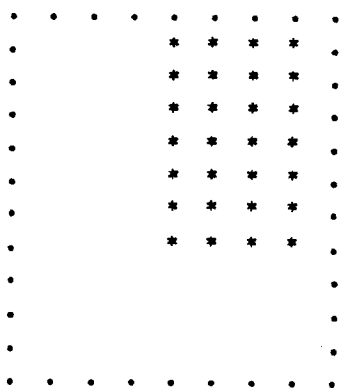
CODE 87 H
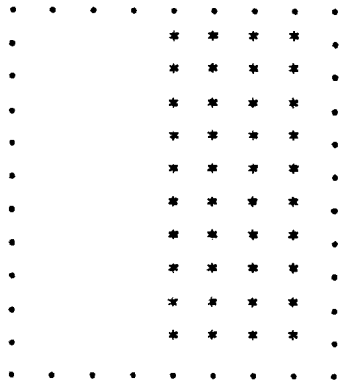
CODE 8A H
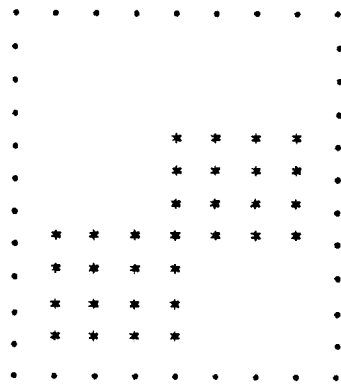
CODE 8D H
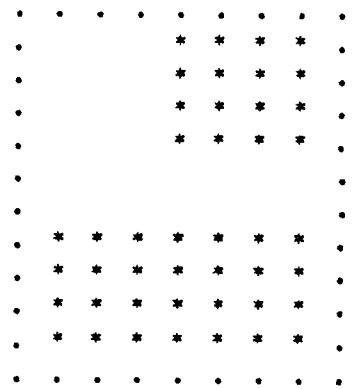
CODE 88 H
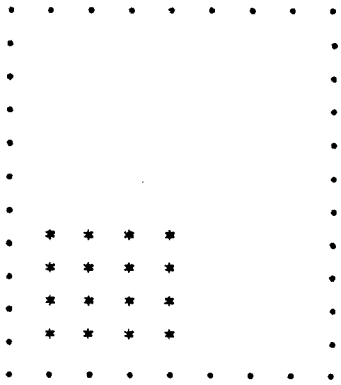
CODE 8B H
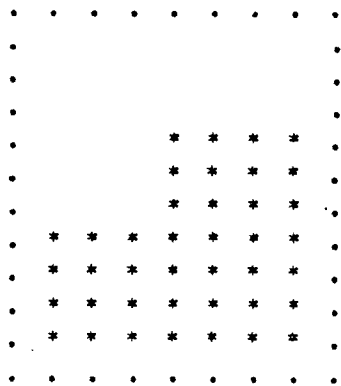
CODE 8E H
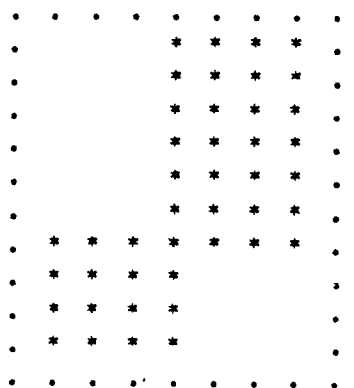
CODE 89 H
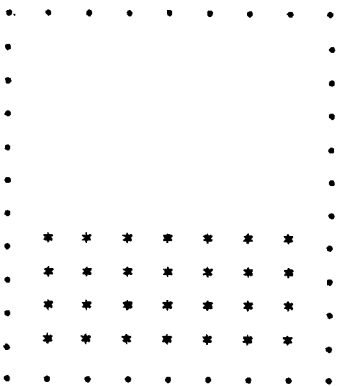
CODE 8C H
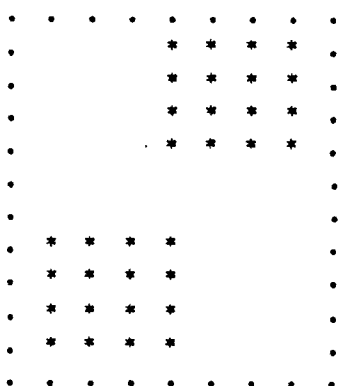
CODE 8F H
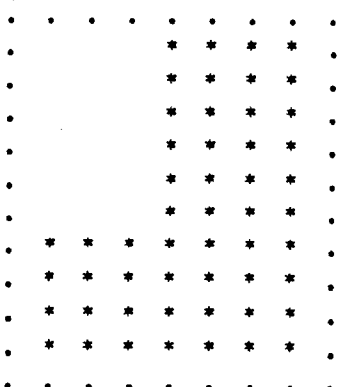
CODE 90 H
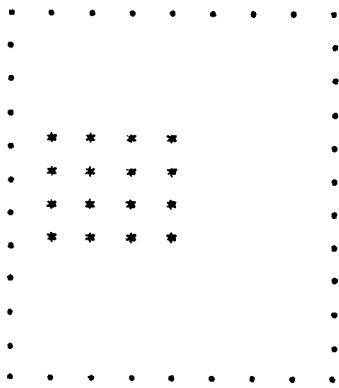
CODE 93 H
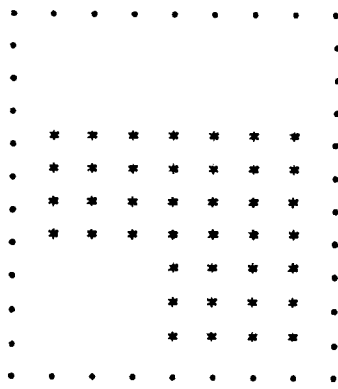
CODE 96 H
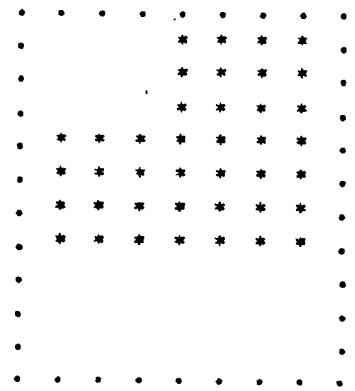
CODE 91 H
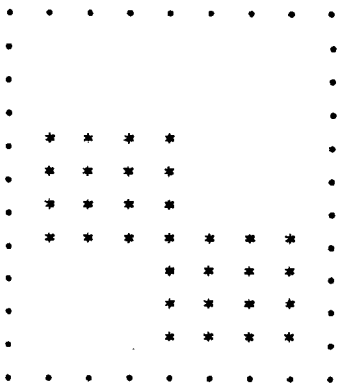
CODE 94 H
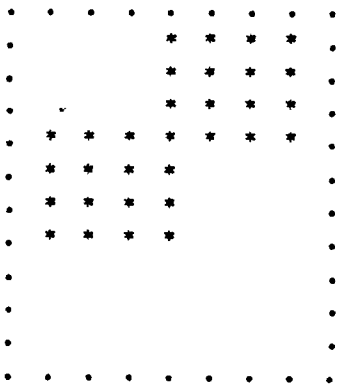
CODE 97 H
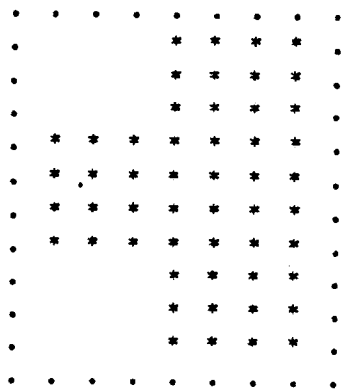
CODE 92 H

CODE 95 H

CODE 98 H

CODE 99 H

CODE 9C H

CODE 9F H

CODE 9A H

CODE 9D H

CODE A0 H

CODE 9B H

CODE 9E H
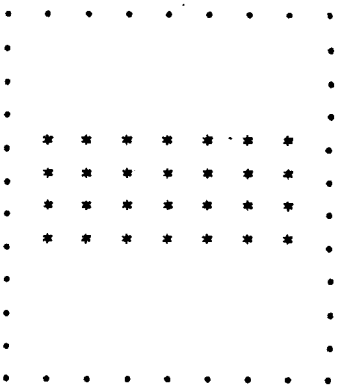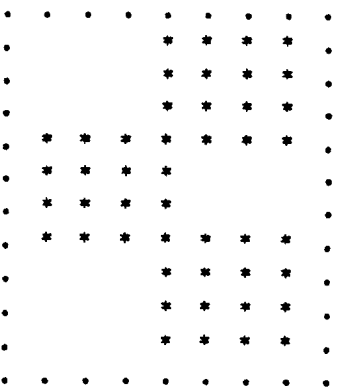
CODE A1 H
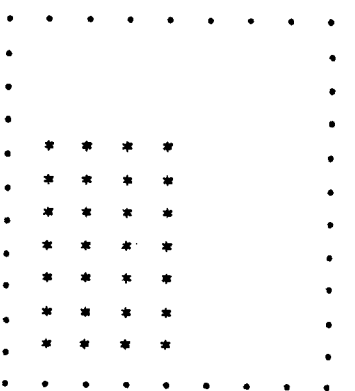
CODE A2 H
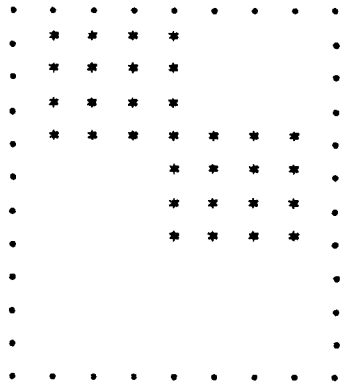
CODE A5 H
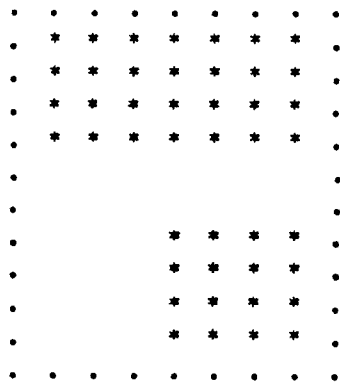
CODE A8 H
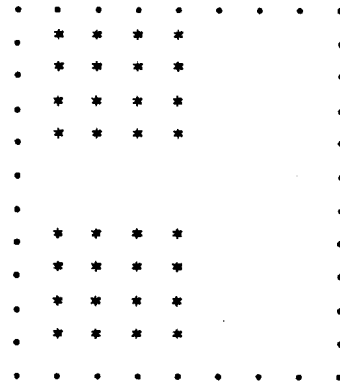
CODE A3 H
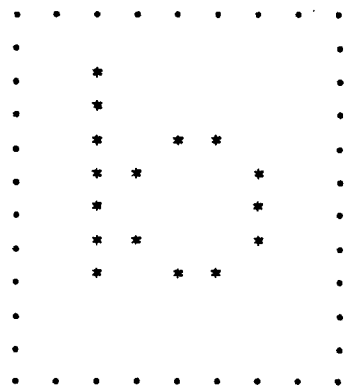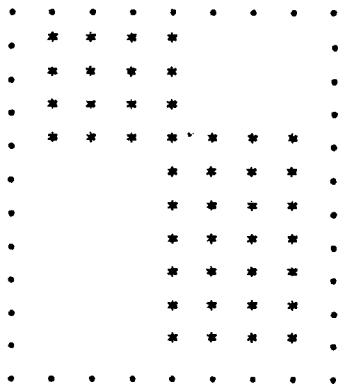
CODE A6 H
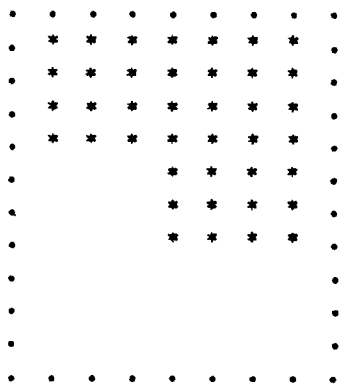
CODE A9 H
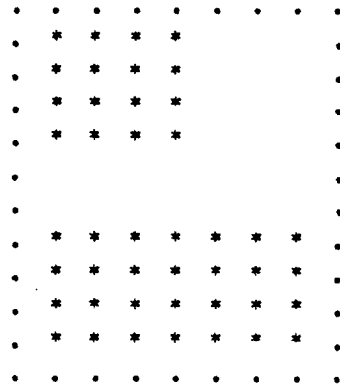
CODE A4 H
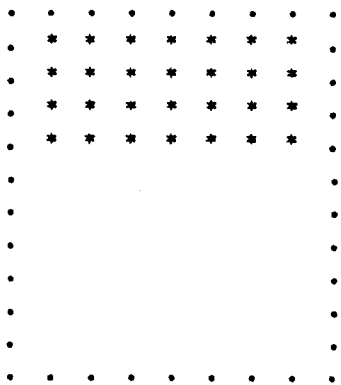
CODE A7 H
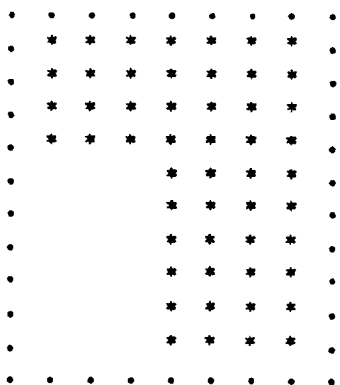
CODE AA H
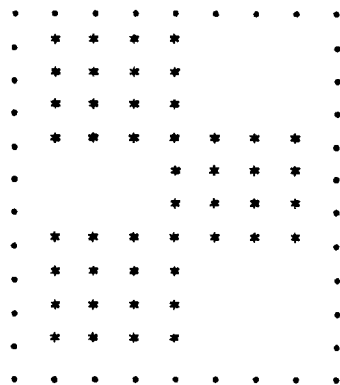
CODE A8 H
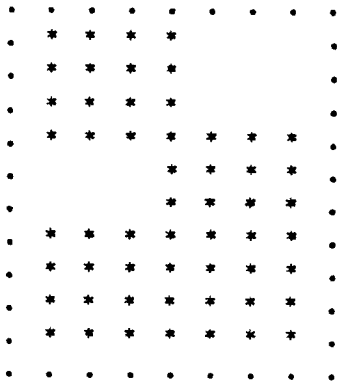
CODE AE H
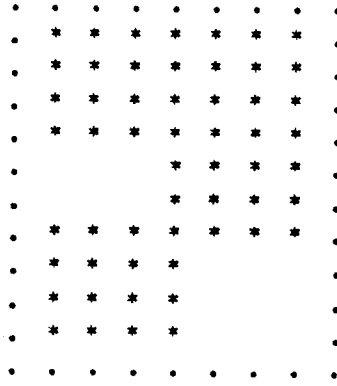
CODE B1 H
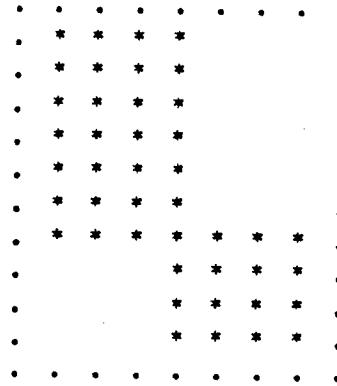
CODE AC H
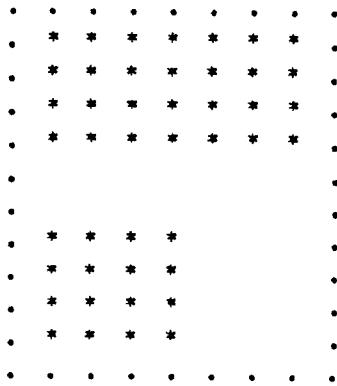
CODE AF H
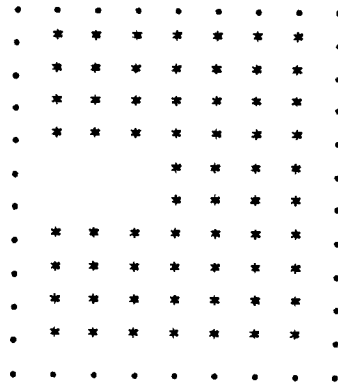
CODE B2 H
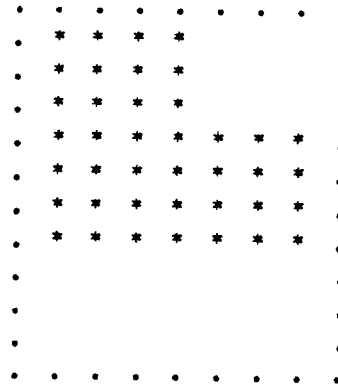
CODE AD H
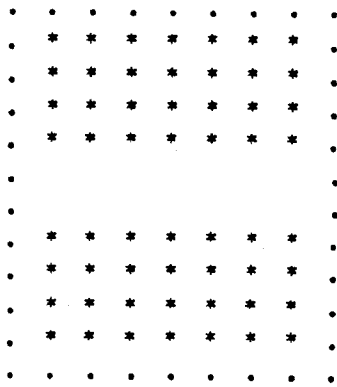
CODE 60 H
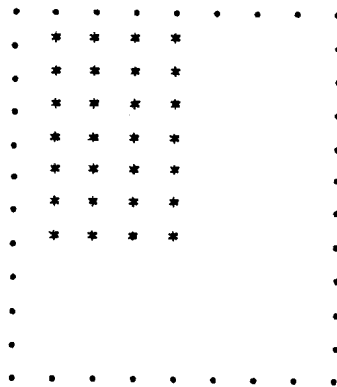
CODE B3 H
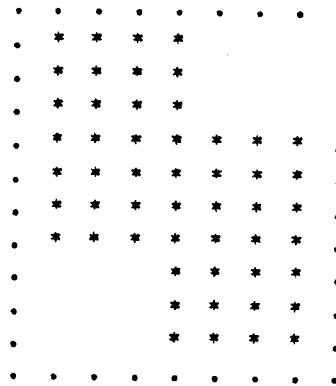
CODE B4 H
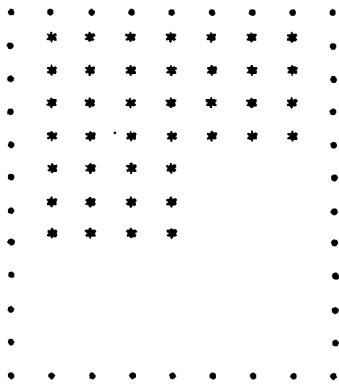
CODE B7 H
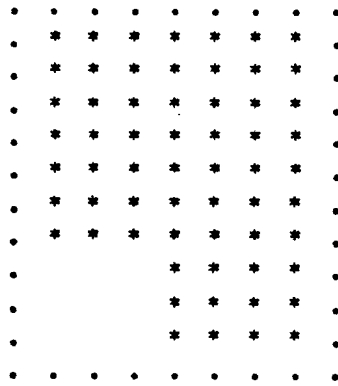
CODE BA H
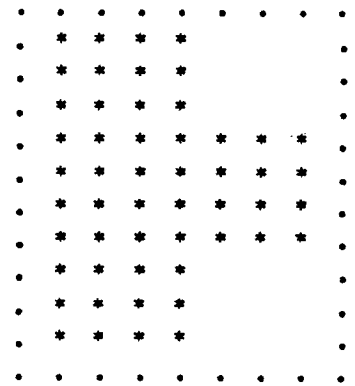
CODE B5 H
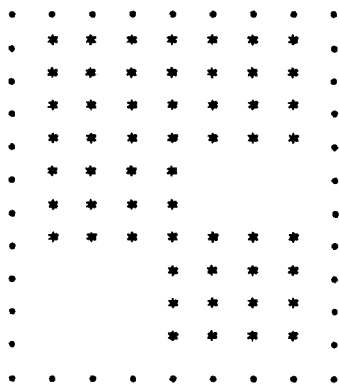
CODE B8 H
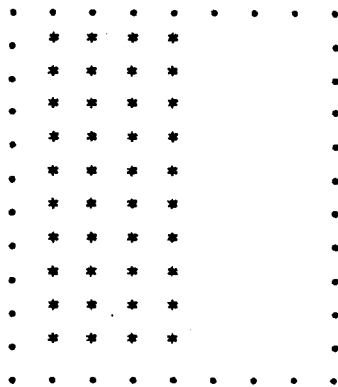
CODE BB H
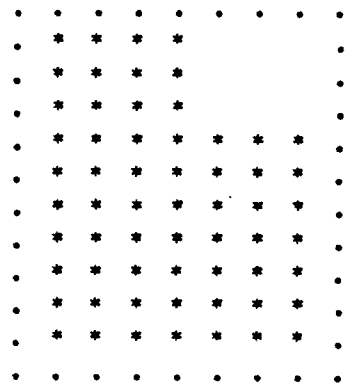
CODE B6 H
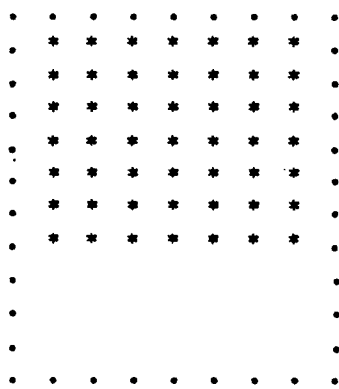
CODE B9 H
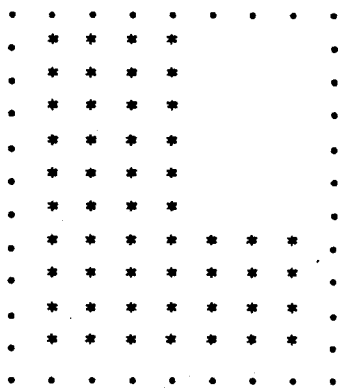
CODE BC H
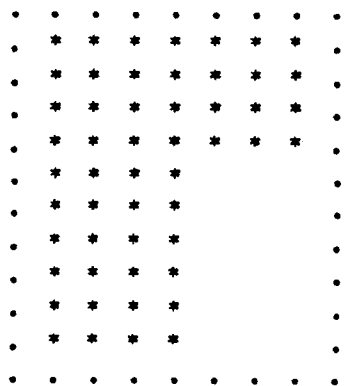
CODE BD H
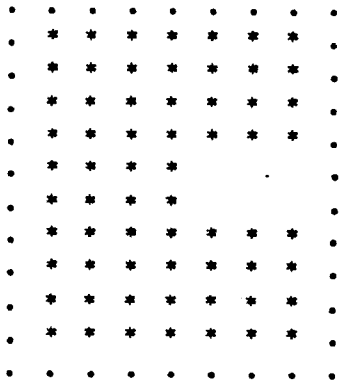
CODE C0 H
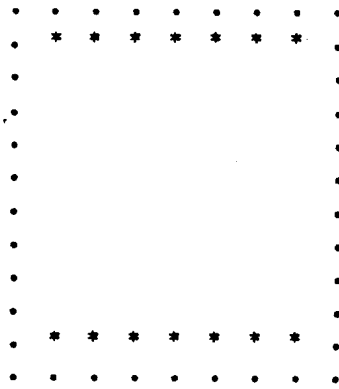
CODE C3 H
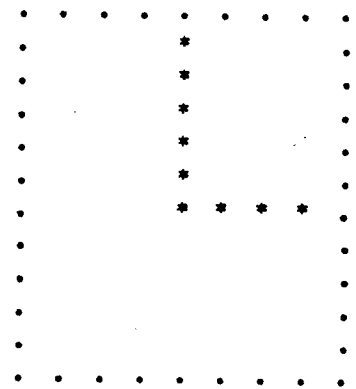
CODE BE H
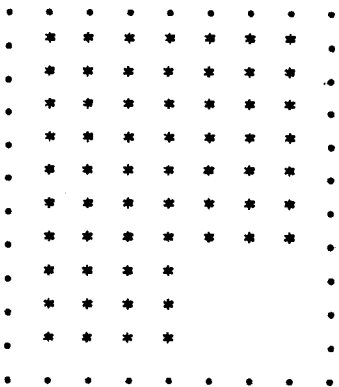
CODE C1 H
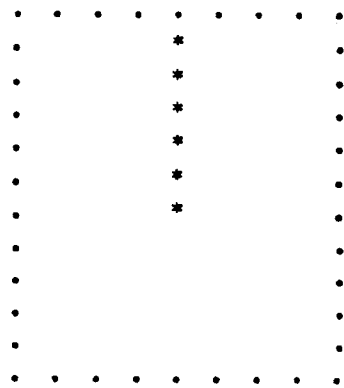
CODE C4 H
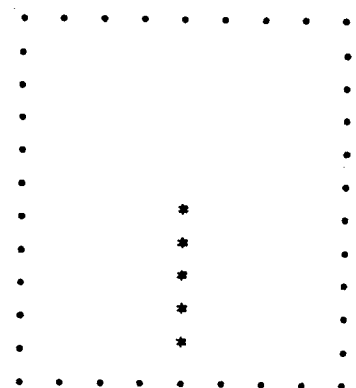
CODE BF H
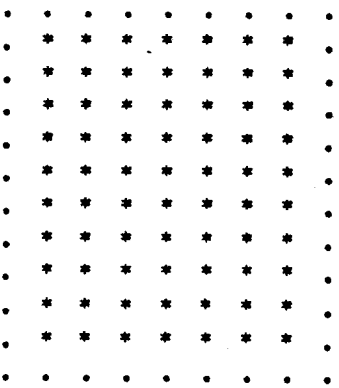
CODE C2 H
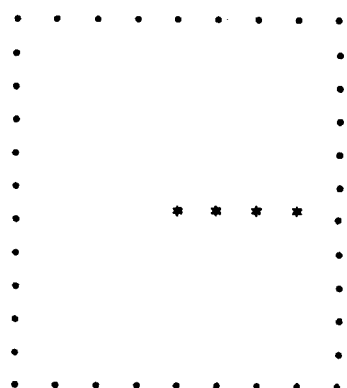
CODE C5 H
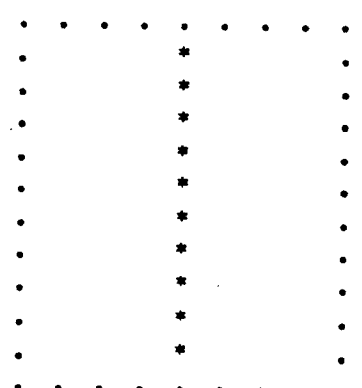
CODE C6 H
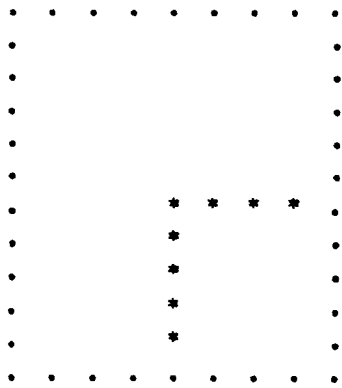
CODE C9 H
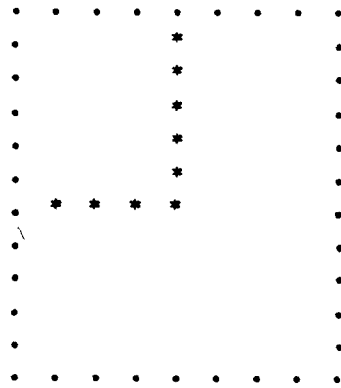
CODE CC H
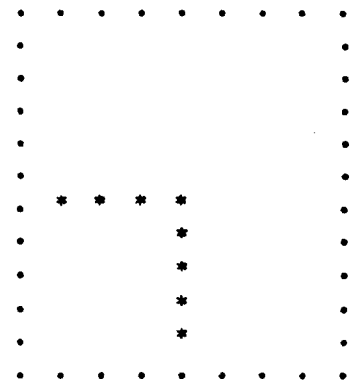
CODE C7 H
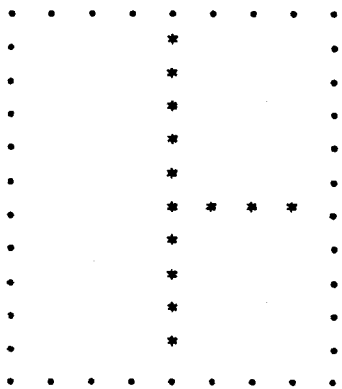
CODE CA H
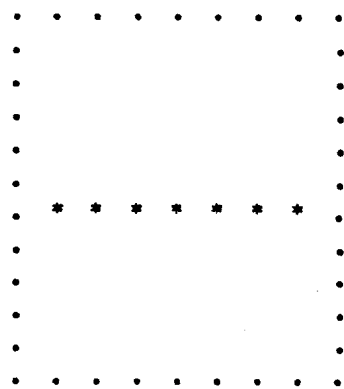
CODE CD H
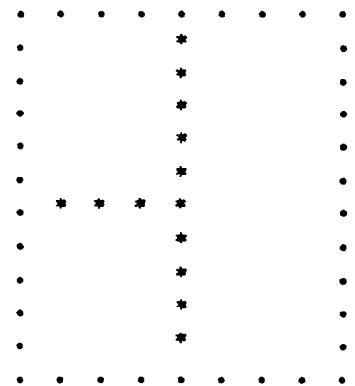
CODE C8 H
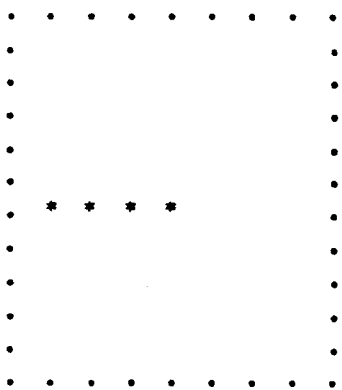
CODE CB H
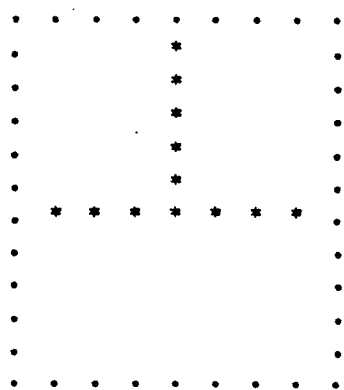
CODE CE H
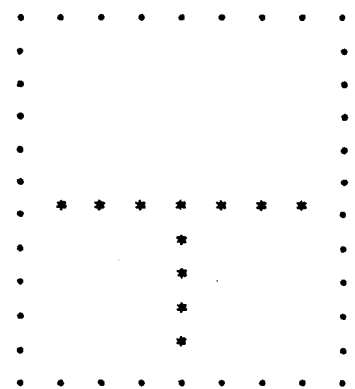
CODE CF H
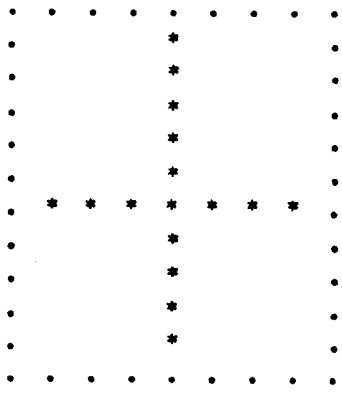
CODE D2 H
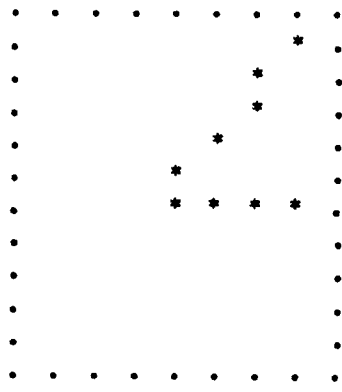
CODE D5 H
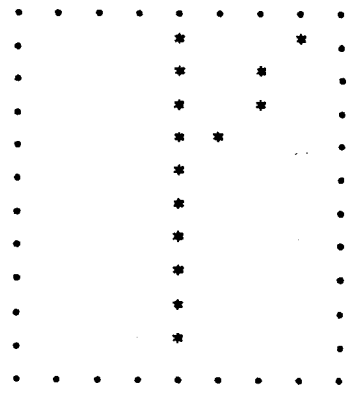
CODE D0 H
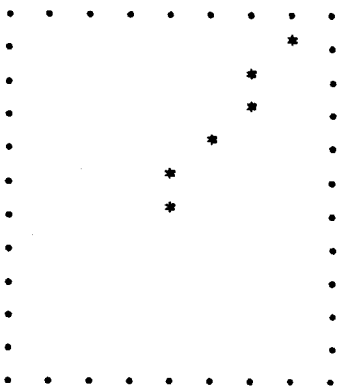
CODE D3 H
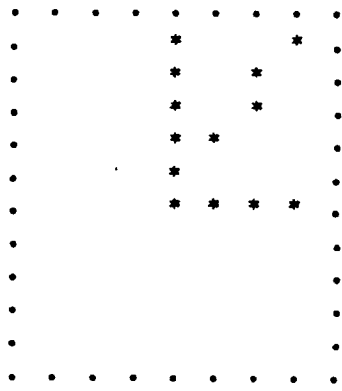
CODE D6 H
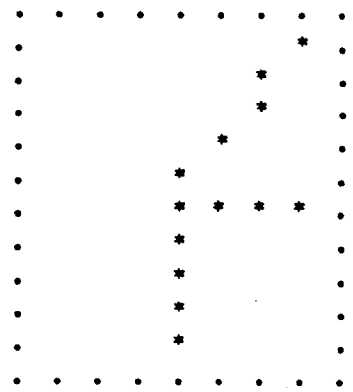
CODE D1 H
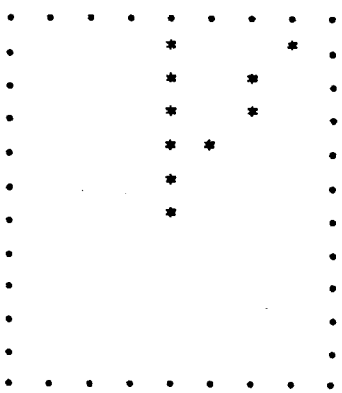
CODE D4 H
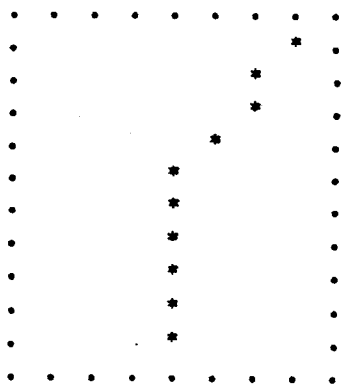
CODE D7 H
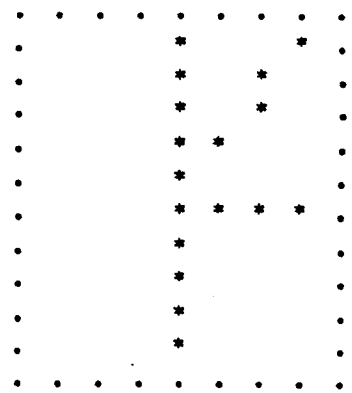
CODE D8 H
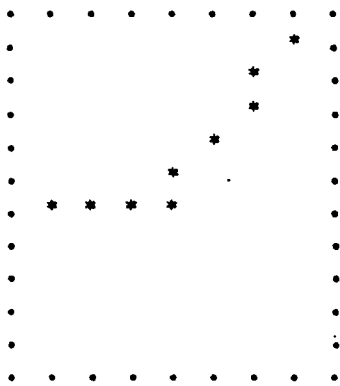
CODE DB H
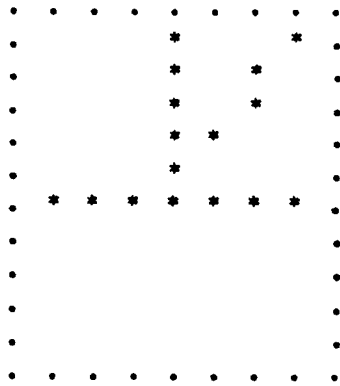
CODE DE H
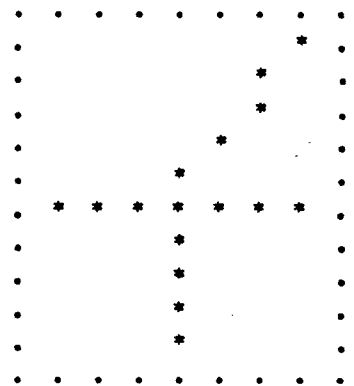
CODE D9 H
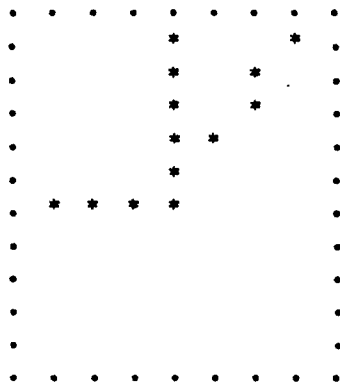
CODE DC H
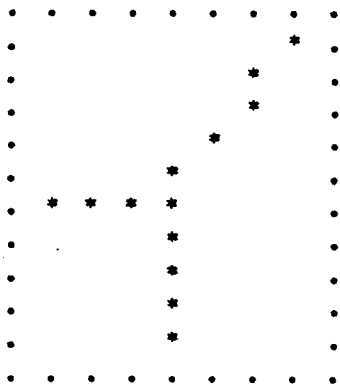
CODE DF H
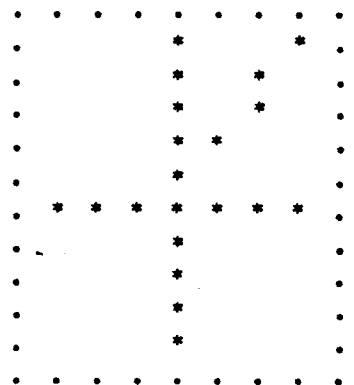
CODE DA H

CODE DD H

CODE E0 H

CODE E1 H

CODE E4 H

CODE E7 H

CODE E2 H

CODE E5 H

CODE E8 H

CODE E3 H

CODE E6 H

CODE E9 H

CODE EA H

CODE ED H

CODE F0 H

CODE EB H

CODE EE H

CODE F1 H

CODE EC H

CODE EF H

CODE F2 H

CODE F3 H

CODE F6 H

CODE F9 H

CODE F4 H

CODE F7 H

CODE FA H

CODE F5 H

CODE F8 H
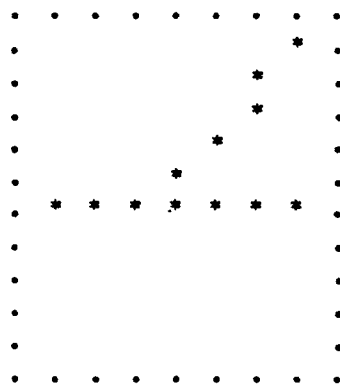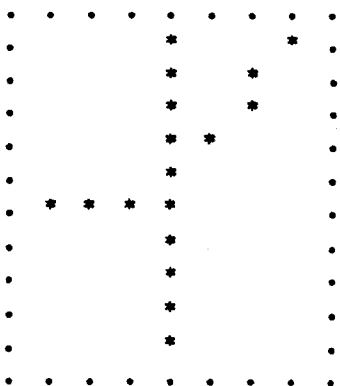
CODE FB H
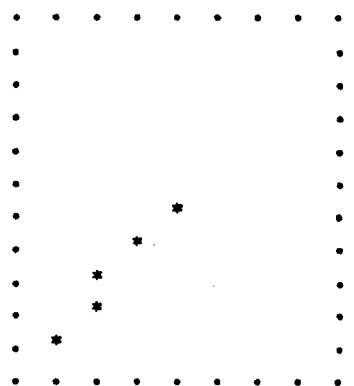
CODE FC H
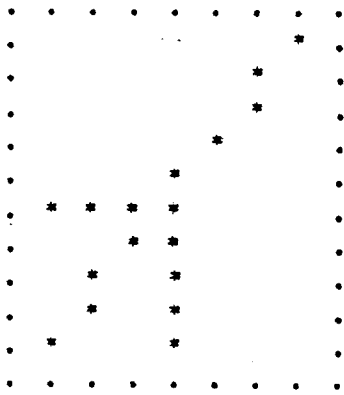
CODE FF H
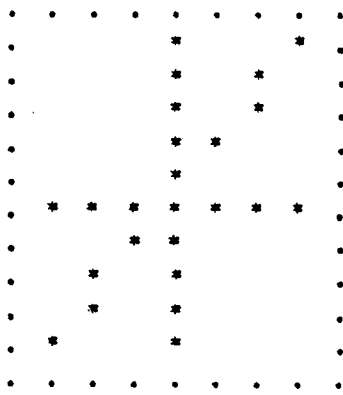
CODE FD H
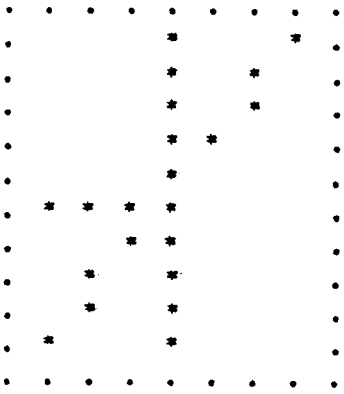
CODE FE H
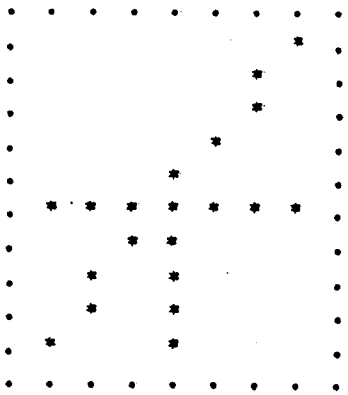
APPENDIX 2 ------------------------------------

X,Y POSITION CODES

Y OR X	CODE	ASCII
1	20 H	SPACE
2	21 H	!
3	22 H	"
4	23 H	#
5	24 H	$
6	25 H	%
7	26 H	&
8	27 H	'
9	28 H	(
10	29 H	)
11	2A H	*
12	2B H	+
13	2C H	`
14	2D H	-
15	2E H	.
16	2F H	/
17	30 H	0
18	31 H	1
19	32 H	2
20	33 H	3
21	34 H	4
22	35 H	5
23	36 H	6
24	37 H	7
25	38 H	8
26	39 H	9
27	3A H	:

X	CODE	ASCII
28	3B H	;
29	3C H	<
30	3D H	=
31	3E H	>
32	3F H	?
33	40 H	@
34	41 H	A
35	42 H	B
36	43 H	C
37	44 H	D
38	45 H	E
39	46 H	F
40	47 H	G
41	48 H	H
42	49 H	I
43	4A H	J
44	4B H	K
45	4C H	L
46	4D H	M
47	4E H	N
48	4F H	O
49	50 H	P
50	51 H	Q
51	52 H	R
52	53 H	S
53	54 H	T
54	55 H	U

X	CODE		ASCII
55	56	H	V
56	57	H	W
57	58	H	X
58	59	H	Y
59	5A	H	Z
60	5B	H	[
61	5C	H	\
62	5D	H	]
63	5E	H	^
64	5F	H	
65	60	H	←
66	61	H	a
67	62	H	b
68	63	H	c
69	64	H	d
70	65	H	e
71	66	H	f
72	67	H	g
73	68	H	h
74	69	H	i
75	6A	H	j
76	6B	H	k
77	6C	H	l
78	6D	H	m
79	6E	H	n
80	6F	H	o

APPENDIX 3 ————————————————

Simple VIO Driver

This appendix lists a simple VIO driver, assembled to reside at location 0100H. The driver assumes that the refresh memory resides in location F000 - F7FF. It will set-up the display with the following parameters:

       40 characters per line
       24 lines per page
       positive video display
       Mode 3 (graphics and ASCII characters may be displayed)

When it is desired to display a character on the screen, the user program should call the driver entry point (0100H) with the character code (for the character to be displayed) in the accumulator. Note that codes 00 - FFH (Appendix 1) may be displayed with the following three exceptions.

       CRTL-Z (1AH) will cause the screen to be erased
       CR (0DH) will be recognized as a carriage return/line feed.
       LF (0AH) will be ignored.

APPENDIX 3 ------------------

SIMPLE VIO DRIVER LISTING

```
                        ;***********************************************
                        ;VIO DRIVER
                        ;PUTS SCREEN IN 40/24 FORMAT AND SCROLLS AT
                        ;CARRIAGE RETURN/LINE FEED.
                        ;RECOGNIZED 1AH AS SCREEN ERASE
                        ;RECOGNIZES 0DH AS CARRIAGE RETURN/LINE FEED
                        ;IGNORES 0AH (LINE FEED)
                        ;***********************************************
     F7FF =             CTRPORT EQU 0F7FFH          ;HARDWARE CONTROL PORT
     F000 =             REFRESH EQU 0F000H          ;REFRESH MEMORY LOCATION
     0100                       ORG 100H
     0100 E5            VIODRV  PUSH H
     0101 C5                    PUSH B
     0102 D5                    PUSH D
     0103 F5                    PUSH PSW
     0104 21FFF7                LXI H,CTRPORT
     0107 36AD                  MVI M,0ADH          ;24 BY 40
     0109 FE1A                  CPI 1AH             ;INIT?
     010B C23301                JNZ VIO1
     010E 01C003      ERASE     LXI B,960
     0111 2100F0                LXI H,REFRESH
     0114 3620        ERASE1    MVI M,' '
     0116 0B                    DCX B
     0117 23                    INX H
     0118 78                    MOV A,B
     0119 B1                    ORA C
     011A C21401                JNZ ERASE1          ;CONTINUE UNTIL SCREEN ERASED
     011D 21C0F3      LNER1     LXI H,REFRESH+960          ;LAST LINE
     0120 0628                  MVI B,40            ;40 CHARS PER LINE
     0122 2B          LNER      DCX H
     0123 3620                  MVI M,' '
     0125 05                    DCR B
     0126 C22201                JNZ LNER
     0129 367F        RETURN    MVI M,7FH           ;CURSOR CHAR
     012B 226301                SHLD CURPTR
     012E F1                    POP PSW
     012F D1                    POP D
     0130 C1                    POP B
     0131 E1                    POP H               ;RESTORE USER STUFF
```

```
0132 C9                      RET
0133 2A6301     VIO1         LHLD CURPTR
0136 3620                    MVI M,' '           ;REMOVE CURSOR
0138 FE0A                    CPI 0AH             ;LINE FEED?
013A CA2901                  JZ RETURN           ;YES.,IGNORE IT
013D FE0D                    CPI 0DH             ;CARRIAGE RETURN?
013F C25801                  JNZ VIO2            ;NO,DO INDIVID CHAR
0142 2100F0     SCROLL       LXI H,REFRESH
0145 1128F0                  LXI D,REFRESH+40
0148 019803                  LXI B,920
014B 1A         SCRL1        LDAX D              ;GET CHAR
014C 77                      MOV M,A             ;MOVE IT
014D 23                      INX H
014E 13                      INX D
014F 0B                      DCX B
0150 78                      MOV A,B
0151 B1                      ORA C
0152 C24B01                  JNZ SCRL1
0155 C31D01                  JMP LNER1
0158 77         VIO2         MOV M,A             ;PUT CHAR ON SCREEN
0159 23                      INX H               ;BUMP CURSOR
015A 7D                      MOV A,L
015B FEC0                    CPI 0C0H            ;END OF LINE?
015D CA4201                  JZ SCROLL           ;YES
0160 C32901                  JMP RETURN
0163 0000       CURPTR       DW 0                ;CURSOR PTR STORAGE
                ;
0200                         ORG 200H
0200 310002     VIOTEST      LXI SP,200H
0203 DB03                    IN 3
0205 E602                    ANI 2
0207 CA0302                  JZ VIOTEST+3
020A DB02                    IN 2
020C CD0001                  CALL VIODRV
020F C30302                  JMP VIOTEST+3
0212                         END
```

APPENDIX 4 ————————————————

PROGRAMMING THE CHARACTER GENERATOR ROMS/PROMS

When it is desired to implement other character sets, the VIO's character generator ROMS/PROMS may be appropriately reprogrammed or replaced.

The three 2308/2708 ROMS/PROMs at locations U47, U48, and U49 contain the dot patterns for the entire 256 character set of the VIO. Each character consists of a 7x10 dot matrix which includes inter–character and inter–line spacing. The top eight rows of the 128 character represented by character codes 00–7F H, are stored in U49. The top eight rows of the second group of characters, (represented by codes 80–FF H), are stored in U48. U47 contains the dot patterns for the bottom two rows of the entire 256 character set.

NOTE: If only U49 is installed, the descenders of all lower case letters will be missing.

CHARACTER GENERATOR EXAMPLE

Figure III–22 shows an example of the ROM/EPROM entries for implimenting the character generator look–up tables. The figure shows a representation of the number "7" in a 5x7 dot format. The VIO uses a 7x10 dot format but this includes the inter–character and inter–line spacing. By including the spacing areas in the character matrix it is possible to form "no gap" graphic characters. Alphanumeric characters are normally limited to the 5x7 dot format. Each row (seven dots) of each character has its own entry in the look–up table. The "dots" are stored in a "negative logic" format, i.e., a zero corresponds to a dot while a one corresponds to a blank. The left–most dot position in a row corresponds to the least significant bit in the associated look–up table entry.

The ASCII code for the number "7" is HEX 37. The correspondance between codes and ROM/PROM addresses is shown in Tables III–2 and III–3. Since 37 falls in the lower half of the character set, Table III–2 should be used. Note that for this character code, scans (rows) 0–9 correspond to locations 037, 0B7, 137, 1B7, 237, 2B7, 337, 3B7 of U49 and 037, 0B7 of U47.

There are no dots in the zero row. Because of the negative logic format this corresponds to all 7 bits being equal to one. Therefore the entry 7F is made at location 037 of ROM U49.

Row one has dots at all locations except the first and last. This requires an entry of 41 in location 0B7 of U49. Row two has a dot only in position 5. Recall that the left–most dot corresponds to the least significant bit. This pattern therefore corresponds to a 5F entry in location 137 of U49.

The remaining entries are as shown in the figure. It is very important to remember that the entries are in negative logic format and that the left–most dot position corresponds to the least significant bit in the entry.

Users who wish to do extensive character set development may wish to use the program CHARGEN included with this chapter. This program is written in IMSAI Extended BASIC.

CODE=37

	0	1	2	3	4	5	6	DOT PATTERN		ROM LOCATION
0								7F ⟶		037
1		▨	▨	▨	▨	▨		41 ⟶		0B7
2						▨		5F ⟶		137
3					▨			6F ⟶		1B7
4				▨				77 ⟶		237
5			▨					7B ⟶		2B7
6		▨						7D ⟶		337
7		▨						7D ⟶		3B7
8								7F ⟶		037
9								7F ⟶		0B7

U49 (rows 0–7), U47 (rows 8–9)
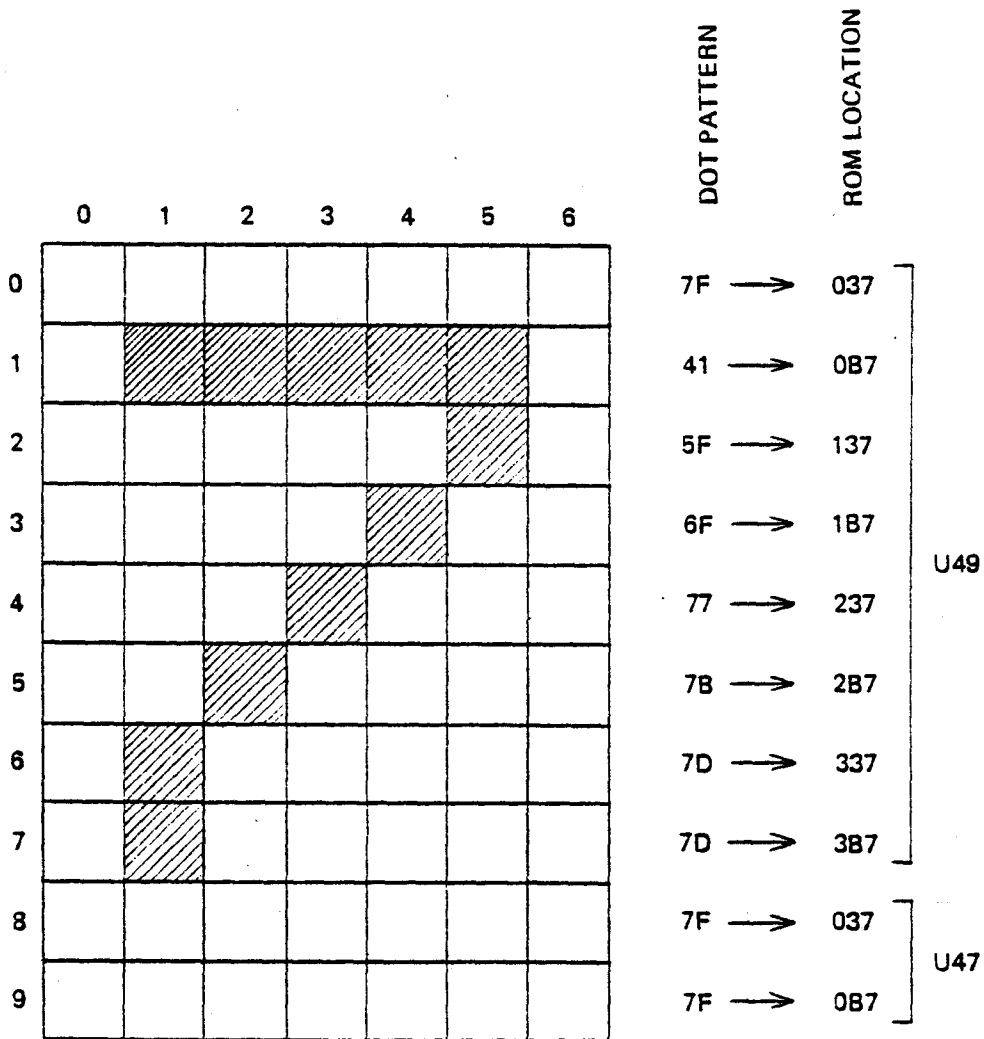
FIGURE III-22 CHARACTER GENERATOR EXAMPLE

III – 103

TABLE III-2. LOW 128 FONT ROM ADDRESSES

CODE	ROM ADDRESS (U49)									(U47)		SCAN
	1	2	3	4	5	6	7	8		9	10	
00	000	080	100	180	200	280	300	380	--	000	080
01	001	081	101	181	201	281	301	381	--	001	081
02	002	082	102	182	202	282	302	382	--	002	082
03	003	083	103	183	203	283	303	383	--	003	083
04	004	084	104	184	204	284	304	384	--	004	084
05	005	085	105	185	205	285	305	385	--	005	085
06	006	086	106	186	206	286	306	386	--	006	086
07	007	087	107	187	207	287	307	387	--	007	087
08	008	088	108	188	208	288	308	388	--	008	088
09	009	089	109	189	209	289	309	389	--	009	089
0A	00A	08A	10A	18A	20A	28A	30A	38A	--	00A	08A
0B	00B	083	10B	18B	20B	28B	30B	38B	--	00B	08B
0C	00C	08C	10C	18C	20C	28C	30C	38C	--	00C	08C
0D	00D	08D	10D	18D	20D	28D	30D	38D	--	00D	08D
0E	00E	08E	10E	18E	20E	28E	30E	38E	--	00E	08E
0F	00F	08F	10F	18F	20F	28F	30F	38F	--	00F	08F
10	010	090	110	190	210	290	310	390	--	010	090
11	011	091	111	191	211	291	311	391	--	011	091
12	012	092	112	192	212	292	312	392	--	012	092
13	013	093	113	193	213	293	313	393	--	013	093
14	014	094	114	194	214	294	314	394	--	014	094
15	015	095	115	195	215	295	315	395	--	015	095
16	016	096	116	196	216	296	316	396	--	016	096
17	017	097	117	197	217	297	317	397	--	017	097
18	018	098	118	198	218	298	318	398	--	018	098
19	019	099	119	199	219	299	319	399	--	019	099
1A	01A	09A	11A	19A	21A	29A	31A	39A	--	01A	09A
1B	01B	09B	11B	19B	21B	29B	31B	39B	--	01B	09B
1C	01C	09C	11C	19C	21C	29C	31C	39C	--	01C	09C
1D	01D	09D	11D	19D	21D	29D	31D	39D	--	01D	09D
1E	01E	09E	11E	19E	21E	29E	31E	39E	--	01E	09E
1F	01F	09F	11F	19F	21F	29F	31F	39F	--	01F	09F
20	020	0A0	120	1A0	220	2A0	320	3A0	--	020	0A0
21	021	0A1	121	1A1	221	2A1	321	3A1	--	021	0A1
22	022	0A2	122	1A2	222	2A2	322	3A2	--	022	0A2
23	023	0A3	123	1A3	223	2A3	323	3A3	--	023	0A3
24	024	0A4	124	1A4	224	2A4	324	3A4	--	024	0A4
25	025	0A5	125	1A5	225	2A5	325	3A5	--	025	0A5
26	026	0A6	126	1A6	226	2A6	326	3A6	--	026	0A6
27	027	0A7	127	1A7	227	2A7	327	3A7	--	027	0A7
28	028	0A8	128	1A8	228	2A8	328	3A8	--	028	0A8
29	029	0A9	129	1A9	229	2A9	329	3A9	--	029	0A9
2A	02A	0AA	12A	1AA	22A	2AA	32A	3AA	--	02A	0AA
2B	02B	0AB	12B	1AB	22B	2AB	32B	3AB	--	02B	0AB

2C	02C	0AC	12C	1AC	22C	2AC	32C	3AC	--	02C	0AC
2D	02D	0AD	12D	1AD	22D	2AD	32D	3AD	--	02D	0AD
2E	02E	0AE	12E	1AE	22E	2AE	32E	3AE	--	02E	0AE
2F	02F	0AF	12F	1AF	22F	2AF	32F	3AF	--	02F	0AF
30	030	0B0	130	1B0	230	2B0	330	3B0	--	030	0B0
31	031	0B1	131	1B1	231	2B1	331	3B1	--	031	0B1
32	032	0B2	132	1B2	232	2B2	332	3B2	--	032	0B2
33	033	0B3	133	1B3	233	2B3	333	3B3	--	033	0B3
34	034	0B4	134	1B4	234	2B4	334	3B4	--	034	0B4
35	035	0B5	135	1B5	235	2B5	335	3B5	--	035	0B5
36	036	0B6	136	1B6	236	2B6	336	3B6	--	036	0B6
37	037	0B7	137	1B7	237	2B7	337	3B7	--	037	0B7
38	038	0B8	138	1B8	238	2B8	338	3B8	--	038	0B8
39	039	0B9	139	1B9	239	2B9	339	3B9	--	039	0B9
3A	03A	0BA	13A	1BA	23A	2BA	33A	3BA	--	03A	0BA
3B	03B	0BB	13B	1BB	23B	2BB	33B	3BB	--	03B	0BB
3C	03C	0BC	13C	1BC	23C	2BC	33C	3BC	--	03C	0BC
3D	03D	0BD	13D	1BD	23D	2BD	33D	3BD	--	03D	0BD
3E	03E	0BE	13E	1BE	23E	2BE	33E	3BE	--	03E	0BE
3F	03F	0BF	13F	1BF	23F	2BF	33F	3BF	--	03F	0BF
40	040	0C0	140	1C0	240	2C0	340	3C0	--	040	0C0
41	041	0C1	141	1C1	241	2C1	341	3C1	--	041	0C1
42	042	0C2	142	1C2	242	2C2	342	3C2	--	042	0C2
43	043	0C3	143	1C3	243	2C3	343	3C3	--	043	0C3
44	044	0C4	144	1C4	244	2C4	344	3C4	--	044	0C4
45	045	0C5	145	1C5	245	2C5	345	3C5	--	045	0C5
46	046	0C6	146	1C6	246	2C6	346	3C6	--	046	0C6
47	047	0C7	147	1C7	247	2C7	347	3C7	--	047	0C7
48	048	0C8	148	1C8	248	2C8	348	3C8	--	048	0C8
49	049	0C9	149	1C9	249	2C9	349	3C9	--	049	0C9
4A	04A	0CA	14A	1CA	24A	2CA	34A	3CA	--	04A	0CA
4B	04B	0CB	14B	1CB	24B	2CB	34B	3CB	--	04B	0CB
4C	04C	0CC	14C	1CC	24C	2CC	34C	3CC	--	04C	0CC
4D	04D	0CD	14D	1CD	24D	2CD	34D	3CD	--	04D	0CD
4E	04E	0CE	14E	1CE	24E	2CE	34E	3CE	--	04E	0CE
4F	04F	0CF	14F	1CF	24F	2CF	34F	3CF	--	04F	0CF
50	050	0D0	150	1D0	250	2D0	350	3D0	--	050	0D0
51	051	0D1	151	1D1	251	2D1	351	3D1	--	051	0D1
52	052	0D2	152	1D2	252	2D2	352	3D2	--	052	0D2
53	053	0D3	153	1D3	253	2D3	353	3D3	--	053	0D3
54	054	0D4	154	1D4	254	2D4	354	3D4	--	054	0D4
55	055	0D5	155	1D5	255	2D5	355	3D5	--	055	0D5
56	056	0D6	156	1D6	256	2D6	356	3D6	--	056	0D6
57	057	0D7	157	1D7	257	2D7	357	3D7	--	057	0D7
58	058	0D8	158	1D8	258	2D8	358	3D8	--	058	0D8
59	059	0D9	159	1D9	259	2D9	359	3D9	--	059	0D9
5A	05A	0DA	15A	1DA	25A	2DA	35A	3DA	--	05A	0DA
5B	05B	0DB	15B	1DB	25B	2DB	35B	3DB	--	05B	0DB

	05C	0DC	15C	1DC	25C	2DC	35C	3DC	--	05C	0DC
5C	05C	0DC	15C	1DC	25C	2DC	35C	3DC	--	05D	0DD
5D	05D	0DD	15D	1DD	25D	2DD	35D	3DD	--	05E	0DE
5E	05E	0DE	15E	1DE	25E	2DE	35E	3DE	--	05F	0DF
5F	05F	0DF	15F	1DF	25F	2DF	35F	3DF	--	060	0E0
60	060	0E0	160	1E0	260	2E0	360	3E0	--	061	0E1
61	061	0E1	161	1E1	261	2E1	361	3E1	--	062	0E2
62	062	0E2	162	1E2	262	2E2	363	3E3	--	063	0E3
63	063	0E3	163	1E3	263	2E3	364	3E4	--	064	0E4
64	064	0E4	164	1E4	264	2E4	365	3E5	--	065	0E5
65	065	0E5	165	1E5	265	2E5	366	3E6	--	066	0E6
66	066	0E6	166	1E6	266	2E6	367	3E7	--	067	0E7
67	067	0E7	167	1E7	267	2E7	368	3E8	--	068	0E8
68	068	0E8	168	1E8	268	2E8	369	3E9	--	069	0E9
69	069	0E9	169	1E9	269	2E9	36A	3EA	--	06A	0EA
6A	06A	0EA	16A	1EA	26A	2EA	36B	3EB	--	06B	0EB
6B	06B	0EB	16B	1EB	26B	2EB	36C	3EC	--	06C	0EC
6C	06C	0EC	16C	1EC	26C	2EC	36D	3ED	--	06D	0ED
6D	06D	0ED	16D	1ED	26D	2ED	36E	3EE	--	06E	0EE
6E	06E	0EE	16E	1EE	26E	2EE	36F	3EF	--	06F	0EF
6F	06F	0EF	16F	1EF	26F	2EF	370	3F0	--	070	0F0
70	070	0F0	170	1F0	270	2F0	371	3F1	--	071	0F1
71	071	0F1	171	1F1	271	2F1	372	3F2	--	072	0F2
72	072	0F2	172	1F2	272	2F2	373	3F3	--	073	0F3
73	073	0F3	173	1F3	273	2F3	374	3F4	--	074	0F4
74	074	0F4	174	1F4	274	2F4	375	3F5	--	075	0F5
75	075	0F5	175	1F5	275	2F5	376	3F6	--	076	0F6
76	076	0F6	176	1F6	276	2F6	377	3F7	--	077	0F7
77	077	0F7	177	1F7	277	2F7	378	3F8	--	078	0F8
78	078	0F8	178	1F8	278	2F8	379	3F9	--	079	0F9
79	079	0F9	179	1F9	279	2F9	37A	3FA	--	07A	0FA
7A	07A	0FA	17A	1FA	27A	2FA	37B	3FB	--	07B	0FB
7B	07B	0FB	17B	1FB	27B	2FB	37C	3FC	--	07C	0FC
7C	07C	0FC	17C	1FC	27C	2FC	37D	3FD	--	07D	0FD
7D	07D	0FD	17D	1FD	27D	2FD	37E	3FE	--	07E	0FE
7E	07E	0FE	17E	1FE	27E	2FE	37F	3FF	--	07F	0FF
7F	07F	0FF	17F	1FF	27F	2FF					

TABLE III-3.   HIGH 128 FONT ROM ADDRESSES

CODE	ROM ADDRESS (U48)									(U47)		SCAN
	1	2	3	4	5	6	7	8		9	10	
80	000	080	100	180	200	280	300	380	--	200	280	
81	001	081	101	181	201	281	301	381	--	201	281	
82	002	082	102	182	202	282	302	382	--	202	282	
83	003	083	103	183	203	283	303	383	--	203	283	
84	004	084	104	184	204	284	304	384	--	204	284	
85	005	085	105	185	205	285	305	385	--	205	285	
86	006	086	106	186	206	286	306	386	--	206	286	
87	007	087	107	187	207	287	307	387	--	207	287	
88	008	088	108	188	208	288	308	388	--	208	288	
89	009	089	109	189	209	289	309	389	--	209	289	
8A	00A	08A	10A	18A	20A	28A	30A	38A	--	20A	28A	
8B	00B	08B	10B	18B	20B	28B	30B	38B	--	20B	28B	
8C	00C	08C	10C	18C	20C	28C	30C	38C	--	20C	28C	
8D	00D	08D	10D	18D	20D	28D	30D	38D	--	20D	28D	
8E	00E	08E	10E	18E	20E	28E	30E	38E	--	20E	28E	
8F	00F	08F	10F	18F	20F	28F	30F	38F	--	20F	28F	
90	010	090	110	190	210	290	310	390	--	210	290	
91	011	091	111	191	211	291	311	391	--	211	291	
92	012	092	112	192	212	292	312	392	--	212	292	
93	013	093	113	193	213	293	313	393	--	213	293	
94	014	094	114	194	214	294	314	394	--	214	294	
95	015	095	115	195	215	295	315	395	--	215	295	
96	016	096	116	196	216	296	316	396	--	216	296	
97	017	097	117	197	217	297	317	397	--	217	297	
98	018	098	118	198	218	298	318	398	--	218	298	
99	019	099	119	199	219	299	319	399	--	219	299	
9A	01A	09A	11A	19A	21A	29A	31A	39A	--	21A	29A	
9B	01B	09B	11B	19B	21B	29B	31B	39B	--	21B	29B	
9C	01C	09C	11C	19C	21C	29C	31C	39C	--	21C	29C	
9D	01D	09D	11D	19D	21D	29D	31D	39D	--	21D	29D	
9E	01E	09E	11E	19E	21E	29E	31E	39E	--	21E	29E	
9F	01F	09F	11F	19F	21F	29F	31F	39F	--	21F	29F	
A0	020	0A0	120	1A0	220	2A0	320	3A0	--	220	2A0	
A1	021	0A1	121	1A1	221	2A1	321	3A1	--	221	2A1	
A2	022	0A2	122	1A2	222	2A2	322	3A2	--	222	2A2	
A3	023	0A3	123	1A3	223	2A3	323	3A3	--	223	2A3	
A4	024	0A4	124	1A4	224	2A4	324	3A4	--	224	2A4	
A5	025	0A5	125	1A5	225	2A5	325	3A5	--	225	2A5	
A6	026	0A6	126	1A6	226	2A6	326	3A6	--	226	2A6	
A7	027	0A7	127	1A7	227	2A7	327	3A7	--	227	2A7	
A8	028	0A8	128	1A8	228	2A8	328	3A8	--	228	2A8	
A9	029	0A9	129	1A9	229	2A9	329	3A9	--	229	2A9	
AA	02A	0AA	12A	1AA	22A	2AA	32A	3AA	--	22A	2AA	
AB	02B	0AB	12B	1AB	22B	2AB	32B	3AB	--	22B	2AB	

AC	02C	0AC	12C	1AC	22C	2AC	32C	3AC	--	22C	2AC
AD	02D	0AD	12D	1AD	22D	2AD	32D	3AD	--	22D	2AD
AE	02E	0AE	12E	1AE	22E	2AE	32E	3AE	--	22E	2AE
AF	02F	0AF	12F	1AF	22F	2AF	32F	3AF	--	22F	2AF
B0	030	0B0	130	1B0	230	2B0	330	3B0	--	230	2B0
B1	031	0B1	131	1B1	231	2B1	331	3B1	--	231	2B1
B2	032	0B2	132	1B2	232	2B2	332	3B2	--	232	2B2
B3	033	0B3	133	1B3	233	2B3	333	3B3	--	233	2B3
B4	034	0B4	134	1B4	234	2B4	334	3B4	--	234	2B4
B5	035	0B5	135	1B5	235	2B5	335	3B5	--	235	2B5
B6	036	0B6	136	1B6	236	2B6	336	3B6	--	236	2B6
B7	037	0B7	137	1B7	237	2B7	337	3B7	--	237	2B7
B8	038	0B8	138	1B8	238	2B8	338	3B8	--	238	2B8
B9	039	0B9	139	1B9	239	2B9	339	3B9	--	239	2B9
BA	03A	0BA	13A	1BA	23A	2BA	33A	3BA	--	23A	2BA
BB	03B	0BB	13B	1BB	23B	2BB	33B	3BB	--	23B	2BB
BC	03C	0BC	13C	1BC	23C	2BC	33C	3BC	--	23C	2BC
BD	03D	0BD	13D	1BD	23D	2BD	33D	3BD	--	23D	2BD
BE	03E	0BE	13E	1BE	23E	2BE	33E	3BE	--	23E	2BE
BF	03F	0BF	13F	1BF	23F	2BF	33F	3BF	--	23F	2BF
C0	040	0C0	140	1C0	240	2C0	340	3C0	--	240	2C0
C1	041	0C1	141	1C1	241	2C1	341	3C1	--	241	2C1
C2	042	0C2	142	1C2	242	2C2	342	3C2	--	242	2C2
C3	043	0C3	143	1C3	243	2C3	343	3C3	--	243	2C3
C4	044	0C4	144	1C4	244	2C4	344	3C4	--	244	2C4
C5	045	0C5	145	1C5	245	2C5	345	3C5	--	245	2C5
C6	046	0C6	146	1C6	246	2C6	346	3C6	--	246	2C6
C7	047	0C7	147	1C7	247	2C7	347	3C7	--	247	2C7
C8	048	0C8	148	1C8	248	2C8	348	3C8	--	248	2C8
C9	049	0C9	149	1C9	249	2C9	349	3C9	--	249	2C9
CA	04A	0CA	14A	1CA	24A	2CA	34A	3CA	--	24A	2CA
CB	04B	0CB	14B	1CB	24B	2CB	34B	3CB	--	24B	2CB
CC	04C	0CC	14C	1CC	24C	2CC	34C	3CC	--	24C	2CC
CD	04D	0CD	14D	1CD	24D	2CD	34D	3CD	--	24D	2CD
CE	04E	0CE	14E	1CE	24E	2CE	34E	3CE	--	24E	2CE
CF	04F	0CF	14F	1CF	24F	2CF	34F	3CF	--	24F	2CF
D0	050	0D0	150	1D0	250	2D0	350	3D0	--	250	2D0
D1	051	0D1	151	1D1	251	2D1	351	3D1	--	251	2D1
D2	052	0D2	152	1D2	252	2D2	352	3D2	--	252	2D2
D3	053	0D3	153	1D3	253	2D3	353	3D3	--	253	2D3
D4	054	0D4	154	1D4	254	2D4	354	3D4	--	254	2D4
D5	055	0D5	155	1D5	255	2D5	355	3D5	--	255	2D5
D6	056	0D6	156	1D6	256	2D6	356	3D6	--	256	2D6
D7	057	0D7	157	1D7	257	2D7	357	3D7	--	257	2D7
D8	058	0D8	158	1D8	258	2D8	358	3D8	--	258	2D8
D9	059	0D9	159	1D9	259	2D9	359	3D9	--	259	2D9
DA	05A	0DA	15A	1DA	25A	2DA	35A	3DA	--	25A	2DA
DB	05B	0DB	15B	1DB	25B	2DB	35B	3DB	--	25B	2DB

DC	05C	0DC	15C	1DC	25C	2DC	35C	3DC	--	25C	2DC
DD	05D	0DD	15D	1DD	25D	2DD	35D	3DD	--	25D	2DD
DE	05E	0DE	15E	1DE	25E	2DE	35E	3DE	--	25E	2DE
DF	05F	0DF	15F	1DF	25F	2DF	35F	3DF	--	25F	2DF
E0	060	0E0	160	1E0	260	2E0	360	3E0	--	260	2E0
E1	061	0E1	161	1E1	261	2E1	361	3E1	--	261	2E1
E2	062	0E2	162	1E2	262	2E2	362	3E2	--	262	2E2
E3	063	0E3	163	1E3	263	2E3	363	3E3	--	263	2E3
E4	064	0E4	164	1E4	264	2E4	364	3E4	--	264	2E4
E5	065	0E5	165	1E5	265	2E5	365	3E5	--	265	2E5
E6	066	0E6	166	1E6	266	2E6	366	3E6	--	266	2E6
E7	067	0E7	167	1E7	267	2E7	367	3E7	--	267	2E7
E8	068	0E8	168	1E8	268	2E8	368	3E8	--	268	2E8
E9	069	0E9	169	1E9	269	2E9	369	3E9	--	269	2E9
EA	06A	0EA	16A	1EA	26A	2EA	36A	3EA	--	26A	2EA
EB	06B	0EB	16B	1EB	26B	2EB	36B	3EB	--	26B	2EB
EC	06C	0EC	16C	1EC	26C	2EC	36C	3EC	--	26C	2EC
ED	06D	0ED	16D	1ED	26D	2ED	36D	3ED	--	26D	2ED
EE	06E	0EE	16E	1EE	26E	2EE	36E	3EE	--	26E	2EE
EF	06F	0EF	16F	1EF	26F	2EF	36F	3EF	--	26F	2EF
F0	070	0F0	170	1F0	270	2F0	370	3F0	--	270	2F0
F1	071	0F1	171	1F1	271	2F1	371	3F1	--	271	2F1
F2	072	0F2	172	1F2	272	2F2	372	3F2	--	272	2F2
F3	073	0F3	173	1F3	273	2F3	373	3F3	--	273	2F3
F4	074	0F4	174	1F4	274	2F4	374	3F4	--	274	2F4
F5	075	0F5	175	1F5	275	2F5	375	3F5	--	275	2F5
F6	076	0F6	176	1F6	276	2F6	376	3F6	--	276	2F6
F7	077	0F7	177	1F7	277	2F7	377	3F7	--	277	2F7
F8	078	0F8	178	1F8	278	2F8	378	3F8	--	278	2F8
F9	079	0F9	179	1F9	279	2F9	379	3F9	--	279	2F9
FA	07A	0FA	17A	1FA	27A	2FA	37A	3FA	--	27A	2FA
FB	07B	0FB	17B	1FB	27B	2FB	37B	3FB	--	27B	2FB
FC	07C	0FC	17C	1FC	27C	2FC	37C	3FC	--	27C	2FC
FD	07D	0FD	17D	1FD	27D	2FD	37D	3FD	--	27D	2FD
FE	07E	0FE	17E	1FE	27E	2FE	37E	3FE	--	27E	2FE
FF	07F	0FF	17F	1FF	27F	2FF	37F	3FF	--	27F	2FF

CHARACTER GENERATOR PROGRAM

The program CHARGEN.BAS was used to develop the standard character font for the IMSAI VIO. Users who wish to do extensive font development may wish to use this or a similar program of their own design. CHARGEN is written in IMSAI Extended BASIC and requires at least 32K of memory. It allows the user to interactively create fonts and generates HEX files for PROM programming.

The following is a sample session with CHARGEN:

```
        A>RUN-E CHARGEN
BASIC-E INTERPRETER - VER 2.2


NAME OF CHARACTER FILE? PFONT
NEW FILE (Y/N)? Y



ENTER CODE OR COMMAND? 00000000
INPUT CHARACTER MATRIX

LINE 1          .? "
LINE 2          .? " *****
LINE 3          .? " *    *
LINE 4          .? 2
ILLEGAL CHARACTER
LINE 4          .? " *    *
LINE 5          .? " *    *
LINE 6          .? " *    *
LINE 7          .? " *****
LINE 8          .? "
LINE 9          .? "
LINE 10         .? "
```

```
CODE IS 00000000

CHARACTER IS:
.  .  .  .  .  .  .  .  .  .
.                          .
.     *  *  *  *  *        .
.     *           *        .
.     *           *        .
.     *           *        .
.     *           *        .
.     *  *  *  *  *        .
.                          .
.                          .
.                          .
.  .  .  .  .  .  .  .  .  .


CHARACTER OK (Y/N)? Y




ENTER CODE OR COMMAND? 0000001
ENTER 8 BINARY DIGITS



ENTER CODE OR COMMAND? 00000001
INPUT CHARACTER MATRIX

LINE 1          .? "*
LINE 2          .? "  *
LINE 3          .? "   *
LINE 4          .? "     *
LINE 5          .? "       *
LINE 6          .? "        *
LINE 7          .? "          *
LINE 8          .? "            *
LINE TOO LONG
LINE 8          .? "
LINE 9          .? "
LINE 10         .? "
```

CODE IS 00000001

CHARACTER IS:

```
.  .  .  .  .  .  .  .  .
.  *                    .
.     *                 .
.        *              .
.           *           .
.              *        .
.                 *     .
.                    *  .
.                       .
.                       .
.                       .
.                       .
.  .  .  .  .  .  .  .  .
```

CHARACTER OK (Y/N)? N
REENTER CHARACTER
INPUT CHARACTER MATRIX

```
LINE 1          .?  "              *
LINE 2          .?  "           *
LINE 3          .?  "        *
LINE 4          .?  "     *
LINE 5          .?  "  *
LINE 6          .?  " *
LINE 7          .?  "*
LINE 8          .?  "
LINE 9          .?  "
LINE 10         .?  "
```

CODE IS 00000001

CHARACTER IS:

```
.  .  .  .  .  .  .  .  .
.                 *     .
.              *        .
.           *           .
.        *              .
.     *                 .
.  *                    .
.                       .
.                       .
.                       .
.                       .
.  .  .  .  .  .  .  .  .
```

CHARACTER OK (Y/N)? Y

ENTER CODE OR COMMAND? E00000000

CODE IS 00000000

CHARACTER IS:
```
.  .  .  .  .  .  .  .  .
.                       .
.     *  *  *  *  *      .
.     *           *      .
.     *           *      .
.     *           *      .
.     *           *      .
.     *  *  *  *  *      .
.                       .
.                       .
.                       .
.  .  .  .  .  .  .  .  .
```

CHARACTER OK (Y/N)? Y

ENTER CODE OR COMMAND? Q

A>

In this example we are working on a font named PFONT. CHARGEN opens four files. The first three are hex files (PFONT1.HEX, PFONT2.HEX and PFONT3.HEX) corresponding to the character generator ROM's U49, U48 and U47. The fourth file PFONT.CHR can be used to dump a printout of the font.

The "Y" response to "NEW FILE (Y/N)?" causes the hex files to be initialized to all blank characters. After the first session with a new font you should always answer "N" to this prompt to prevent destroying the entries of previous sessions.

The characters may be entered in any order. When prompted by "ENTER CODE OR COMMAND?" type in exactly eight binary digits representing the code for the character you desire to enter. In response to the line prompt, type a double quote followed by seven or fewer spaces and *'s as appropriate for that character. After the tenth line the code and dot matrix will be echoed as a check for proper entry. If you respond "Y" to the "CHARACTER OK (Y/N)?" prompt, the character will be entered into the hex files. Any other response will require that the dot matrix be re-entered.

If you wish to examine a previously entered character, type "E" before the eight bit code. If you wish to change this character, respond "N" to the OK prompt.

There are three valid commands which may be given the program. The command "Q" terminates a session by simply closing all files. The command "DUMP" causes the complete 256 character font to be printed on the terminal and written into the type CHR file. This may take several hours if the terminal speed is low. The command "CKSUM" causes the check sums for all hex files to be updated. These computations require several minutes. The "CKSUM" command is only required when the entry of characters is completed and you are ready to use the hex files for burning PROM's.

As can be noted in the example, CHARGEN will respond with error messages when invalid entries are made. Attempting to operate on uninitiated hex files will cause program termination without any error message. The message "HEX ERROR" indicates a program malfunction and should not be encountered by the user.

The "CKSUM" and "DUMP" commands should not be used unless the user allows them to run to completion. Attempts to abort these commands may cause the file contents to be lost.

The program listing follows:

```
REM
REM        PROGRAM CHARGEN
REM
REM        THIS PROGRAM INTERACTIVELY GENERATES
REM        HEX FILES FOR THE CHARACTER GENERATOR
REM        ROMS FOR THE IMSAI VIO VIDEO BOARD
REM
REM
REM        COPYRIGHT 1977 IMSAI MANAFACTURING CORP.
REM
REM        VERS 1.0 GEE


                        .


IF END #1 THEN 9999
IF END #2 THEN 9999
IF END #3 THEN 9999

        HEXCONV$="0123456789ABCDEF"
        DIM LN$(10)
        DIM STAR$(16)
        STAR$(0)="****"
        STAR$(1)=" ***"
        STAR$(2)="* **"
        STAR$(3)="  **"
        STAR$(4)="** *"
        STAR$(5)=" * *"
        STAR$(6)="*  *"
        STAR$(7)="   *"
        STAR$(8)="*** "
        STAR$(9)=" ** "
        STAR$(10)="* * "
        STAR$(11)="  * "
        STAR$(12)="**  "
        STAR$(13)=" *  "
        STAR$(14)="*   "
        STAR$(15)="    "

        INPUT "NAME OF CHARACTER FILE";SFILE$
        DFILE$=LEFT$(SFILE$,1)
        IF LEN(SFILE$)=1 THEN 20
        FOR I=2 TO LEN(SFILE$)
        F$=MID$(SFILE$,I,1)
        IF F$="." THEN 20
        DFILE$=DFILE$+F$
        IF LEN(DFILE$)>6 THEN DFILE$=LEFT$(DFILE$,6)
        NEXT I

20
        DFILE1$=DFILE$+"1.HEX"
```

```
          DFILE2$=DFILE$+"2.HEX"
          DFILE3$=DFILE$+"3.HEX"
          DFILE4$=DFILE$+".CHR"


          FILE DFILE1$(50),DFILE2$(50),DFILE3$(50),DFILE4$

          INPUT "NEW FILE (Y/N)";A$
          IF LEFT$(A$,1)<>"Y" THEN 100

          FOR I=1 TO 64
          N=I-1
          GOSUB 200
          LINE$=":100"+HEX$+"000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF"
          PRINT #1,I;LINE$
          PRINT #2,I;LINE$
          PRINT #3,I;LINE$
          NEXT I
          LINE$=":00000000000"
          PRINT #1,65;LINE$
          PRINT #2,65;LINE$
          PRINT #3,65;LINE$
100
          PRINT:PRINT:PRINT
          INPUT "ENTER CODE";A$
          DFLAG=0
          IF A$<>"DUMP" THEN 101
          DFLAG=1
          GOTO 700
101
          IF LEFT$(A$,1)="Q" THEN 9999
          IF A$="CKSUM" THEN 999
          EFLAG=0
          IF LEFT$(A$,1)<>"E" THEN 102
          EFLAG=1
          A$=RIGHT$(A$,LEN(A$)-1)
102
          IF LEN(A$)=8 THEN 110
          PRINT "ENTER 8 BINARY DIGITS"
          GOTO 100
110
          V=0
          F=128
          FOR I=1 TO 8
          AA$=MID$(A$,I,1)
          IF (AA$="1") OR (AA$="0") THEN 115
          PRINT "ILLEGAL CHARACTER CODE"
          GOTO 100
115       IF AA$="1" THEN V=V+F
          F=F/2
```

```
        NEXT I

        IF EFLAG=1 THEN 800
120
        PRINT "INPUT CHARACTER MATRIX"
        PRINT
        EFLAG=0
        FOR J=1 TO 10
125
        PRINT "LINE ";J,".";
        INPUT LN$(J)
        IF LEN(LN$(J))<8 THEN 130
        PRINT "LINE TOO LONG"
        GOTO 125
130
        IF LEN(LN$(J))=7 THEN 135
        LN$(J)=LN$(J)+" "
        GOTO 130
135
        FOR K=1 TO 7
        LNN$=MID$(LN$(J),K,1)
        IF (LNN$=" ") OR (LNN$="*") THEN 140
        PRINT "ILLEGAL CHARACTER"
        GOTO 125
140
        NEXT K
        NEXT J
142
        PRINT:PRINT:PRINT "CODE IS ";A$
        PRINT:PRINT "CHARACTER IS:":PRINT ".  .  .  .  .  .  .  .  ."
        IF DFLAG<>1 THEN 104
        PRINT #4; "  "
        PRINT #4; "  "
        PRINT #4;"CHARACTER CODE:",A$
        PRINT #4;"  "
        PRINT #4;".  .  .  .  .  .  .  ."
104
        FOR J=1 TO 10
        LL$=".  "
        FOR K=1 TO 7
        LL$=LL$+MID$(LN$(J),K,1)+"  "
        NEXT K
        LL$=LL$+"."
        PRINT LL$
        IF DFLAG=1 THEN PRINT #4;LL$
        NEXT J
        PRINT ".  .  .  .  .  .  .  ."
        IF DFLAG<>1 THEN 106
        PRINT #4;".  .  .  .  .  .  .  ."
        RETURN
```

```
106
        PRINT:PRINT:INPUT "CHARACTER OK (Y/N)";B$
        IF (LEFT$(B$,1)="Y") AND (EFLAG=1) THEN 100
        IF LEFT$(B$,1)="Y" THEN 145
        PRINT "REENTER CHARACTER"
        GOTO 120
145
        FOR J=1 TO 10
        JJ=J-1
        N=0
        F=1
        FOR K=1 TO 7
        IF MID$(LN$(J),K,1)=" " THEN N=N+F
        F=F*2
        NEXT K
        GOSUB 200

        IF V>=128 THEN VV=V-128 ELSE VV=V
        IF JJ>7 THEN 150
        PPOSIT=JJ*128+VV
        LOWADR=INT(PPOSIT/16)+1
        IF V>=128 THEN READ #2,LOWADR;LINE$ ELSE READ #1,LOWADR;LINE$

        POSIT=(PPOSIT-16*LOWADR+16)*2+9
        LINE$=LEFT$(LINE$,POSIT)+HEX$+RIGHT$(LINE$,41-POSIT)
        IF V>=128 THEN PRINT #2,LOWADR;LINE$ ELSE PRINT #1,LOWADR;LINE$
        GOTO 157
150
        IF V>=128 THEN VVV=1 ELSE VVV=0
        PPOSIT=(JJ-8)*128+VVV*512+VV

        LOWADR=INT(PPOSIT/16)+1
        READ #3,LOWADR;LINE$
        POSIT=(PPOSIT-16*LOWADR+16)*2+9
        LINE$=LEFT$(LINE$,POSIT)+HEX$+RIGHT$(LINE$,41-POSIT)
        PRINT #3,LOWADR;LINE$
157
        NEXT J
        PRINT:PRINT
        GOTO 100
800
        FOR J=1 TO 10
        JJ=J-1
        IF V>=128 THEN VV=V-128 ELSE VV=V
        IF JJ>7 THEN 820
        PPOSIT=JJ*128+VV
        LOWADR=INT(PPOSIT/16)+1
        IF V>=128 THEN READ #2,LOWADR;LINE$ ELSE READ #1,LOWADR;LINE$
        GOTO 830
820
```

```
          IF V>=128 THEN VVV=1 ELSE VVV=0
          PPOSIT=(JJ-8)*128+VVV*512+VV
          LOWADR=INT(PPOSIT/16)+1
          READ #3,LOWADR;LINE$
830
          POSIT=(PPOSIT-16*LOWADR+16)*2+10
          CK$=MID$(LINE$,POSIT,1)
          GOSUB 500
          LN$(J)=LEFT$(STAR$(M),3)
          CK$=MID$(LINE$,POSIT+1,1)
          GOSUB 500
          LN$(J)=STAR$(M)+LN$(J)
          NEXT J
          GOTO 142


200       HEX16=INT(N/16)
          IF HEX16=16 THEN 250
          HEX=INT(N-HEX16*16)
          H16$=MID$(HEXCONV$,HEX16+1,1)
          H$=MID$(HEXCONV$,HEX+1,1)
          HEX$=H16$+H$
          RETURN
250
          HEX$="00"
          RETURN


400
          N=0
          L=2
410
          CK$=MID$(LINE$,L,1)
          GOSUB 500
420
          N=N+16*M
          L=L+1
          CK$=MID$(LINE$,L,1)
          GOSUB 500
430
          N=N+M
          IF N>=256 THEN N=N-256
          IF N<0 THEN N=0
          L=L+1
          IF L<42  THEN 410
          IF N>=256 THEN N=N-256
          N=ABS(256-N)
          GOSUB 200
          LINE$=LEFT$(LINE$,41)+HEX$
```

```
        RETURN

500     M=0
        IF CK$="0" THEN RETURN
        M=M+1
        IF CK$="1" THEN RETURN
        M=M+1
        IF CK$="2" THEN RETURN
        M=M+1
        IF CK$="3" THEN RETURN
        M=M+1
        IF CK$="4" THEN RETURN
        M=M+1
        IF CK$="5" THEN RETURN
        M=M+1
        IF CK$="6" THEN RETURN
        M=M+1
        IF CK$="7" THEN RETURN
        M=M+1
        IF CK$="8" THEN RETURN
        M=M+1
        IF CK$="9" THEN RETURN
        M=M+1
        IF CK$="A" THEN RETURN
        M=M+1
        IF CK$="B" THEN RETURN
        M=M+1
        IF CK$="C" THEN RETURN
        M=M+1
        IF CK$="D" THEN RETURN
        M=M+1
        IF CK$="E" THEN RETURN
        M=M+1
        IF CK$="F" THEN RETURN
        PRINT "HEX ERROR"
        GOTO 9999

999
        FOR I=1 TO 64
        READ #1,I;LINE$
        GOSUB 400
        PRINT #1,I;LINE$
        READ #2,I;LINE$
        GOSUB 400
        PRINT #2,I;LINE$
        READ #3,I;LINE$
        GOSUB 400
        PRINT #3,I;LINE$
        NEXT I
```

```
        GOTO 9999

700
        P$="0"
        FOR P=0 TO 1
        Q$="0"
        FOR Q=0 TO 1
        R$="0"
        FOR R=0 TO 1
        S$="0"
        FOR S=0 TO 1
        T$="0"
        FOR T=0 TO 1
        U$="0"
        FOR U=0 TO 1
        W$="0"
        FOR W=0 TO 1
        X$="0"
        FOR X=0 TO 1
        A$=P$+Q$+R$+S$+T$+U$+W$+X$
        V=X+2*W+4*U+8*T+16*S+32*R+64*Q+128*P
        GOSUB 800
        X$="1"
        NEXT X
        W$="1"
        NEXT W
        U$="1"
        NEXT U
        T$="1"
        NEXT T
        S$="1"
        NEXT S
        R$="1"
        NEXT R
        Q$="1"
        NEXT Q
        P$="1"
        NEXT P
9999 END
```
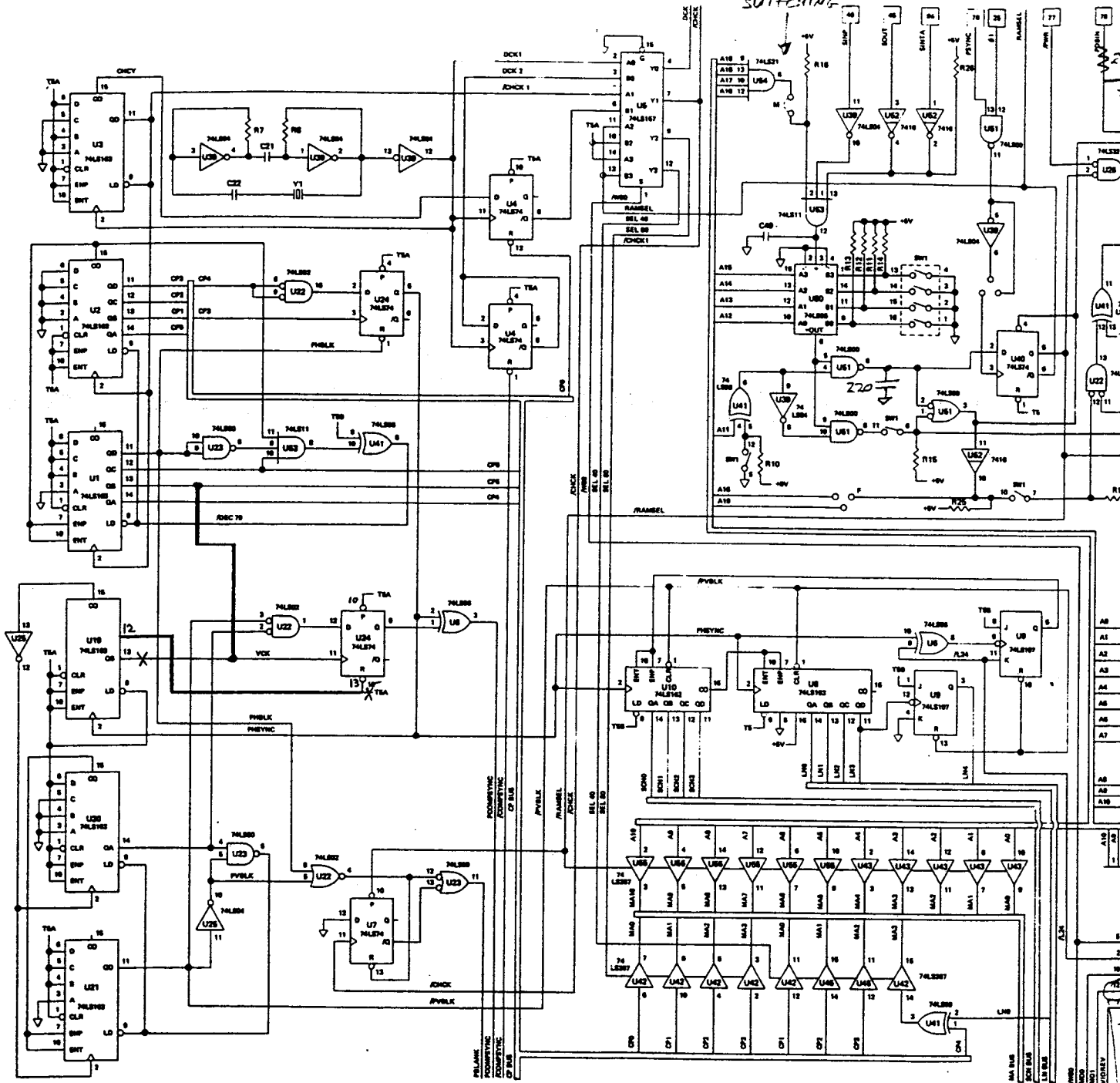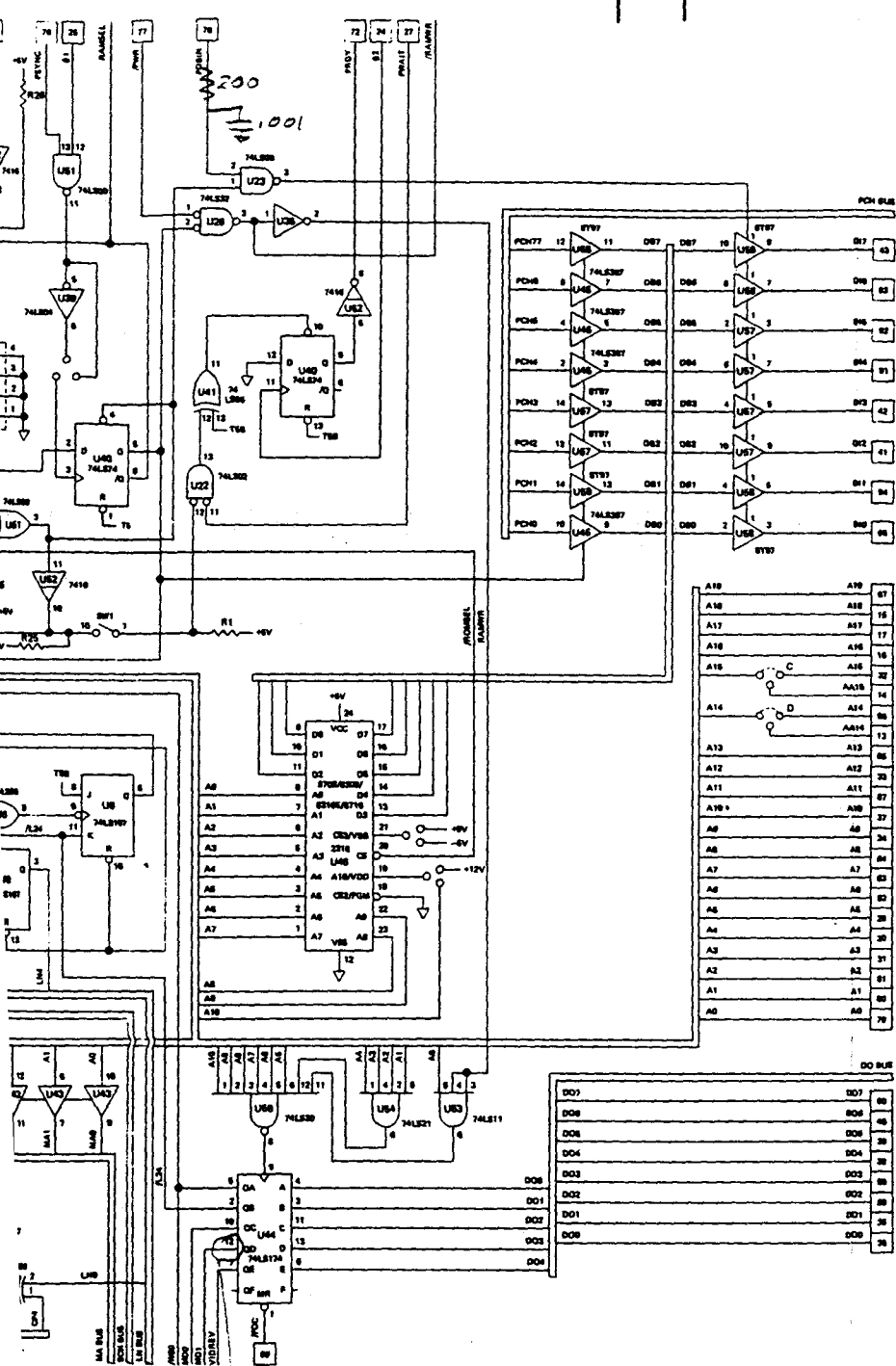
LIFT PIN 12
FOR 08 DEFAULT

TOLERANCES UNLESS OTHERWISE SPECIFIED			
FRACTIONS	DEC.	ANGLES	
±	±	±	

APPROVALS	DATE
DRAWN	
CHECKED	

IMSAI SYSTEM
VIO REV. 1
SCHEMATIC DIAGRAM

SCALE		SIZE	DRAWING NO.
		B	
DO NOT SCALE DRAWING		SHEET	1 OF 2

U1 THRU U3	74LS163	U62	74LS166	
U8		C1 THRU C20		
U20		C23 THRU C26		
U21		C35	.1 uF	
U4 U7 U24 U40	74LS74	C37 THRU C48		
U5	74LS157			
U6 U41	74LS86	C21 C49 C36	.001 uF	
U9	74LS107	C22	20 pF	
U10 U19	74LS162	C27 THRU C34	2.2 uF 25V	
U11 THRU U18	2102 AL-4	C50	470 pF	
U28 THRU U35		Q1	2N3904	
U22	74LS02	R1 R10 THRU R16	2.2K ¼W	
U23 U51	74LS00	R25 R26		
U25 U39	74LS04	R2 R9 R21	1K ¼W	
U26	74LS32			
U27	3624 W/ VIO MAP	R3 R7	470 ¼W	
U36 U37	SPARE	R8		
U42 U43 U45 U55	74LS367	R4	100 ¼W	
		R5 R6	150 ¼W	
U44 U59 U60	74LS174	R17 THRU R20	4.7K ¼W	
U46	2316 SMP/80.0	R22 THRU R24		
U47 THRU U49	8308/8708	S1	8 POSITION DIP	
U50	74LS85			
U52	7416	Y1	XTAL 12.2304 MHz	
U53	74LS11			
U54	74LS21			
U56	74LS30			
U57 U58	8T97			
U61	74LS153			

TOLERANCES UNLESS OTHERWISE SPECIFIED
FRACTIONS DEC. ANGLES
± ± ±

APPROVALS	DATE
DRAWN	
CHECKED	

IMSAI SYSTEM
VIO REV. 1
SCHEMATIC DIAGRAM

SCALE	SIZE	DRAWING NO.
	B	

DO NOT SCALE DRAWING | SHEET 2 OF 2