(2) - 1 - 2

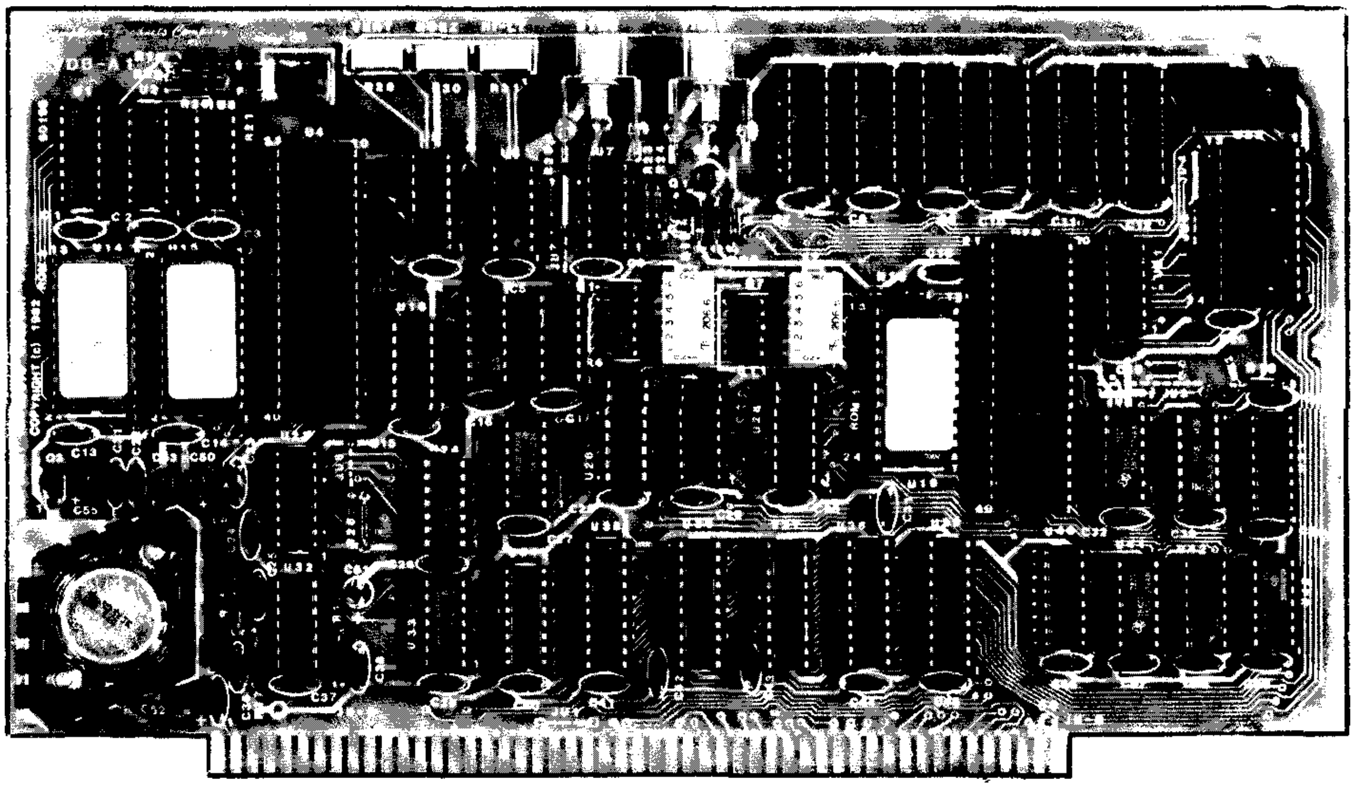
LS32 —
LS125 —
LS-74 —
S00
LS74
LS29

Simpliway Products Company

P.O. BOX 601
HOFFMAN ESTATES, IL 60195

VDB-A

S-100 VIDEO TERMINAL BOARD



The VDB-A is a video terminal interface for S-100 based computer systems. It provides a totally programable C.R.T. display, including: an 80 character by 24 line display, blinking cursor, highlighted/blinking text formatting, etc. The Intel 8275 C.R.T. Controller is fully implemented, including the light pen feature (via software linkages in the program eprom). A complete parallel-input keyboard port is provided, and a portion of the on-board '2114 static ram is used to allow the user to type commands ahead to the host computer while it is executing other code. The on-board Z80 CPU and 2 kbyte program eprom provide lightning fast keyboard and video operations.

A second '2716 eprom provides the full ASCII character set with lower case descenders, while an optional third eprom may be used for an alternate character set or custom graphics. Pre-programmed eproms can be purchased separately. The on-board ram (3 kbytes) is used for the keyboard buffer, video and attribute data, and scratch pad/stack for the CPU.

The board may be configured as either I/O port, or memory mapped, and may be changed from one to the other under keyboard or software coltrol. In the I/O mode, the on-board ram is not addressable by the host CPU (ie: does not occupy system memory map), and all video and screen attribute data flow through the I/O ports. In the memory mapped mode, the I/O port is still fully active; the screen portion of the ram may be written to by either the host CPU, or through the I/O port, although the video data ram may not be directly read by the host CPU.

Thus, the on-board ram occupies a portion of the system memory map, but only as "write-only" memory. It is required however, that the memory in the host system be tolerant of wait states, as some operations will be delayed by screen refresh periods. The VDB-A does not use direct memory access (DMA), and thus compatibly resides in most S-100 systems. This eliminates such things as bus conflicts during disk operations, etc. The design presumes a 2 MHz system clock, although 4 MHz is permitted with the use of faster rams and eproms, or the use of an on-board divide-by-two circuit.

The software provided features complete cursor control and addressing, up to 16 attribute changes per line (ie: blinking, reverse video, highlight, underline, etc.). The software emulates the Micro-Term (TM) ACT-4 terminal, but can be easily modified for others. While the IEEE-696 bus definitions are observed, the higher address pages are not decoded, and systems of more than 65 Kbytes could experience difficulties.

The PC board is of high quality, with plated thru holes, solder plated runners, solder masks on both sides, a component side legend, and gold plating on the edge connector.

OTHER FEATURES include:

-- Bell (beeper) driver for direct connection to a speaker

-- Separated video and sync, or composite video

-- I/O port and screen addressing individually switch selectable

-- Choice of positive or negative keyboard strobe

-- Keyboard port socket (16 pin dip) also provides +5 v. and -12 v. to power keyboard

-- All static ram ('2114's , 300 ns.)

-- Documentation includes : 49 pages of schematics and parts list, theory of operation, construction hints, a well annotated source listing of the "ROM 1" video driver software, the bit pattern listing for the "ROM 2" character generator eprom, and trouble shooting chart.

-- Easily configured to allow emulation of most terminals by modification of video driver software.

-- NEW OPTIONAL 25 line non-scrolling for STATUS DISPLAY (Note: deletes underline attribute)

-- Optional interupts output, "AND MUCH MORE".

Please note:

The VDB-A is a somewhat complex product, and its construction should not be attempted by those without some previous experience in assembling and testing circuits of similar complexity. Should you feel un-sure, please order the documentation package first.

VDB – A VIDEO TERMINAL

FOR S-100 SYSTEMS

USER'S MANUAL

SIMPLIWAY PRODUCTS COMPANY
P.O. Box 601
Hoffman Estates, Il. 60195
(312) 359-7337

The *Simpliway Products Co. (SPC)* reserves the right to make changes to any products described herein to improve the function, design, or reliability. The information contained in this document has been carefully reviewed, and is believed to be accurate. However, *SPC* does not assume any liability arising out of the construction, use, or application of any product or circuit described herein, nor does *SPC* convey any license under its patent or copyrights, or the rights of others.

RELEASE A-2 09-30-83

TABLE OF CONTENTS

Please note:

The VDB-A is a somewhat complex product, and its construction should not be attempted by those without some previous experience in assembling and testing circuits of similar complexity. Should you feel it is too difficult a project, please feel free to return the complete un-assembled product for a cash refund, after obtaining prior permission from SPC.

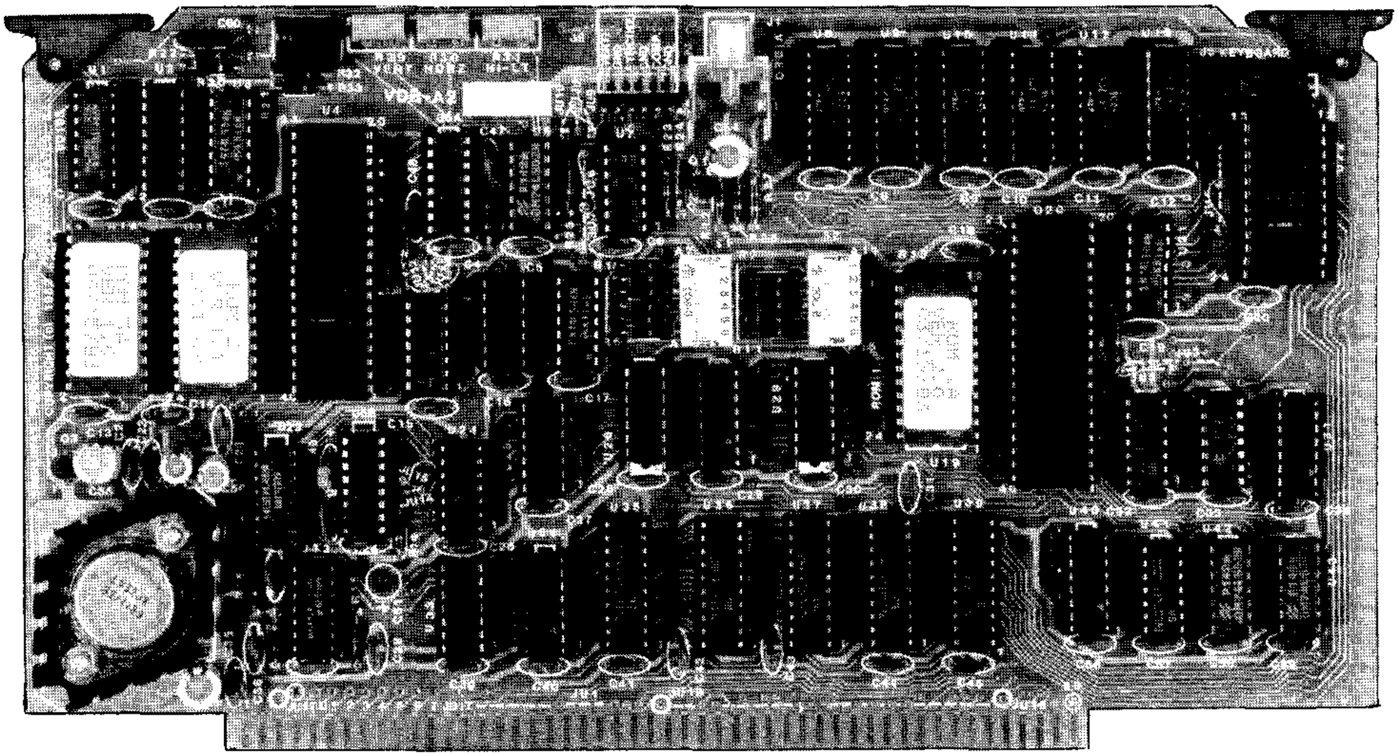
*** SPECIAL NOTE ---- Monitor Requirements ***

Like many other high performance video display boards, the VDB-A Video Board must be used with a monitor that has a high video bandwidth (at least 12 MHz), and is capable of operating at a 19 kHz horizontal sync rate. If the monitor to be used cannot be adjusted externally for this higher sync rate, then internal modifications (usually quite simple) will be necessary. Please refer to the note on page 12, and the Trouble-shooting chart for more information. Alternatively, the 25 line software option may be ordered.

INTRODUCTION and SPECIFICATIONS

V D B – A

S–100 VIDEO TERMINAL BOARD



THE MANY FEATURES of the VDB–A include:

-- WORDSTAR & dBASE II compatability for highlighted display *

-- 25th non-scrolling status line - TIME STAMP or SOFT KEYS *
Note: Deletes underline attribute; allows 15,750 kHz sync

-- All 8275 attributes available: BLINK, HIGHLIGHT, REVERSE VIDEO, and UNDERLINE (rel's 1.x & 3.x only)

-- Bell (beeper) driver for direct connection to a speaker

-- TYPE-AHEAD keyboard buffer holds 80 characters

-- Alternate character eprom socket, software controlled

-- Separated video and sync signals, or composite video

-- Memory mapped screen addressing is switch selectable to any 2K boundry, and enabled under host system control

-- I/O port addressing switch selectable to any group of 4

-- Choice of positive or negative keyboard strobe

-- Keyboard socket provides +5 and -12 v. to power keyboard

-- All static ram ('2114's , 300 ns.)

WORDSTAR is a trademark of Micropro International Corp.
dBASE II is a trademark of Ashton-Tate Corp.
VDB-A and VIDSTAR are trademarks of The Simpliway Products Co.

-- Easily reconfigured to allow emulation of most terminals by modification of video software: Televideo 920 is standard.

-- Optional interupts output, "AND MUCH MORE".

* Requires optional software --- See ordering info page

The VDB-A is a video terminal interface for S-100 based computer systems. It provides a totally programable C.R.T. display, including: an 80 character by 24 line display, blinking cursor, highlighted/blinking text formatting, etc. The Intel 8275 C.R.T. Controller is fully implemented, including the light pen feature (via software linkages in the program eprom). A complete parallel input keyboard port is provided, and a portion of the on-board '2114 static ram is used to allow the user to type commands ahead to the host computer while it is executing other code. The on-board Z80 CPU and 2 kbyte program eprom provide lightning fast keyboard and video operations (over 19,000 baud!).

A second '2716 eprom provides the full ASCII character set with lower case descenders, while an optional third eprom may be used for an alternate character set or custom graphics. Preprogrammed eproms can be purchased separately. The on-board ram (3 kbytes) is used for the keyboard buffer, video and attribute data, and scratch-pad / stack for the CPU. The board may be configured as either an I/O port or memory mapped, and may be changed from one to the other under keyboard or software con trol. In the I/O mode, the on-board ram is not addressable by the host CPU (ie; does not occupy system memory map), and all video and screen attribute data flow through the I/O ports. In the memory mapped mode, the I/O port is still fully active; the screen portion of the ram may be written to by the host CPU directly or through the I/O port. Although direct reading of the video ram by the host CPU is not supported, it is possible to do.

Thus, the on-board ram occupies a portion of the system memory map, but only as "write-only" memory. It is required however, that the memory in the host system be tolerant of wait states, as some operations will be delayed by screen refresh periods. The VDB-A does not use direct memory access (DMA), and thus compatibly resides in most S-100 systems. This eliminates such things as bus conflicts during disk operations, etc. The design presumes a 2 MHz system clock, although 4 MHz is permitted with the use of faster rams and eproms. The software provided features complete cursor control and addressing, and up to 16 non-displayed attribute changes per line (ie: reverse video, bilnking, highlight, underline, etc.) in any combination. The software emulates the Micro-term (tm) ACT-4 (VDB-A software version 1.x) or the TELEVIDEO 910/920 (versions 3.x & 4.x) ter minals, but can be easily modified for others. While the IEEE 696 bus definitions are observed, the higher address pages are not decoded, and systems of more than 65 Kbytes could experience difficulties, unless certian system software precautions are observed, and then only when in the memory-mapped mode.

The FR4 PC board is of high quality, with plated thru holes, solder plated runners, solder masks on both sides, a component side legend, and gold plating on the edge connector.

ASSEMBLY INSTRUCTIONS

Start by obtaining a small needle nose pliers and wire cutter, a screwdriver, a flat work surface, and a strong light. A rosen core solder is mandatory for printed circuit work, and the thinner .040 diameter type is preferred. Use a 25-40 watt soldering iron with a small chisel or pencil tip. Apply the iron to the circuit pad to be soldered, apply the solder to the pad, touching the iron as well, and add enough solder to wet the tip and pad. Hold the iron on long enough to cause the solder to bubble down into the plated thru hole. If not enough heat is used, flux may form around the component lead and leave an intermittant or cold solder joint. If too much heat is used, the foil pad will lift off the board, and the component may be damaged as well.

The general procedure is to start with those components that have the lowest profile (height), then add the next lowest parts, etc., until all the components are soldered. The component side of the board has the component legend. Insert the leads of the component into the correct holes as shown by the legend (and printed component placement guide), seat the part as close to the board as possible, and bend the leads over at about a 45 degree angle. Then solder as described above and clip the leads as close as possible to the board. The recommended order of assembly is as follows:

[] First, examine the board for visible shorts. If possible, use an ohmmeter to check for shorts between address lines, data lines, and between +5 v. and ground (pins 1 & 50) on the edge connector.

[] Insert all 1/4 watt resistors:

[✓]	R1-R14	10.0 kohm	(brown-black-orange)	[✓]	R20	220 ohm	(red-red-brown)
[✓]	R15	220 kohm	(red-red-yellow)	[✓]	R16	68 kohm	(blue-gray-orange)
[✓]	R17	6.8 kohm	(blue-gray-red)	[✓]	R18	4.7 kohm	(yellow-violet-red)
[✓]	R21	560 ohm	(green-blue-brown)	[✓]	R22	1.5 kohm	(brown-green-red)
[✓]	R23,24	470 ohm	(yellow-violet-brown)	[✓]	R25,19	330 ohm	(orange-orange-brown)
[✓]	R26	75 ohm	(violet-green-black)	[✓]	R27	150 ohm	(brown-green-brown)
[✓]	R28,32,33	2.2 kohm	(red-red-red)				

[] Turn the board over, lay it on a flat surface and solder all resistors.

[] Insert all IC sockets, noting the proper position of pin 1 on each (usually a notch in the plastic). Sockets should not be used for switches S1 and S2. Place a heavy cardboard (the back of a clip-board does nicely) over the sockets, and carefully turn the sandwich over to expose the solder tails of the sockets. Solder only adjacent pins of each socket (ie: pins 1 & 8 on a 14 pin socket). Now the board can be lifted without the sockets falling out. Re-heat each joint, forcing the socket down against the board to seat it properly. All this is to insure that the sockets won't tear foil loose when the IC's are inserted. After this, the rest of the pins on the sockets can be soldered.

[] Insert and solder the 79L12 regulator, Q3. Observe the marking on the board. Note: if -12 volts is not required on the keyboard socket, this part may be deleted.

[] Insert and solder all disk ceramic capacitors:

[✓]	C1-C45, C56-59	0.01 uF, 25 v.	[✓]	C46,49	.047 uF, 25 v.
[✓]	C47	.001 uF, 25 v.	[]	C48	470 pF, 25 v.
[]	C60	33 pF, silver-mica			

[] Insert and solder the tantalum and electrolytic capacitors, noting the polarity sign (+) marked on the board:

[]	C50,52-55	15 uF, 20 v.	[]	C51	1.0 uF, 20 v.

[] Insert and solder the potentiometers:

[]	R29,30	20 kohm
[]	R31	5 kohm

[] Insert and solder Q1, the 2N3904 transistor, being sure to position the emitter (E), base, and collector (C) leads correctly.

[] Mount the phono connector and solder. Also, assemble the coax cable to the plug (optional).

[] Mount Q2, the LM323, 5 volt regulator and its heatsink. The two screws used for the flange also mount the heatsink as well. Do not use an insulator under the device, but a very thin coating of heat conducting grease (Wakefield) between the device flange and the heatsink is quite desirable. The heads of the screws should be located on the solder side of the board, and care should be taken to not over-tighten them. The leads of Q2 may now be soldered.

[] Bend the leads of the crystal, taking care not to stress the points where the leads enter the bottom of the crystal case. Insert and solder. A small piece of double-sided foam tape makes an excellent retainer and shock mount.

[] Insert and solder the jumper options as described below:

- [] JU1 For non-maskable interupts to host CPU
- [] JU2 For 2 MHz system clock (standard S-100)
- [] JU3 For 4 MHz clock (non-standard & seldom used)
- [] JU4 If keyboard strobe is positive
- [] JU5 If keyboard strobe is negative
- [] JU6,9,13,15 For composite video output to connector J1
- [] JU7,9,13,15 For video output to J1 and sync output to J2
- [] JU7,10,12,15 Seperated video, horiz, & vert to J2 ***
- [] JU8 To NOT invert the signal from the light pen
- [] JU16 To use "preset" instead of "POC". Also cut the runner at pin 99 of the S100 edge connector.
- [] JU17 To use Vectored Interupts, VI 0 thru 7

[] Insert and solder switches S1 and S2.

[] Perform a thorough visual check of the board under a strong light to locate and clear any solder bridges between adjacent pads.

[] Insert board into host system and apply power. Observe that +5 volts appears on each IC socket pin as described in the parts listing. Also check that +5 volts appears on pin 16 of the keyboard socket, and that there is -12 volts on pin 14, if Q3 is installed. Note that this pin can only source about 100 mA to the keyboard.

[] Turn off the power, remove the board from the system, and with the board lying on a conductive, grounded surface, insert all IC's into their sockets, using the component placement guide and parts list. Take extreme care to insert each IC in the correct direction, and to observe that none of the leads of the IC's fold up under the body (a very common occurance, and very hard to see!).

***** SPECIAL NOTES *****

The Z80 CPU, 8275 CRT controller, the 2114 rams, and the 2716 eproms are STATIC SENSITIVE devices. Static preventive measures MUST be employed.

[] By this time, you should have prepared (or purchased) the program ROM (1) and the character generator ROM (2). Insert these as well.

[] Replace the board into the system, and connect the monitor RF coax to J1 (or J2 if using separate sync).

[] Apply power, and using a thumb or forefinger, quickly check the body of each part for overheating. Only the regulator Q2 should get noticeably warm after a minute or so. If any other part gets very warm, the part may be defective or inserted wrong. Turn the power off immediately and correct this!

[] Adjust the 20 kohm pots (R29 and R30) to obtain a raster on the CRT. R30 controls the horizontal sync and position, and R29 controls the vertical sync. When adjusted properly, the cursor should be observed flashing in the upper left-hand corner of the screen.

[] Remove the power. If the on-board keyboard port is not to be used, skip the rest of this section.

Connect the keyboard to the socket, using a 16 pin dip plug and cable wired as follows:

Pins 1 thru 7 of this socket correspond to the ASCII parallel data bits 0 - 6 respectively. Wire these accordingly. Connect the keyboard strobe line to pin 9 of the socket, and use both pins 11 & 12 for the ground (common) return. Power for the keyboard is available on pin 16 (+5 v) and pin 14 (-12 v), when installed.

[] A speaker (8 ohms or greater) may be connected to pin 15 of the keyboard socket. A 1000 ohm volume control, mounted on the keyboard, and wired in series with the speaker, is often desirable.

[] Switch block S1 sets the I/O port address of the board. Position 1 represents the highest order bit (A7) of the address; position 2 - A6; etc. The software supplied assumes the board is addressed at ports F0 to F3 (although F3 is not used), and thus the switch would be set as follows:

```
 A7   A6   A5   A4   A3   A2
  |    |    |    |    |    |          ON = 0
  1    2    3    4    5    6
  |    |    |    |    |    |         OFF = 1
 OFF  OFF  OFF  OFF  ON   ON
```

If this port assignment conflicts with the host system assignments, the polling program (described below) and the port equates in the CRTLOOP listing (or host system driver patch) must be redefined. Also, the S1 switch block must be set accordingly.

[] Switch block S2 sets the memory mapped address of the screen ram (2 Kbytes) at any 2K boundry. Thus, if the screen is to be addressed at C000 to DFFF, the switch settings would be:

```
 A15    A14    A13    A12    A11
  |      |      |      |      |
  1      2      3      4      5      6          X = Not Used
  |      |      |      |      |      |
 OFF    OFF     ON     ON     ON     X
```

[] With the board properly address, and with all required IC's and jumpers installed, replace the board into the system (power off, of course), install the keyboard plug and the CRT cable plug(s) as described above.

[] Apply the power again, and with the cursor flashing, enter by some means a short program that will read the keyboard and display the keyboard data. Refer to the sample program furnished, CRTLOOP, for a basic example. In this program, both the on-board keyboard port and a system keyboard port are polled. In say, a CP/M system, a routine similar to this would be part of the CBIOS.
*Trademark of Digital Research Corp.

[] Execute the small program previously loaded into system ram, and observe the CRT's response to the keyboard as you press each key. The system should be acting as a video typewriter, with only the I/O ports active.

[] Next, fry the escape and control codes. The cursor should be moveable, and the flashing, underlining, etc., should be operable. Refer to the QUICK REFERENCE CHART for the definitions of each code, as defined by the ROM 1 software supplied. NOTE: Except for the memory map commands, all escape and control codes are toggle entries, ie: on & off, and effect only keyboard entries following each command.

[] Adjust the contrast and brightness controls of your monitor device for the best overall character definition. Remember, a converted TV set will not provide the high resolution of a wide bandwidth video monitor unit. Again, only characters typed after invoking a control or escape code will be displayed in that fashion.

[] With the highlight mode enabled (^N), type a string of characters. Adjust R31 for the desired brightness, relative to characters typed with the highlighting disabled. Alternate control-N's (^N) will enable, then disable, the highlighting function for succeeding keystrokes. NOTE: the ROM 1 software initializes the board to the 'bright' condition. (Note: Release 1.x software is presumed in this text)

[] Now enter an esc-M from the keyboard. This enables the memory mapped mode. Until an esc-O is entered, the on-board ram occupies a part of the system memory map, at the address assigned by switch block S2.

[] With the speaker connected as described previously, activating a control-G (^G) should produce a pleasent tone. A ^L (formfeed) will clear the screen.

The VDB-A should be working properly when all these checks have been successfully completed. A trouble-*shooting chart is included in this document to assist in locating some hardware faults. We recognize* that it may be difficult to diagnose faults on a product of this type, since the program cannot be modified directly by the host system to assist in isolating problem areas.

If a problem occurs that seems to defy easy solution, PLEASE, PLEASE call or write us. If you find a solution to a problem, again, let us know about it, as we may be able to pass it along to others similarly troubled.

Theory of Operation for the Video Display Board

General Description

The Video Display Board (VDB-A) is a self contained computer terminal, less the monitor and keyboard. The display is comprised of 24 lines of 80 columns, and uses the Intel CRT Controller 8275 and a Z80 microprocessor to control the functions. The 8275 provides cursor control, reverse video, blinking, underlining, highlighting, and light pen function. The displayed characters are in a 5 X 7 format with upper and lower case. Two character generators are available to the user. The board may be interfaced to the host computer via an I/O port or memory mapped mode. The keyboard interface is 8 bit parallel with strobe and a 80 character type ahead buffer. A speaker oscillator circuit is also provided for the audible "bell" tone.

The VDB-A may be divided into five logic blocks: S-100 interface, Microprocessor and control logic, system memory, video display circuits, and memory map circuit. Each block will be discussed below.

S-100 Interface

This portion of the VDB-A provides the I/O interface to the S-100 bus. The Interface uses three I/O addresses whose upper six bits are determined by the dip switch block (S1) connected to the decoder U-26. The two remaining least significant bits perform the following functions:

00	Display port status (IN), Display data (OUT)
01	Keyboard status (IN).
10	Keyboard data (IN).

For example, if the dip switches were set to 8CH then the address 8DH would read the keyboard status.

The VDB is selected by a correct address presented to U-26 which in turn enables U-34. U-34 is also enabled by SIN or SOUT via a NOR gate. When a display port status is desired pin 15 of U-34 is low and DBIN is high. This allows the port status stored in flip-flop U-24 to be put on the S-100 bus via the tri state gate U-17. The keyboard status is read in the same manner. When the host computer writes to the display, pin 14 of U-34 is low and PWR* is low. This activates the latch U-38 to accept the data on the S-100 bus and hold it for the Z80, and it also sets the status flip-flop U-24. When the host computer reads the keyboard, pin 13 of U-34 is low and DBIN is high. This puts the data stored in latch U-39 on the S-100 bus, and resets the status flip-flop U-24.

The Z80 also writes to U-39, and reads U-38 and the status latches. This is done through the I/O decoder, U-40 and U-33.

Microprocessor and control logic

The Z80 processor controls all of the functions of the VDB-A via the internal address and data buses. Eight I/O ports are addressed via decoder U-40. These functions are:

S-100 interface status (2 ports)
S-100 display data (IN)
S-100 keyboard data (OUT)
Keyboard latch
Speaker oscillator circuit
Memory map ports
8275 CRT Controller
Interrupt flip-flop

There is also special interface circuits to transfer display characters to the 8275. ***When the 8275 wants*** data for the next line to be displayed, pin 5 of the 8275 goes high. This sets the flip-flop U-29, which generates an interupt in the Z80. The Z80 transfers the data to the 8275 by reading the refresh memory. The Z80 tells the 8275 to latch the data by activating pin 5 on the Z80 (A15). This pin is fed to U-43, along with the read signal and the data request from the 8275. The output of U-43 goes low telling the 8275 to latch the data via pin 6 and pin 10, data acknowledge and WR*. When the transfer is finished, the interrupt flip-flop is reset by the Z80.

The keyboard latch is activated by the strobe produced by a parallel keyboard. Once the data is latched, an interrupt is sent to the Z80 processor, which in turn reads the latch.

When the speaker oscillator U-32 is activated by decoder U-40, a 1kHz tone is produced for 500ms.

System Memory

The system memory consists of 2k of ROM and 3k of RAM. The ROM 1 contain the VDB-A control program. The RAM is used for the display refresh memory, keyboard buffer, and control program scatch pad area. All of memory is addressed via the decoder U-31 and the Z80 control lines WR* and MREQ*.

Video Display Circuits

The video display circuits are contolled by the 8275 and the dot clock oscillator. The oscillator is made up of inverters in U-3 and a 13.63 MHz crystal. This output is called the dot clock, and is used to drive the paraellel-to-serial shift register, U-1, and the divide by seven, U-2. The output of U-2 is called the character clock, since its period is the width of one character on the srceen. The character clock drives the logic in the 8275 to produce the video timing signals. At the rising edge of the character clock, the next character to be displayed is presented to the character generator. On the falling edge of the character clock, the output of the character generator is latched into the shift register, U-1, where the data is serially shifted by the dot clock to the monitor. Before the character is presented to the monitor it is combined with other timing signals from the 8275. These are latched into U-23 on the falling edge of the character clock. These signals are:

Reverse video
Video suppression (used during horizontal and vertical retrace,
and blinking
Light enable (used to display the cursor and underline)
Highlight
Horizontal Retrace
Vertical Retrace

The Horizontal and Vertical retrace pulse widths are controlled by U-5A & B to match most monitors. Also, these signals can be separated from the video data.

Memory Map Interface

The memory map interface used with VDB-A does not access the display memory directly from the S-100 bus as in most systems. This is because the memory is not organized in the same manner as most memory mapped video boards. When the host computer wishes to put a chracter on the screen, the address and data are latched. The latched data is then read by the Z80 processor, which then calculates then proper display address and puts the character on the screen. If the host computer tries to output another character before the Z80 processor has read all of the latches, the S-100 ready line is forced low until the last latch is read.

Address decoding is by the comparator, U-28. The switch block (S2) connected to U-28 allows the user to set the memory map in 2 kbyte increments. The decoder is only activated by a memory write cycle. When a proper address is decoded by U-28, U's 36, 35, and 37 are lached. The address and data is latched by PWR*, and the latch status flip-flop U-41 is set. This tells the Z80 processor that there is a new character to be placed, causing it to read the latches, through the decoder, U-27. Also, when latch U-37 is read, the status flip-flop is reset. If the status flip-flop is set and the host computer wishes to place a character, the ready line is set low via the address decoder and U's 42 and 7.

*** SPECIAL NOTE on monitor requirements ***

In order to get 12 scan lines per character row (needed for lower case descenders and underlining), a horizontal sync rate of 18.720 kHz is used, which is much above the 15.750 kHz normally associated with TV systems. The monitor selected to be used with the VDB-A Video Board must be capable of operation at this higher sync rate, and may require an internal modification to the horizontal oscillator (usually quite simple), if the external adjustment on the monitor does not have enough control range. If the sync frequency is incorrect, multiple cursors, spread out characters, etc. will be displayed.

Parts list for Video Display Board Ver 6.1   9-30-83

IC#	DESCRIPTION	STRAPPED PINS	
		5V	GND
30	74LS00	14	7
42	74LS02	14	7
6	74LS02	14	7
25	74LS32	14	7
33	74LS32	14	7
18	74LS04	14	7
21	74LS04	14	7
3	74LS04	14	7
7	74LS05	14	7
16	74LS86	14	7
24	74LS74	14	7,2,3,11,12
29	74LS74	14,4	7
41	74LS74	14,10,4	7
26	8131	16	7,8
28	8131	16	8,1,2
34	74LS138	16,6	8,3
40	74LS138	16,6	8,5
31	74LS138	16,6	8,5
27	74LS138	16,6	8
39	74LS373	20	10
37	74LS373	20	10
35	74LS373	20	10
38	74LS373	20	10
36	74LS373	20	10,8,13,14,17,18
17	74LS125	14	7
32	556	14	7
22	8212	24	12,2
20	Z80	11,24,25	29
4	8275	40	20
8	2114	18	9
9	2114	18	9
10	2114	18	9
11	2114	18	9
12	2114	18	9
13	2114	18	9
19	2716 *	21,24	12,18
15	2716 *	21,24	12,18
14	2716 *(OPTIONAL)	21,24	12,18
1	74LS166	16	8
23	74LS174	16,1	8
2	74LS161	16,1,3,6,7,10	8,4,5
5A	74LS123	16,3,11	8,1,9
5B	74LS123	16,3,11	8,1,9
43	74LS20	14	7

* AVAILABLE PRE-PROGRAMMED FROM THE SIMPLIWAY CO.
NOTE: Use single supply, 5 volt type only.

MISC. COMPONENTS

QTY	DESCRIPTION
1	LM323 5v, 3a TO3 CASE
1	* HEAT SINK THERMALLOY #6013-B
2	40 PIN SOCKET
5	20 PIN SOCKET
4	24 PIN SOCKET
6	18 PIN SOCKET
12	16 PIN SOCKET
16	14 PIN SOCKET
1	3 FT 16 PIN DIP CABLE (FOR KEYBOARD)
49	.01uF CAPS, DISK 25V +80/-20%
5	15uF TANT CAPS 20V +/- 20%
1	1.0uF TANT CAP 20V +/- 10%
1	.001 uF CAP, DISK 25V +/- 10%
2	.047 uF CAP, DISK 25V +/- 10%
1	470 pF CAP, DISK 25V +/- 10%
1	33.0 pF CAP, Silver mica +/- 10% style DM15
1	68K OHMS 1/4W 5%
1	6.8K OHMS 1/4W 5%
1	220K OHMS 1/4W 5%
1	220 OHMS 1/4W 5%
1	150 OHMS 1/4W 5%
1	75 OHMS 1/4W 5%
1	560 OHMS 1/4W 5%
1	1.5K OHMS 1/4W 5%
2	470 OHMS 1/4W 5%
2	330 OHMS 1/4W 5%
14	10K OHMS 1/4W 5%
1	4.7K OHMS 1/4W 5%
3	2.2K OHMS 1/4W 5%
1	* 5000 OHM POT 3/8" SIDE ADJUST, STANDUP, 0.15" SPACING,
2	* 20K OHM POT (BOURNS SERIES 3386S OR EQUIV.)
1	* CRYSTAL: 13.628 MHz (60 Hz USA) ROM-1 ver.: 1x & 3x 11.356 MHz (50 Hz FOREIGN) " " 11.340 MHz (60 Hz USA) 2x & 4x 9.450 MHz (50 Hz FOREIGN) " " Note: Frequency also determined by ROM-1 version used.
1	NPN TRANSISTOR 150 MHZ Ft 2N3904 OR EQUIV.
1	REGULATOR, -12.0 VOLTS, MC79L12 OR EQUIV.
2	6 POSITION DIP SWITCH
1	6 POSITION PIN HEADER Molex #22-05-3061, GC# 41-046 (Opt'l)
1	* P.C. PHONO JACK (OPTIONAL)
2 EA.	6-32 X 3/8" ROUND HEAD MACHINE SCREW, NUT, LOCKWASHER
2	Ejectors Scanbe, Inc. # S-203 or equiv. (Optional)

* AVAILABLE FROM THE SIMPLIWAY CO.

SOFTWARE OVERVIEW

VIDEOxx.PRN ---

This is the complete source listing of the ROM 1 program, with addresses and annotations. The program, as written will emulate a Micro Term model ACT-4 video terminal. However, with the table of escape and control character equates at the beginning, it should be quite simple to modify the listing for any other codes. It is even possible to use any control and/or escape character for the same function. The program occupies about 1100 bytes, and thus there is plenty of room for any embellishments desired. The listing is self-explanatory; Zilog mnemonics are used. xx = version no.

VIDEOxx.Z80 ---

Same as above, but without address listing for use with Z80 assemblers. Provided only on 8 inch CP/M disk.

VIDEOxx.DTA ---

The object code for above, with origin at 00H. Load with DDT, and offset to a clear portion of ram for your prom-burner device.

CHARGEN.PRN ---

This is a HEX listing of each character block in ROM 2. Each 16 consecutive addresses defines one character from top to bottom, though only the first 10 bytes in each block is used, and the first is allways zero. The overall matrix size is 9x7, though 5x7 is used for alpha-numerics.

CHARGEN.DTA ---

Same as above, but in BINARY format to load into system ram at 100H. Supplied on disk only.

CRTLOOP.Z80 ---

This is a demonstration program to exercise the VDB-A I/O ports, and to output video to the monitor device. The program tests both the system keyboard and the VDB-A keyboard port for data, and then outputs the data to the VDB-A, after first checking its status. If a host system keyboard driver program ia to be used, just replace the IN (PORT) instructions with CALL (ADDR.) to those routines.

```
0000
0000
0000          ;              CRTLOOP.Z80
0000
0000
0000
0000
0000          ;       ROUTINE TO TEST CRT PORTS
0000          ; VERSION 1.1 12-30-82
0000          ;
0000          ;         ORG=100H  & 8080 FORMAT
0000          ;
0000          CRTS    EQU     0F9H    ;HOST SYSTEM KEYBOARD STATUS PORT
0000          CRTIN   EQU     0FAH    ;HOST SYSTEM KEYBOARD DATA PORT
0000          CRTO    EQU     0F0H    ;VDB-A STATUS OUT/ DATA INPUT PORT
0000          CRTIN2  EQU     0F2H    ;VDB-A KEYBOARD DATA PORT
0000          CRTS2   EQU     0F1H    ;VDB-A KEYBOARD STATUS PORT
0000          ;
0100          ORG     100H
0100          ;
0100 DBF9     START:  IN      A,(CRTS)        ;CHECK STATUS OF HOST KEYBOARD
0102 17               RLA                     ;ROTATE TO CARRY
0103 DA1101           JP      C,CRT1          ;NO-CONTINUE
0106 DBF1             IN      A,(CRTS2)       ;CHECK VDB-A KEYBOARD STATUS
0108 17               RLA
0109 D20001           JP      NC,START        ;NO DATA - START OVER
010C DBF2             IN      A,(CRTIN2)      ;YES - GET DATA
010E C31301           JP      CTLC
0111 DBFA     CRT1:   IN      A,(CRTIN)       ;YES-GET DATA
0113 FE03     CTLC:   CP      3               ;CTL-C CHECK
0115 CA0000           JP      Z,0             ;REBOOT TO CP/M
0118 4F               LD      C,A             ;TEMP STORE
0119 DBF0     CRTOUT: IN      A,(CRTO)        ;CHECK STATUS
011B 17               RLA                     ;ROTATE TO CARRY
011C DA1901           JP      C,CRTOUT        ;NO-TRY AGAIN
011F 79               LD      A,C             ;GET DATA
0120 D3F0             OUT     (CRTO),A        ;SEND DATA
0122 3E00             LD      A,0             ;FOR NEATNESS
0124 C30001           JP      START           ;LOOP
0000          END


0111 CRT1    00FA CRTIN   00F2 CRTIN2
00F0 CRTO    0119 CRTOUT  00F9 CRTS
00F1 CRTS2   0113 CTLC    0100 START


00 ERRORS
```

ROM 2 Character Generator HEX Listing

```
Add.          Character cell data              Char.
====   ================================        =====
0000   00FE82828282828282FE000000000000        ' □ '
0010   00FE8080808080808000000000000000        ' ┌ '
0020   001010101010101010FE000000000000        ' ┴ '
0030   000202020202020202FE000000000000        ' ┘ '
0040   00402010087C20100804000000000000        ' ↯ '
0050   00FE82C6AA92AAC682FE000000000000        ' ⊠ '
0060   000002040890A0C08000000000000000        ' ✓ '
0070   003844828282FE2828EE000000000000        ' Ω '
0080   002040F8442202020202000000000000        ' ↰ '
0090   0000100804FE04081000000000000000        ' → '
00A0   00FE000000FE000000FE000000000000        ' ≡ '
00B0   00001010109254381000000000000000        ' ↓ '
00C0   00101054381092543810000000000000        ' ↕ '
00D0   0000102040FE40201000000000000000        ' ← '
00E0   003844C6AA92AAC64438000000000000        ' ⊗ '
00F0   00384482829282824438000000000000        ' ⊙ '
0100   00FE828282FE828282FE000000000000        ' ⊟ '
0110   00385492929E82824438000000000000        ' ◷ '
0120   00384482829E92925438000000000000        ' ◶ '
0130   0038448282F292925438000000000000        ' ◵ '
0140   0038549292F282824438000000000000        ' ◴ '
0150   000022140894A2C08000000000000000        ' × '
0160   007C44444444444444C6000000000000        ' Π '
0170   0002020202FE02020202000000000000        ' ┤ '
0180   00FE82442810284482FE000000000000        ' ⌛ '
0190   00101010383810101010000000000000        ' ┼ '
01A0   00788484806010100010000000000000        ' ? '
01B0   0038448282FE82824438000000000000        ' ⊖ '
01C0   00FE929292F2828282FE000000000000        ' ◰ '
01D0   00FE828282F2929292FE000000000000        ' ◱ '
01E0   00FE8282829E929292FE000000000000        ' ◲ '
01F0   00FE9292929E828282FE000000000000        ' ◳ '
0200   00000000000000000000000000000000        '   '
0210   00101010100010100000000000000000        ' ! '
0220   00484848000000000000000000000000        ' " '
0230   002828FE28FE28280000000000000000        ' # '
0240   00103C50381478100000000000000000        ' $ '
0250   0042A44810244A840000000000000000        ' % '
0260   00205050324C4C320000000000000000        ' & '
0270   00181810200000000000000000000000        ' ' '
0280   00081020202010080000000000000000        ' ( '
0290   00100804040408100000000000000000        ' ) '
02A0   00105438103854100000000000000000        ' * '
02B0   000010107C1010000000000000000000        ' + '
02C0   00000000000030302040000000000000        ' , '
02D0   000000007C0000000000000000000000        ' - '
02E0   00000000000030300000000000000000        ' . '
02F0   00020408102040800000000000000000        ' / '
0300   0038444C546444380000000000000000        ' 0 '
0310   00103010101010380000000000000000        ' 1 '
0320   003844041820407C0000000000000000        ' 2 '
0330   00384404180444380000000000000000        ' 3 '
```

```
0340     00081828487C08080000000000000000     '  4  '
0350     007C4040780404780000000000000000     '  5  '
0360     00384440784444380000000000000000     '  6  '
0370     007C0408101010100000000000000000     '  7  '
0380     00384444384444380000000000000000     '  8  '
0390     003844443C0444380000000000000000     '  9  '
03A0     00000030300030300000000000000000     '  :  '
03B0     00000030300030302040000000000000     '  ;  '
03C0     00081020402010080000000000000000     '  <  '
03D0     0000007C007C00000000000000000000     '  =  '
03E0     00201008040810200000000000000000     '  >  '
03F0     00384404081000100000000000000000     '  ?  '
0400     003C44BAAABC403E0000000000000000     '  @  '
0410     001028444447C4444000000000000000     '  A  '
0420     00784444784444780000000000000000     '  B  '
0430     00384440404044380000000000000000     '  C  '
0440     00784444444444780000000000000000     '  D  '
0450     007C404078404070C000000000000000     '  E  '
0460     007C4040784040400000000000000000     '  F  '
0470     0038444040 5C44380000000000000000     '  G  '
0480     004444447C4444440000000000000000     '  H  '
0490     007C10101010107C0000000000000000     '  I  '
04A0     003C0808080848300000000000000000     '  J  '
04B0     00444850605048440000000000000000     '  K  '
04C0     004040404040407C0000000000000000     '  L  '
04D0     00446C54544444440000000000000000     '  M  '
04E0     004444645 44C44440000000000000000     '  N  '
04F0     00384444444444380000000000000000     '  O  '
0500     00784444784040400000000000000000     '  P  '
0510     00384444445448340000000000000000     '  Q  '
0520     00784444785048440000000000000000     '  R  '
0530     00384440380444380000000000000000     '  S  '
0540     007C1010101010100000000000000000     '  T  '
0550     00444444444444380000000000000000     '  U  '
0560     00444444444428100000000000000000     '  V  '
0570     004444445454 6C440000000000000000     '  W  '
0580     00444428102844440000000000000000     '  X  '
0590     00444428101010100000000000000000     '  Y  '
05A0     007C04081020407C0000000000000000     '  Z  '
05B0     00382020202020380000000000000000     '  [  '
05C0     00804020100804020000000000000000     '  \  '
05D0     00380808080808380000000000000000     '  ]  '
05E0     00102844000000000000000000000000     '  ^  '
05F0     00000000000000FE0000000000000000     '  _  '
0600     00303010080000000000000000000000     '  `  '
0610     000000380 43C443C0000000000000000     '  a  '
0620     00404078444444780000000000000000     '  b  '
0630     00000038444040380000000000000000     '  c  '
0640     0004043C4444443C0000000000000000     '  d  '
0650     00000038447840380000000000000000     '  e  '
0660     00102820702020200000000000000000     '  f  '
0670     00000438444444380478000000000000     '  g  '
0680     00404070484848480000000000000000     '  h  '
0690     00100030101010380000000000000000     '  i  '
06A0     00080018080808082810000000000000     '  j  '
06B0     00404048506050480000000000000000     '  k  '
```

```
06C0    00301010101010380000000000000000     ’ l ’
06D0    00000068545454540000000000000000     ’ m ’
06E0    00000070484848480000000000000000     ’ n ’
06F0    00000038444444380000000000000000     ’ o ’
0700    00000038444444784040000000000000     ’ p ’
0710    000000438444443C0406000000000000     ’ q ’
0720    00000058644040400000000000000000     ’ r ’
0730    0000003C403804780000000000000000     ’ s ’
0740    00202078202024180000000000000000     ’ t ’
0750    000000444444443C0000000000000000     ’ u ’
0760    00000044444428100000000000000000     ’ v ’
0770    00000044545454280000000000000000     ’ w ’
0780    00000044281028440000000000000000     ’ x ’
0790    000000444444443C0478000000000000     ’ y ’
07A0    0000007C0810207C0000000000000000     ’ z ’
07B0    00182020402020180000000000000000     ’ { ’
07C0    00101010001010100000000000000000     ’ ¦ ’
07D0    00300808040808300000000000000000     ’ } ’
07E0    00205408000000000000000000000000     ’ ~ ’
07F0    00489224489224489224000000000000     ’ ▨ ’
0000    END
```

EXAMPLE:

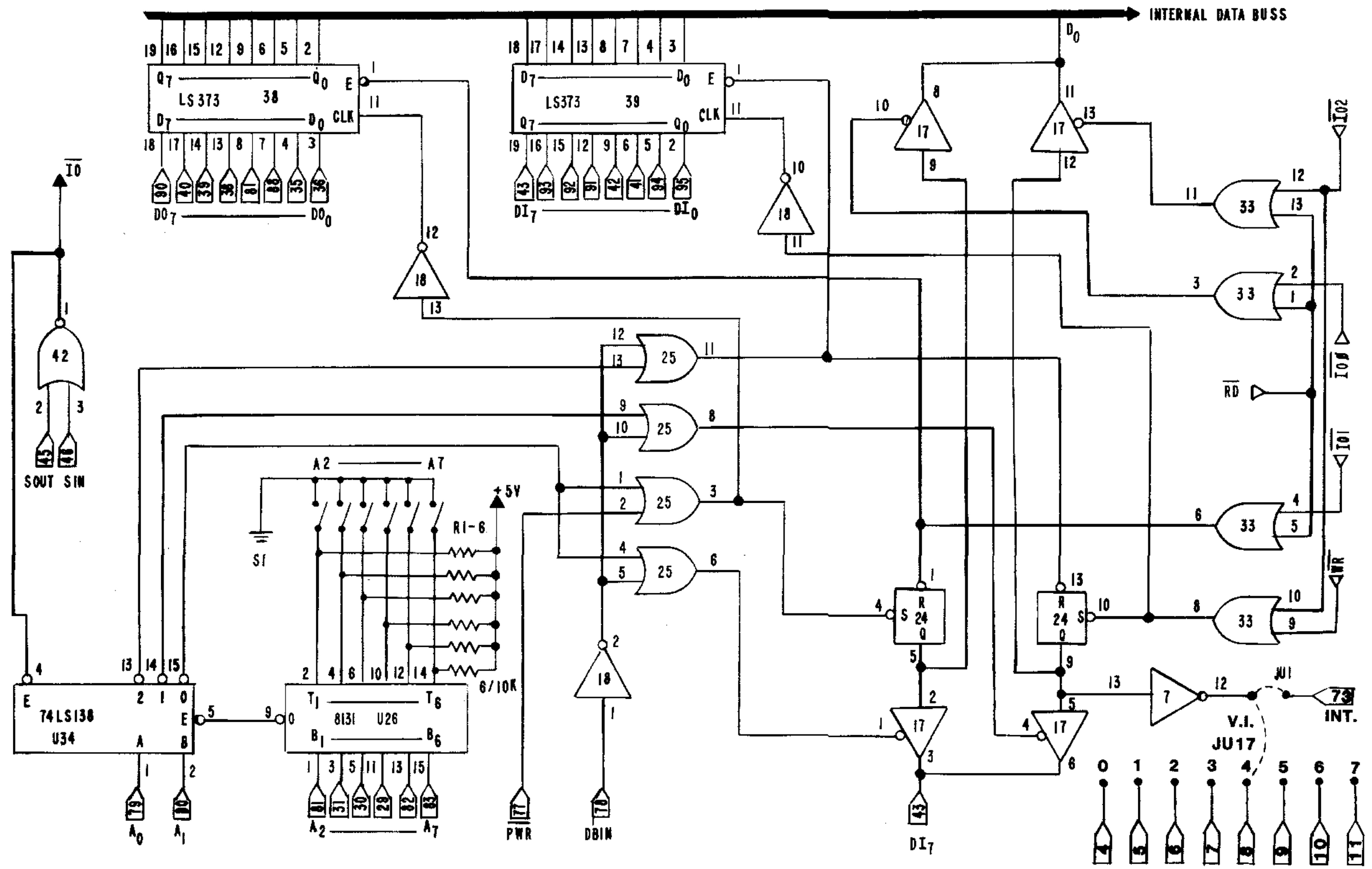
Addr.	Data	Pattern
0410H	00	00000000
0411H	10	00010000
0412H	28	00101000
0413H	44	01000100
0414H	44	01000100
0415H	7C	01111100
0416H	44	01000100
0417H	44	01000100
0418H	00	00000000
0419H	00	00000000

Result: ’A’ ASCII 41H

Note:
1. The first byte of each block of cell data is allways 00H to provide vertical spacing between characters
2. The last 6 bytes of each block is unused, 00H
3. Bit 0 (lsb) of each byte is not used (0) or output to the CRT
4. The cell matrix is actually 7x9 dots, but a 5x7 matrix is used for alpha-numerics to allow lower case decenters. USE 7x9 MATRIX, WITH NOTES 1&2 ABOVE, FOR YOUR CUSTOM GRAPHICS EPROM (ROM 3)

Troubleshooting Hints for the VDB-A Video Display Board

Problem	Probable Cause
1. No Sync on Screen	1a. Adjust Sync Pots b. No POC c. bad 8275, U4. d. check sync ckts e. Program not executing
2. No Character on Screen & Cursor does not move	2a. I/O address wrong b. Check S100 interface cicuits
3. No Character on Screen but cursor moves ok	3a. 8275, U4, not sending DRQ b. Z80, U20, not accepting NMI c. NMI flip-flop not being set or reset
4. No keyboard response	4a. Strobe not geting to 8212, U22, or wrong polarity b. Z80, U20, not accepting INT c. Check operation of S100 interface circuits
5. No attributes, or only some missing	5a. check outputs of 8275, U4 b. check 74LS125, U17, and video summing circuits
6. Z80, U20, seems lost - random chars. on screen	6a. check RAM and ROM and Z80 b. address line foil shorts
7. Memory Map wrong address	7a. Wrong address selected b. screen parameters not matched c. check address latches
8. Memory Map not working	8a. enable flip-flop not set
9. Ready line not setting or releasing	9a. status flip-flop not working
10. Cursor lags behind character or every 8th char. repeats	10a. raise C48 to .001 uF. (VDB-A & -A1 revisions only)
11. Crystal won't start, or is erratic.	11a. replace U3, 74LS04 b. change U3 to a 7404.
12. Multiple cursors, spread out char's unstable display	12a. Monitor must be adjusted or modified to get 19 kHz horiz. sync frequency: R-C network in horiz osc changed
13. character line tears	13a. increase value of C54 tO 47uF
14. scan line jitters on warm-up; char shifts (VDB-A & -A1 only)	14a. Change C14 to .0022uF poly styrene b. replace U23



NOTES: 1. ALL RESISTORS 1/4 5% (UNLESS OTHERWISE NOTED).

2. ALL CAPACITORS IN MICRO-FARADS (UNLESS OTHERWISE NOTED).

3. VOLTAGES SHOWN ARE NOMINAL.

ISSUE	DATE
B	5/09/83
A	3/30/82

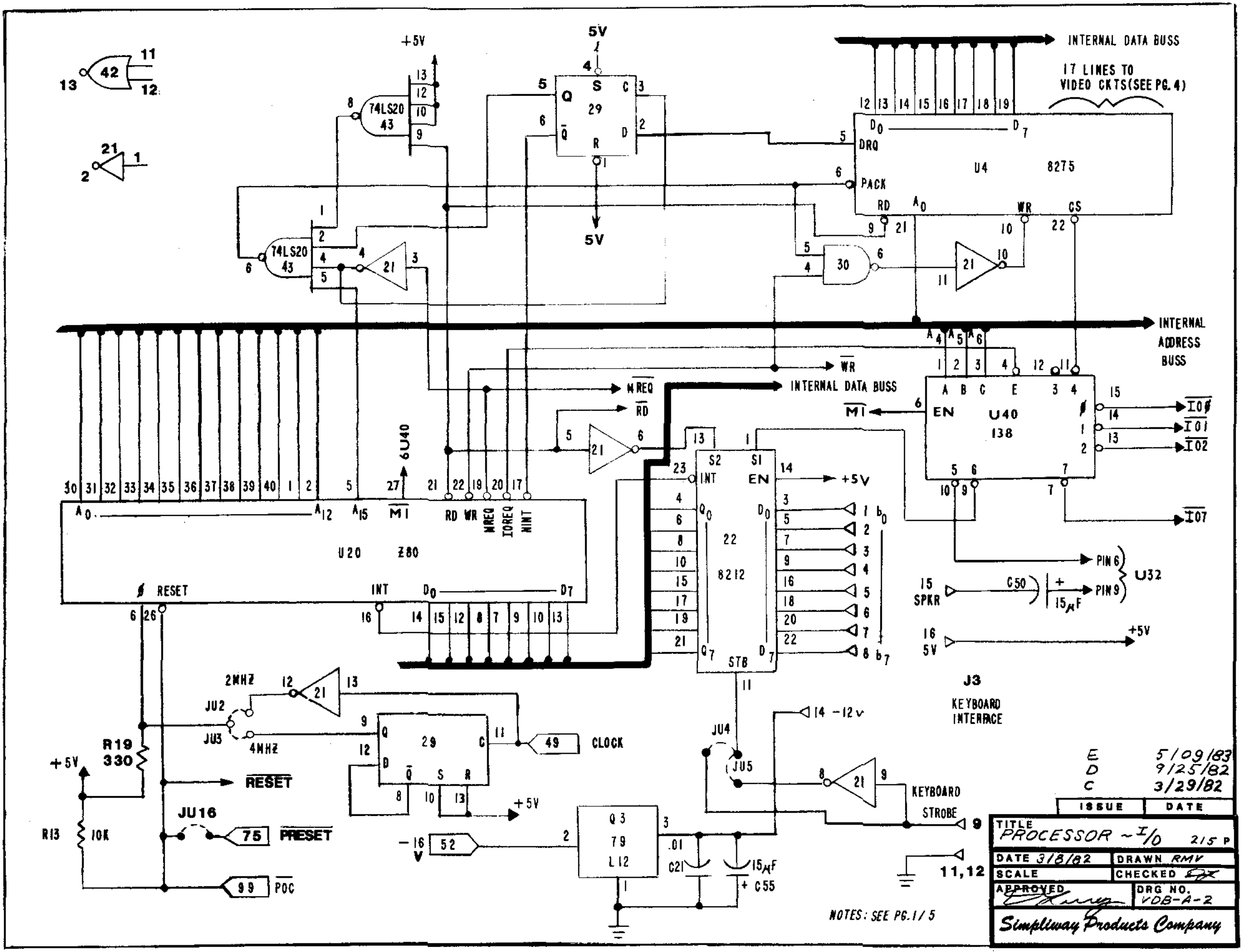
TITLE S-100 INTERFACE 115 P

DATE 3/18/82 DRAWN RMV

SCALE ~ CHECKED LR

APPROVED DRG NO. VDB-A-1

Simpliway Products Company



ISSUE	DATE
E	5/09/83
D	9/25/82
C	3/29/82

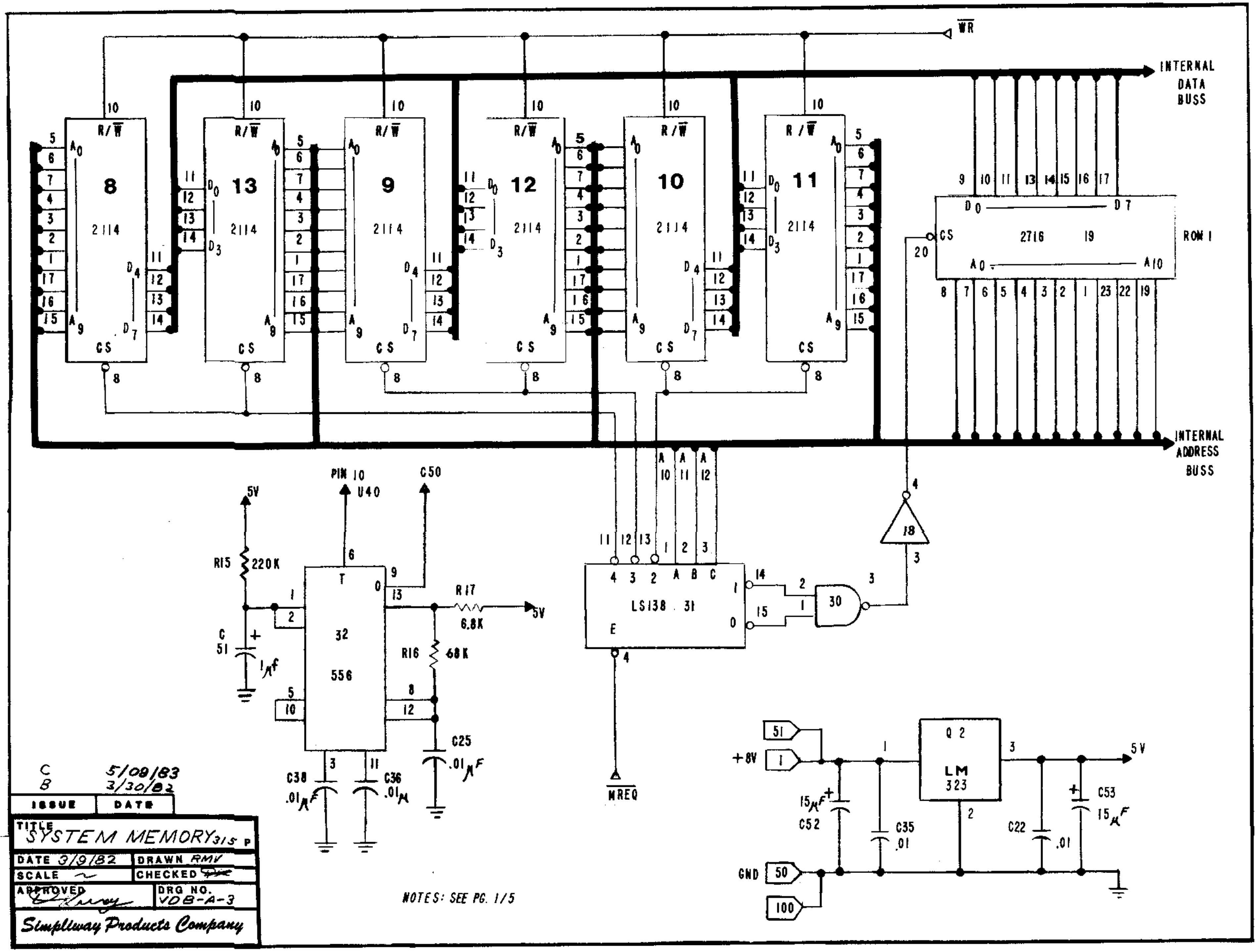
TITLE PROCESSOR ~ I/O 2/5 P

DATE 3/8/82 DRAWN RMV

SCALE CHECKED

APPROVED DRG NO. VDB-A-2

Simpliway Products Company



NOTES: SEE PG. 1/5

ISSUE	DATE
C	5/09/83
B	3/30/82

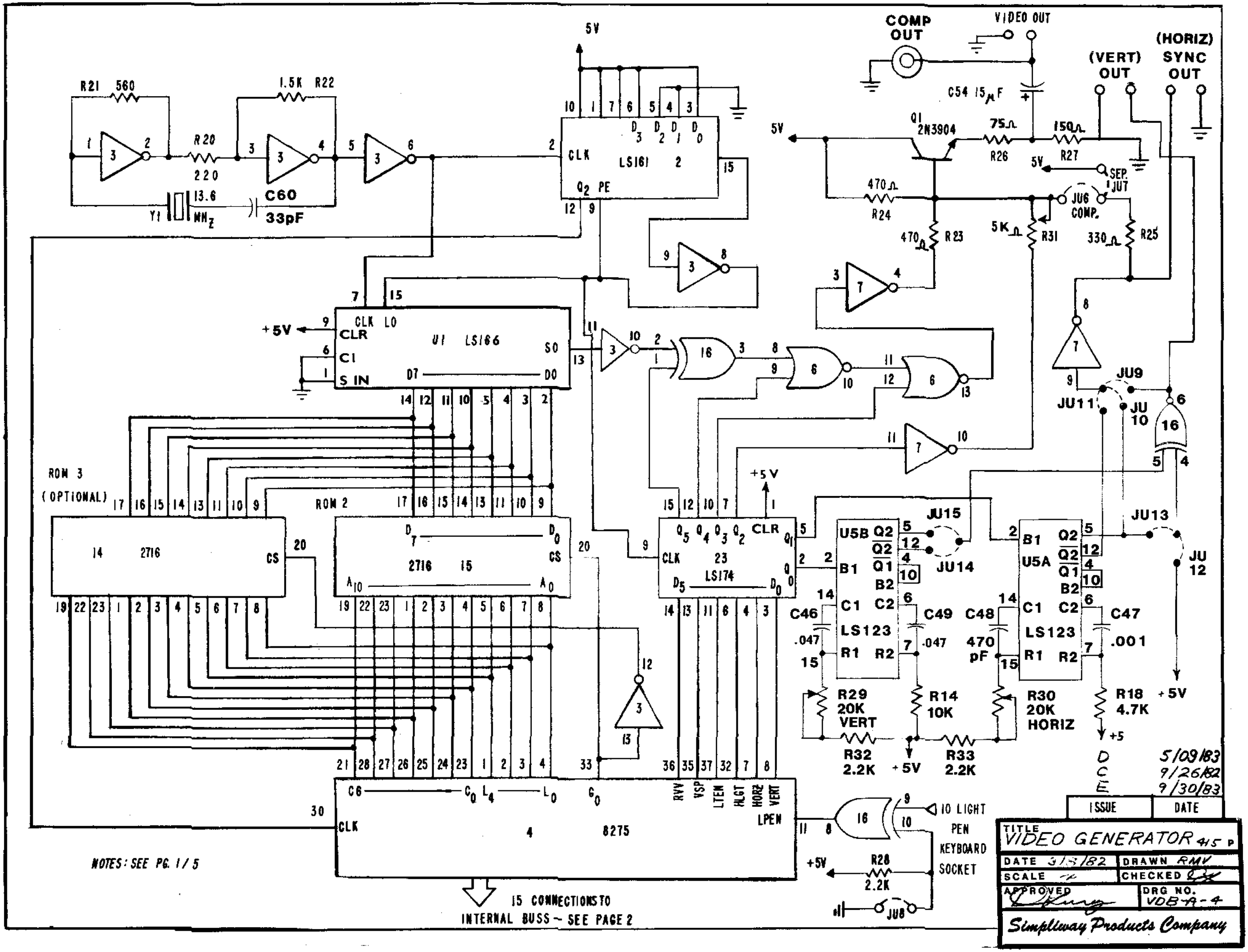
TITLE SYSTEM MEMORY 315 P

DATE 3/9/82 DRAWN RMV

SCALE ~ CHECKED

APPROVED DRG NO. VDB-A-3

Simpliway Products Company



NOTES: SEE PG. 1/5

ISSUE	DATE
D	5/09/83
C	9/26/82
E	9/30/83

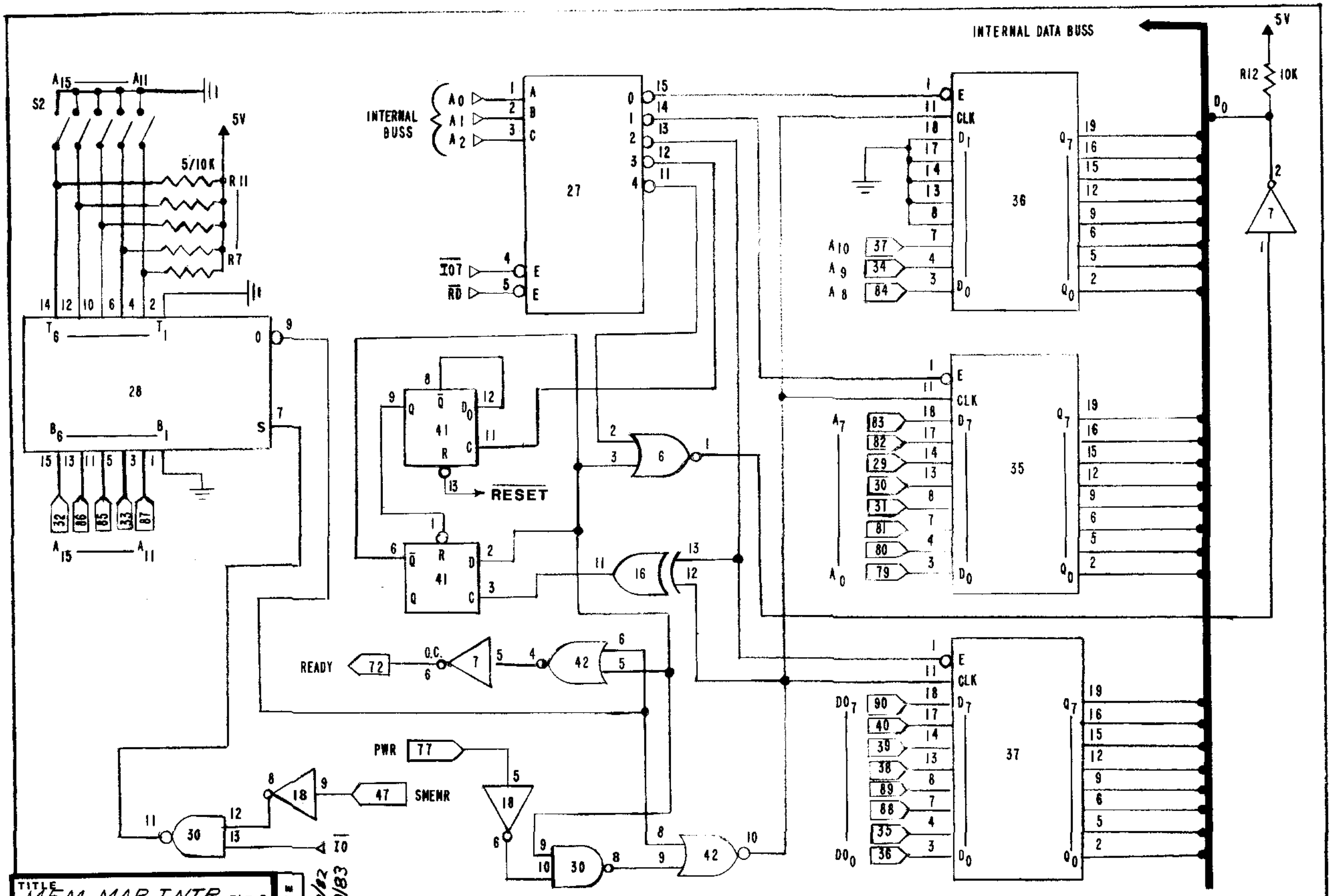
TITLE VIDEO GENERATOR 4/5 P

DATE 3/5/82 DRAWN RMV

SCALE ~ CHECKED

APPROVED DRG NO. VDB-A-4

Simpliway Products Company



TITLE MEM. MAP INTR. 5/5 P

DATE 3/9/82

DRAWN RMV

SCALE ~

CHECKED

APPROVED

DRG NO. VDB-A-5

Simpliway Products Company

ISSUE	DATE
A	3/30/82
B	5/09/83

ORDERING INFORMATION Oct. 10, 1984

TO ORDER -- Please check the items desired:

PROTOTYPING BOARD price information:

PTB-1 with documentation

[] Bare board	$ 67.50
[] Complete kit	$ 89.50
[] Assembled and tested	$ 95.00
[] Complete instruction manual (Refundable with PTB-1 purchase)	$ 5.00

BUFFERED I/O BOARD price information:

BI/O-1 with documentation and ROM 1 source code.

[] Bare board with heat-sink	$ 59.95
[] Complete kit, 64K, WITH eprom & source	$289.00
[] Assembled and soldered, but WITHOUT ic's or eprom. You use your own, and test it.	$159.00
[] Assembled and tested, with 64 Kbyte of ram, 3.68 MHz crystal, and source code.	$325.00

BI/O Interface board. This board mounts to the rear panel of your computer via a DB-25 connector. It may be wired for serial or parallel port operation

[] BI/O I-B, bare board w/documentation	$ 9.95
[] Complete kit, serial or parallel oper.	$ 25.00
[] Assembled and tested, serial operation	$ 35.00
[] Assembled and tested, parallel operation	$ 30.00

Miscellaneous

[] Serial cable assembly for BI/O I-B, 1.5 ft.	$ 10.95
[] Parallel cable assembly as above	$ 8.95
[] ROM 1 programmed eprom, 350 nS, with source	$ 30.00
[] Crystal, 3.6864 MHz.	$ 7.00
[] I-C set with programmed eprom	$145.50

Documentation

[] Complete instruction manual, less source code	$ 5.00
[] Source code for ROM 1	$ 15.00
(Both above refundable with BI/O-1 purchase)	
[] Source code for Utility Program below	$ 20.00
[] Disk, containing the source and data files for the eprom, ROM 1, and a utility program for easy formatting of the BI/O-1 for your system (8" ss,sd)	$ 35.00
Please add $3.00 for shipping & handling (or $1.50 for documentation only)	3.00
TOTAL	______

(Please add to SUB-TOTAL on other side)

<Ordering information continued on other side>

TO ORDER: Please check the items desired: May. 30, 1984

VDB-A2 Video Board can be ordered in 3 ways:

[] Bare board with standard 24 x 80 source code, includes heat sink and phono connector.	$ 49.50
[] Complete kit less crystal & eproms (Order eproms & crystal to match Rel. #, below)	$197.50
[] Assembled & soldered, but WITHOUT IC's/eproms you get your own & test. (state crystal Freq)	$182.50

VDB-A2 Assembled and tested, with programmed eproms

[] With standard 24 x 80 source code	$269.00
[] With VIDSTAR (Wordstar/dBase II) source code	279.00
[] With VIDSTAR 25th line non-scrolling, status "VIDSTAR" emulates the Televideo 920 terminal!	285.00

ROM 1 Pre-programmed eproms for above source code

[] Release 1.x : 24 lines x 80 characters, std.	18.50
[] Release 3.x : VIDSTAR for Wordstar / dBASE II	28.50
[] Release 4.x : VIDSTAR 25th line non-scrolling	32.50

Character Generator eproms, '2716, 450 nS (ok at 4 MHz)

[] ROM 2 Std. alpha-numeric, 5 x 7 matrix & 1-c	18.50
[] ROM 3 Graphics, Greek, a Potpourri (no source)	18.50

Source and Data on 8" SSSD CP/M(tm) Disc (inc ROM 2)

[] Release 1.x : 24 lines x 80 characters, std.	8.50
[] Release 3.x : VIDSTAR for Wordstar / dBASE II	22.75
[] Release 4.x : VIDSTAR with 25th line nonscroll	27.75

Crystals, HC-18 size, wire leads, for Rel's 1.x & 3.x

[] 13.628 MHz crystal (60 Hz. - USA)	18.9kHz	7.00
[] 11.357 MHz crystal (50 Hz. - foreign)	18.9kHz	9.00
For Rel. 2.x (obsolete) and 4.x only :		
[] 11.340 MHz crystal (60 Hz. - USA)	15.7kHz	8.00
[] 9.4500 MHz crystal (50 Hz. - foreign)	15.7kHz	9.00

Miscellaneous

[] Serial Keyboard port adapter **	T.B.A.
[] LP-A LIGHT PEN for the VDB-A **	T.B.A.
[] Trim-pot set (2 - 20kohm, 1 - 5kohm)	4.50
[] '8275 CRT controller IC	27.75
[] Complete set of IC's & Q1, less eproms	73.00

Documentation	Complete	source only	
[] Release 1.x,	10.00	5.00	
[] Release 3.x,	16.00	11.00	
[] Release 4.x,	20.00	15.00	
[] No source list	5.00	-----	
(refundable with VDB-A purchase) + 1.50 S&H			______

Please add $3.00 for shipping & handling	3.00
(Ill. residents please add 7% sales tax)	______
Prices subject to change without notice TOTAL	______

[] VISA / [] MASTERCARD NO. _____ _____ _____ _____

Expiration date __________ Init. ________

[] Send C.O.D. [] Payment enclosed Amount $_________

Mail to:

SIMPLIWAY PRODUCTS CO.
P.O. Box 601
Hoffman Estates, Il.
60195

or Phone:
(312) 359 - 7337

** COMING SOON !!

The form below is your shipping label. Please type or print:

Ship To: Name ______________________________________

Address ______________________________________

City ______________________________________

State ________________ ZIP __________

Simpliway Products Company

P.O. BOX 601
HOFFMAN ESTATES, IL 60195

HOW TO CONFIGURE 'WORDSTAR' & 'dBASE II'
FOR THE
SIMPLIWAY VDB-A VIDEO BOARD
Releases 3.x & 4.x

Using the control codes that come standard with your Simpliway Video Board, configure your Wordstar in the following mannor. With the INSTALL program, configure Wordstar to use the Televideo 912/920 terminal and the printer you are using. After you have installed the terminal and printer, enter the patch area by answering 'NO' to the promt, and make the following changes:

ADDRESS	DATA
ERAEOL:	2
ERAEOL:+1	1B
ERAEOL:+2	54
LINDEL:	2
LINDEL:+1	1B
LINDEL:+2	52
LININS:	2
LININS:+1	1B
LININS:+2	45
DELCUS:	0
DELMIS:	0

Now save your Wordstar by typing 'O' and 'Y'.

Your Simpliway Video Board can now take advantage of all of the Televideo 912/920 terminal features offered by Wordstar.

Installing DBASE II

1. Install Televideo 920 terminal
2. Change clear screen command from 'ESC- *' to 'ESC- Z'.

```
0000            ;       VIDEO DISPLAY SOFTWARE ROM-1    VERSION 4.00 09-09-83
0000            ;
0000            ;COPYRIGHT (c) by
0000            ;The SIMPLIWAY PRODUCTS CO.
0000            ;all rights reserved
0000            ;
0000            ;BY ROGER VILMUR
0000            ;This version has 25 lines. The 25th line is a status line.
0000            ;This version uses a 15 KHz horizontal sweep rate.
0000            ;
0000            ;       DEFINED PARAMETERS
0000            ;
0000            ;       DEFINED USE OF REGISTERS
0000            ;
0000            ;       For transfer of data to screen
0000            ;
0000            ;       A = transfer of data
0000            ;       B = temperary storage       C = cursor line
0000            ;       D = zero                    E = cursor column
0000            ;       HL = screen address pointer
0000            ;
0000            ;       For transfer of data from keyboard
0000            ;
0000            ;       A = transfer of data
0000            ;       HL = buffer address pointer
0000            ;       IX = buffer counter address pointer
0000            ;
0000            ;       For transfer of data to CRT controler
0000            ;
0000            ;       DE = start address of transfer
0000            ;
0000            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
0000            ;                                                           ;
0000            ;               USER MODIFIABLE AREA                        ;
0000            ;                                                           ;
0000            ;       To reconfigure an attribute to the code             ;
0000            ;       desired place the ASCII code after the EQU.         ;
0000            ;       Control codes may only be place in the              ;
0000            ;       control character area.  Attributes maybe           ;
0000            ;       disabled by placing a COH after the EQU.            ;
0000            ;                                                           ;
0000            ;       SCREEN ATTRIBUTES USING CONTROL CHARACTERS          ;
0000            ;                                                           ;
0000            CHOME   EQU     1AH     ;HOME                               ;
0000            CSTCUR  EQU     00H     ;SET CURSOR POSTION                 ;
0000            CCURUP  EQU     0BH     ;CURSOR UP                          ;
0000            CCURDN  EQU     0AH     ;CURSOR DOWN                        ;
0000            CCURRT  EQU     0CH     ;CURSOR RIGHT                       ;
0000            CCURLT  EQU     08H     ;CURSOR LEFT                        ;
0000            CEOL    EQU     00H     ;ERASE TO END OF LINE               ;
0000            CEOS    EQU     00H     ;ERASE TO END OF SCREEN             ;
0000            CHOMUP  EQU     1EH     ;HOME UP                            ;
0000            CBELL   EQU     07H     ;BELL                               ;
0000            CHLITE  EQU     00H     ;HIGHLITE ON                        ;
0000            CNHLIT  EQU     00H     ;HIGHLITE OFF                       ;
0000            CREVID  EQU     00H     ;REVERSE VIDEO ON                   ;
0000            CNRVID  EQU     00H     ;REVERSE VIDEO OFF                  ;
0000            CBLINK  EQU     00H     ;BLINK ON                           ;
0000            CNBLNK  EQU     00H     ;BLINK OFF                          ;
0000            CLPEN   EQU     00H     ;LIGHT PEN POSTION                  ;
0000            CMEM    EQU     00H     ;MEMORY MAP MODE                    ;
```

```
0000            CMEMO   EQU     00H     ;MEMORY MAP OFF                               ;
0000            CRDCUR  EQU     00H     ;READ CURSOR POSITION                         ;
0000            CSCLWT  EQU     00H     ;CHANGE SCROLL SPEED                          ;
0000            CALTCH  EQU     00H     ;CHANGE CHARACTER ROM                         ;
0000            CDELLN  EQU     00H     ;DELETE A LINE                                ;
0000            CINLN   EQU     00H     ;INSERT A LINE                                ;
0000            CGRAF   EQU     16H     ;PUT CTL CHAR ON SCREEN                       ;
0000            CHTOG   EQU     0EH     ;HIGHLITE TOGGLE                              ;
0000            COLDPN  EQU     01H     ;RETURN TO OLD POSITION                       ;
0000            ;                                                                     ;
0000            ;       SCREEN ATTRIBUTES USING ESCAPE CHARACTER CODES                ;
0000            ;                                                                     ;
0000            EHOME   EQU     'Z'     ;HOME                                         ;
0000            ESTCUR  EQU     '='     ;SET CURSOR POSTION                           ;
0000            ECURUP  EQU     00H     ;CURSOR UP                                    ;
0000            ECURDN  EQU     00H     ;CURSOR DOWN                                  ;
0000            ECURRT  EQU     00H     ;CURSOR RIGHT                                 ;
0000            ECURLT  EQU     00H     ;CURSOR LEFT                                  ;
0000            EEOL    EQU     'T'     ;ERASE TO END OF LINE                         ;
0000            EEOS    EQU     'Y'     ;ERASE TO END OF SCREEN                       ;
0000            EHOMUP  EQU     00H     ;HOME UP                                      ;
0000            EBELL   EQU     00H     ;BELL                                         ;
0000            EHLITE  EQU     ')'     ;HIGHLITE ON                                  ;
0000            ENHLIT  EQU     '('     ;HIGHLITE OFF                                 ;
0000            EREVID  EQU     'j'     ;REVERSE VIDEO ON                             ;
0000            ENRVID  EQU     'k'     ;REVERSE VIDEO OFF                            ;
0000            EBLINK  EQU     5EH     ;BLINK ON                                     ;
0000            ENBLNK  EQU     'q'     ;BLINK OFF                                    ;
0000            ELPEN   EQU     'L'     ;LIGHT PEN POSITION                           ;
0000            EMEM    EQU     'M'     ;MEMORY MAP MODE                              ;
0000            EMEMO   EQU     'D'     ;MEMORY MAP OFF                               ;
0000            ERDCUR  EQU     '?'     ;READ CURSOR POSTION                          ;
0000            ESCLWT  EQU     'S'     ;CHANGE SCROLL SPEED                          ;
0000            EALTCH  EQU     'X'     ;CHANGE CHARACTER ROM                         ;
0000            EDELLN  EQU     'R'     ;DELETE A LINE                                ;
0000            EINLN   EQU     'E'     ;INSERT A LINE                                ;
0000            EGRAF   EQU     00H     ;PUT CTL CHAR ON SCREEN                       ;
0000            EHTOG   EQU     00H     ;HIGHLITE TOGGLE                              ;
0000            EOLDPN  EQU     00H     ;RETURN OLD CURSOR POSITION                   ;
0000            ;                                                                     ;
0000            ;       MEMORY MAP DIMENSIONS                                         ;
0000            ;                                                                     ;
0000            MMLINE  EQU     24      ;NUMBER OF SCREEN LINES                       ;
0000            MMCOL   EQU     80      ;NUMBER OF COLUMNS                            ;
0000            ;                                                                     ;
0000            ;       NUMBER OF ATTRIBUTES                                          ;
0000            ;                                                                     ;
0000            ATTCTR  EQU     27      ;ATTRIBUTE TABLE COUNTER                      ;
0000            ;                                                                     ;
0000            ;       CURSOR OFFSET                                                 ;
0000            ;                                                                     ;
0000            OFFSET  EQU     20H     ;SOME TERMINALS USE IT                        ;
0000            ;                                                                     ;
0000            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
0000            ;
0000            ;       INTERNAL BUSS PORT ASSIGNMENTS
0000            ;
0000            ;       This is a list of I/O ports used on the
0000            ;       internal buss
0000            ;
0000            CRTC    EQU     41H     ;CRTC COMMAND/STATUS
```

```
0000        CRTD    EQU     40H     ;CRTC DATA
0000        DMARST  EQU     30H     ;DMA INTERRUPT RESET
0000        SPEAK   EQU     50H     ;SPEAKER
0000        KYS100  EQU     20H     ;KEYBOARD TO S100
0000        S100ST  EQU     00H     ;S100 PORT STATUS
0000        VDS100  EQU     10H     ;S100 TO VIDIO DISPLAY
0000        KLYBRD  EQU     60H     ;KEYBOARD TO DISPLAY BOARD
0000        MMH     EQU     70H     ;MEM MAP HI ADDR
0000        MML     EQU     71H     ;MEM MAP LO ADDR
0000        MMD     EQU     72H     ;MEM MAP DATA
0000        MMEN    EQU     73H     ;MEM MAP ENABLE
0000        MMS     EQU     74H     ;MEM MAP STATUS
0000        ;
0000        ;       COMMONLY USED PARAMETERS
0000        ;
0000        ;       This is a list of commonly used values
0000        ;
0000        HCR     EQU     4FH     ;SCREEN CONTROL PARAMETERS
0000        VVVR    EQU     58H     ;SEE INTEL 8275 DATA SHEET
0000        ULL     EQU     79H     ;FOR USE
0000        MFCH    EQU     09H
0000        LITP    EQU     60H     ;LIGHT PEN COMMAND
0000        CRTEI   EQU     0A0H    ;ENABLE CRT INTERRUPT
0000        STRSCR  EQU     23H     ;START SCREEN COMMAND
0000        CURPOS  EQU     80H     ;LOAD CURSOR COMMAND
0000        CR      EQU     0DH     ;CARRAGE RETURN
0000        LF      EQU     0AH     ;LINE FEED
0000        NULL    EQU     00H     ;NULL
0000        ESC     EQU     1BH     ;ESCAPE CODE
0000        ENDDMA  EQU     0F1H    ;STOP LINE DMA COMMAND
0000        ELCODE  EQU     0F0H    ;END OF LINE CODE
0000        STPDMA  EQU     0F3H    ;STOP SCREEN DMA COMMAND
0000
0000        ;
0000        ;       ABSOLUTE ADDRESSES
0000        ;
0000        ;       This is a list of common addresses used
0000        ;       by the program
0000        ;
0000        TOPSCR  EQU     800H    ;TOP OF DISPLAY MEMORY
0000        TOPSCP  EQU     TOPSCR+8000H    ;DISPLAY ADDR WITH DMA BIT
0000        STACK   EQU     11A0H   ;TOP OF STACK
0000        TOPKEY  EQU     11A0H   ;TOP OF KEYBOARD BUFFER
0000        BOTKEY  EQU     00D0H   ;MAX KEYBD BUFFER COUNT
0000        WTCTR   EQU     11D1H   ;SCROLL DELAY NUMBER
0000        BCTR    EQU     11D2H   ;KEYBOARD BUFFER COUNTER
0000        KYINA   EQU     11D3H   ;HOLDS ADDRESS OF KEYBOARD BUFFER IN PTR
0000        KYOTA   EQU     11D5H   ;HOLDS ADDRESS OF KEYBOARD BUFFER OUT PTR
0000        MMSTUS  EQU     11D7H   ;I/O OR MEMORY MAP FLAG
0000        ATTBUF  EQU     11D8H   ;CURRENT ATTRIBUTE BYTE REGISTER
0000        ATTFLG  EQU     11D9H   ;NEW ATTRIBUTE REQUEST FLAG
0000        TEMP    EQU     11DAH   ;TEMPORARY STORAGE
0000        TEMP2   EQU     11DBH   ;TEMPORARY ADDRESS STORAGE
0000        SAVLN   EQU     11DDH   ;OLD LINE POSITION POINTER
0000        SAVCOL  EQU     11DEH   ;OLD LINE POSITION POINTER
0000        SCRTBL  EQU     1200H   ;CRT LINE TRANSFER TABLE
0000        TEMPLN  EQU     1250H   ;TEMPORARY LINE BUFFER FOR ATT CHANGES
0000        DUMMY   EQU     0000H   ;DUMMY ADDRESS NAME
0000        ;
0000        ;       START OF PROGRAM
0000        ;
```

```
0000                    ORG     0000H
0000            ;
0000            ;       This begining of the program initializes all
0000            ;       hardware, sets the stack, initializes the RAM
0000            ;       and sets interrupts
0000            ;
0000 00                 NOP                     ;RESET STACK RETURN ADDRESS
0001 00                 NOP
0002 31A011             LD      SP,STACK        ;SET STACK
0005 ED56               IM1                     ;SET INTERRUPT MODE
0007 3E00               LD      A,0
0009 32D711             LD      (MMSTUS),A      ;ZERO FLAG TO I/O
000C D341               OUT     (CRTC),A        ;RESET CRTC
000E 0E40               LD      C,CRTD          ;LOAD C WITH CRTC ADDR
0010 0604               LD      B,4
0012 210A02             LD      HL,PARTBL       ;LOAD SREEEN PARAMETER TABLE
0015 EDB3               OTIR                    ;OUTPUT TO CRTC
0017 21A011             LD      HL,TOPKEY       ;INITIALIZE KEYBD BUFFER POINTERS
001A 22D311             LD      (KYINA),HL
001D 22D511             LD      (KYOTA),HL
0020 DD21D211           LD      IX,BCTR         ;SET KEYBD BUFFER COUNTER
0024 CD1102             CALL    INTST           ;SET UP SCREEN
0027 DB10               IN      A,(VDS100)      ;CLEAR PORTS
0029 DB60               IN      A,(KEYBRD)
002B D330               OUT     (DMARST),A      ;RESET DMA INTERRUPT
002D C3CF00             JP      MORINT
0030            ;
0030            ;       KEYBOARD SERVICE ROUTINE
0030            ;
0030            ;       This routine reads the keyboard port and places the
0030            ;       data in the keyboard buffer where the data is held
0030            ;       until the host computer is ready for it.
0030            ;
0038                    ORG     38H
0038            ;
0038 E5                 PUSH    HL              ;SAVE REGISTERS
0039 F5                 PUSH    AF
003A 2AD311             LD      HL,(KYINA)      ;GET ADDRESS POINTER
003D 7D                 LD      A,L             ;CHECK TO SEE IF AT END OF BUFFER
003E FED0               CP      BOTKEY
0040 2003               JR      NZ,NOBOT
0042 21A011             LD      HL,TOPKEY       ;LOAD POINTER TO TOP
0045 DB60       NOBOT:  IN      A,(KEYBRD)      ;GET DATA FROM PORT
0047 77                 LD      (HL),A          ;SAVE DATA IN BUFFER
0048 23                 INC     HL              ;INC POINTER
0049 DD3400             INC     (IX+0)          ;INC COUNTER
004C 22D311             LD      (KYINA),HL      ;SAVE ADDRESS POINTER
004F F1                 POP     AF              ;RESTORE REGISTERS
0050 E1                 POP     HL
0051 FB                 EI                      ;ENABLE KEYBOARD INTERRUPT
0052 C9                 RET                     ;RETURN FROM INTERRUPT
0053            ;
0053            ;       SCREEN REFRESH SERVICE ROUTINE
0053            ;
0053            ;       This routine transfers the characters from memory
0053            ;       and loads them into the CRT controller.  This is
0053            ;       done by placing the start of the line in the stack
0053            ;       and "poping" the data onto the buss.  The CRT con-
0053            ;       troller latches the data when address 15 is high.
0053            ;       To decrease overhead the characters are periodicly
0053            ;       checked for end of line codes.
```

```
0053            ;
0066                    ORG 66H
0066            ;
0066 08                 EX      AF,AF'          ;USE ALT A
0067 D9                 EXX                     ;USE ALT REGISTERS
0068 FD5E00             LD      E,(IY+0)        ;LOAD DE' WITH NEXT LINE
006B FD5601             LD      D,(IY+1)
006E 210000             LD      HL,00           ;ZERO HL
0071 39                 ADD     HL,SP           ;SAVE SP
0072 EB                 EX      DE,HL
0073 F9                 LD      SP,HL           ;LOAD LINE ADDR IN SP
0074 F1                 POP     AF              ;TRANSFER TO CRTC
0075 F1                 POP     AF
0076 F1                 POP     AF
0077 F1                 POP     AF
0078 F1                 POP     AF
0079 F1                 POP     AF
007A F1                 POP     AF
007B F1                 POP     AF
007C FEF1               CP      ENDDMA          ;END OF LINE?
007E 2839               JR      Z,NOMORE
0080 F1                 POP     AF
0081 F1                 POP     AF
0082 F1                 POP     AF
0083 F1                 POP     AF
0084 F1                 POP     AF
0085 F1                 POP     AF
0086 F1                 POP     AF
0087 F1                 POP     AF
0088 FEF1               CP      ENDDMA          ;END OF LINE?
008A 282D               JR      Z,NOMORE
008C F1                 POP     AF
008D F1                 POP     AF
008E F1                 POP     AF
008F F1                 POP     AF
0090 F1                 POP     AF
0091 F1                 POP     AF
0092 F1                 POP     AF
0093 F1                 POP     AF
0094 FEF1               CP      ENDDMA          ;END OF LINE?
0096 2821               JR      Z,NOMORE
0098 F1                 POP     AF
0099 F1                 POP     AF
009A F1                 POP     AF
009B F1                 POP     AF
009C F1                 POP     AF
009D F1                 POP     AF
009E F1                 POP     AF
009F F1                 POP     AF
00A0 FEF1               CP      ENDDMA          ;END OF LINE?
00A2 2815               JR      Z,NOMORE
00A4 F1                 POP     AF
00A5 F1                 POP     AF
00A6 F1                 POP     AF
00A7 F1                 POP     AF
00A8 F1                 POP     AF
00A9 F1                 POP     AF
00AA F1                 POP     AF
00AB F1                 POP     AF
00AC FEF1               CP      ENDDMA          ;END OF LINE?
00AE 2809               JR      Z,NOMORE
```

```
00B0 F1                 POP     AF
00B1 F1                 POP     AF
00B2 F1                 POP     AF
00B3 F1                 POP     AF
00B4 F1                 POP     AF
00B5 F1                 POP     AF
00B6 F1                 POP     AF
00B7 F1                 POP     AF
00B8 F1                 POP     AF
00B9 EB         NOMORE: EX      DE,HL
00BA F9                 LD      SP,HL           ;RESTORE SP
00BB AF                 XOR     A               ;CLEAR A
00BC FD23               INC     IY              ;POINT TO NEXT LINE
00BE FD23               INC     IY
00C0 FDBE01             CP      (IY+1)          ;IS IY AT TABLE BOTTOM
00C3 2004               JR      NZ,CONT1
00C5 FD210012           LD      IY,SCRTBL       ;YES - POINT TO TOP OF TABLE
00C9 D330       CONT1:  OUT     (DMARST),A      ;RESET INTERRUPT
00CB D9                 EXX                     ;RETURN REGISTERS
00CC 08                 EX      AF,AF'
00CD ED45               RETN
00CF            ;
00CF            ;       TERMINAL STATUS ROUTINE
00CF            ;
00CF            ;       For I/O mode
00CF            ;
00CF AF         MORINT: XOR     A               ;ZERO A REG
00D0 32D211             LD      (BCTR),A        ;ZERO BUFFER COUNTER
00D3 32D111             LD      (WTCTR),A       ;ZERO SCREEN SPEED CTR
00D6 32D911             LD      (ATTFLG),A      ;ZERO ATTRIBUTE FLAG
00D9 32DD11             LD      (SAVLN),A       ;ZERO POSITION POINTER
00DC 32DE11             LD      (SAVCOL),A      ;ZERO POSITION POINTER
00DF 3E23               LD      A,STRSCR        ;START DISPLAY
00E1 D341               OUT     (CRTC),A
00E3 FB                 EI                      ;ENABLE INTERRUPTS
00E4 CDEC00     BDSTAT: CALL    STATUS          ;GET STATUS
00E7 CDFA00             CALL    VIDEO           ;SERVICE HOST
00EA 18F8               JR      BDSTAT          ;LOOP
00EC            ;
00EC            ;       I/O PORT STATUS ROUTINE
00EC            ;
00EC            ;       This routine scans the S100 I/O port for
00EC            ;       video characters or commands and checks the
00EC            ;       keyboard buffer counter for new keystrokes.
00EC            ;
00EC DB00       STATUS: IN      A,(S100ST)      ;DATA FROM S100?
00EE 1F                 RRA
00EF D8                 RET     C               ;SERVICE HOST
00F0 3AD211             LD      A,(BCTR)        ;LOAD KEYBD BUFFER COUNT
00F3 FE00               CP      0               ;KEY BUFFER EQ ZERO?
00F5 C4D902             CALL    NZ,KEYOUT       ;SEND CHARACTER TO HOST
00F8 18F2               JR      STATUS          ;LOOP
00FA            ;
00FA            ;       SCREEN SERVICE ROUTINE
00FA            ;
00FA            ;       This routine reads the video character or
00FA            ;       command and takes the appropriate action.
00FA            ;       It also update the cursor and address pointers
00FA            ;       and places attributes if changed.
00FA            ;
00FA DB10       VIDEO:  IN      A,(VDS100)      ;GET CHARACTER
```

```
00FC E67F               AND     7FH                 ;MASK BIT 7
00FE FE00               CP      NULL                ;IGNORE NULL
0100 C8                 RET     Z
0101 FE0D               CP      CR                  ;CHECK FOR CARRAGE RETURN
0103 2864               JR      Z,CRLF
0105 FE0A               CP      LF                  ;CHECK FOR LINE FEED
0107 286E               JR      Z,LIFD
0109 FE1B               CP      ESC                 ;CHECK FOR ESCAPE CODE
010B 2848               JR      Z,ESCAPE
010D FE20               CP      ' '                 ;CHECK FOR CONTROL CHARACTERS
010F 3824               JR      C,CTRAP
0111 47        PLCHAR:  LD      B,A                 ;SAVE CHARATER
0112 3AD911             LD      A,(ATTFLG)          ;IS NEW ATTRIBUTE REQUESTED
0115 FE01               CP      1
0117 CCBF04             CALL    Z,FRATT             ;YES - PUT ON SCREEN
011A 7E                 LD      A,(HL)              ;GET CURRENT CHARACTER
011B FEF0               CP      ELCODE              ;AT EOL?
011D CA6401             JP      Z,WRAP              ;YES - WRAP TO NEXT LINE
0120 E6C0               AND     0C0H                ;STRIP LOW BITS
0122 FE80               CP      80H                 ;IS IT AND ATTRIBUTE
0124 CC9105             CALL    Z,ADJSCR            ;YES ADJUST SCREEN
0127 78                 LD      A,B                 ;GET CHARACTER
0128 77                 LD      (HL),A              ;STORE CHARACTER
0129 7B                 LD      A,E                 ;LOAD CURSOR COLUMN
012A FE4F               CP      79                  ;CURSOR EQ 80?
012C CA6401             JP      Z,WRAP              ;WRAP CURSOR TO NEXT LINE
012F 1C                 INC     E                   ;UPDATE CURSOR
0130 23                 INC     HL                  ;UPDATE POINTER ADDRESS
0131 CDC602             CALL    CURSOR              ;UPDATE CURSOR POSITION
0134 C9                 RET
0135           ;
0135           ;        CONTROL CHARACTER TRAP ROUTINE
0135           ;
0135           ;        This routine determine which control code
0135           ;        has been called and jump to its service routine.
0135           ;        It ignores control codes not implemented.
0135           ;
0135 E5        CTRAP:   PUSH    HL                  ;SAVE REGISTERS
0136 D5                 PUSH    DE
0137 C5                 PUSH    BC
0138 218707             LD      HL,CTLTBL           ;LOAD TOP OF CONTROL TABLE
013B 011C00    FDMAT:   LD      BC,ATTCTR+1         ;LOAD CONTROL TABLE COUNTER
013E EDB1               CPIR                        ;FIND A MATCH
0140 E25101             JP      PO,NOMAT            ;RETURN IF NO MATCH
0143 0D                 DEC     C                   ;SUBTRACT OFFSET
0144 CB01               RLC     C                   ;MULT COUNT X2
0146 21BF07             LD      HL,ATTTBL           ;LOAD CONTROL JMP TABLE
0149 09                 ADD     HL,BC               ;ADD OFFSET
014A 5E                 LD      E,(HL)              ;GET JUMP ADDRESS
014B 23                 INC     HL
014C 56                 LD      D,(HL)
014D EB                 EX      DE,HL               ;PUT ADDRESS IN HL
014E C1                 POP     BC                  ;RESTORE REGISTERS
014F D1                 POP     DE
0150 E9                 JP      (HL)                ;EXECUTE CTL CHARACTER
0151 C1        NOMAT:   POP     BC                  ;RESTORE REGISTERS
0152 D1                 POP     DE
0153 E1                 POP     HL
0154 C9                 RET
0155           ;
0155           ;        ESCAPE TRAP ROUTINE
```

```
0155           ;
0155           ;        This routine looks for a ASCII character after
0155           ;        an ESCAPE and determines the service routine
0155           ;        requested.  It ignores any unused codes.
0155           ;
0155 E5        ESCAPE:  PUSH    HL                  ;SAVE REGISTERS
0156 D5                 PUSH    DE
0157 C5                 PUSH    BC
0158 CDEC00             CALL    STATUS              ;GET NEXT CHARACTER
015B DB10               IN      A,(VDS100)          ;GET CHARACTER
015D E67F               AND     7FH                 ;STRIP PARITY
015F 21A307             LD      HL,ESCTBL           ;LOAD TOP ESC TABLE
0162 18D7               JR      FDMAT               ;FIND THE MATCH
0164           ;
0164           ;        WRAP TO NEXT LINE ROUTINE
0164           ;
0164           ;        This routine puts the cursor on the next
0164           ;        line in column zero.
0164           ;
0164 CD6901    WRAP:    CALL    CRLF                ;DO CARRAGE RETURN
0167 180E               JR      LIFD                ;DO A LINE FEED
0169           ;
0169           ;        CARRAGE RETURN ROUTINE
0169           ;
0169           ;        This routine puts the cursor in column zero
0169           ;        on the same line.
0169           ;
0169 1E00      CRLF:    LD      E,0                 ;SET CURSOR COLUMN TO ZERO
016B CDDF01             CALL    LNADD               ;CALC NEW LINE ADDRESS
016E CDC602             CALL    CURSOR              ;UPDATE CURSOR
0171 3E01               LD      A,1                 ;SET ATTRIBUTE REQUEST FLAG
0173 32D911             LD      (ATTFLG),A
0176 C9                 RET
0177           ;
0177           ;        LINE FEED ROUTINE
0177           ;
0177           ;        This routine moves the cursor to the next line
0177           ;        in the same column.
0177           ;
0177 79        LIFD:    LD      A,C                 ;BOTTOM OF SCREEN?
0178 FE18               CP      24                  ;IF STATUS LINE - IGNORE
017A C8                 RET     Z
017B FE17               CP      23
017D CC8F01             CALL    Z,SCROLL            ;YES THEN SCROLL SCREEN
0180 0C                 INC     C                   ;ADD LINE TO CURSOR
0181 CDDF01    CALLN:   CALL    LNADD               ;CALC NEW LINE ADDRESS
0184 CDF201             CALL    FILL                ;REMOVE ENDDMA IN FRONT OF CURSOR
0187 19                 ADD     HL,DE               ;ADD COLUMN NUMBER
0188 CDBF04             CALL    FRATT               ;CHECK ATTRIBUTES
018B CDC602             CALL    CURSOR              ;MOVE CURSOR
018E C9                 RET
018F           ;
018F           ;        SCROLL ROUTINE
018F           ;
018F           ;        This routine moves the screen up one line
018F           ;        and blanks the last line.
018F           ;
018F 3AD111    SCROLL:  LD      A,(WTCTR)           ;START OF WAIT LOOP
0192 FE00      CHECK:   CP      0                   ;IF ZERO NO WAIT
0194 2809               JR      Z,NOWAIT
0196 0600               LD      B,0
```

```
0198 E3         CHI:    EX      (SP),HL         ;WASTE TIME
0199 E3                 EX      (SP),HL
019A 10FC               DJNZ    CHI             ;LOOP TILL DONE
019C 3D                 DEC     A               ;DEC WAIT COUNT
019D 18F3               JR      CHECK
019F C5         NOWAIT: PUSH    BC              ;SAVE REGISTERS
01A0 D5                 PUSH    DE
01A1 E5                 PUSH    HL
01A2 F3                 DI                      ;DISABLE KEYBOARD
01A3 DB41               IN      A,(CRTC)        ;CLEAR IRQ BIT
01A5 DB41       CRTWT:  IN      A,(CRTC)        ;WAIT TILL VERT RETRACE
01A7 E620               AND     20H             ;MASK IRQ BIT
01A9 28FA               JR      Z,CRTWT
01AB 2A0012             LD      HL,(SCRTBL)
01AE E5                 PUSH    HL
01AF 210212             LD      HL,SCRTBL+2     ;LOAD TOP OF TABLE +2
01B2 110012             LD      DE,SCRTBL       ;LOAD TOP OF TABLE
01B5 012E00             LD      BC,46           ;LOAD COUNTER
01B8 EDB0               LDIR
01BA E1                 POP     HL              ;STORE OLD TOP
01BB 222E12             LD      (SCRTBL+46),HL  ;STORE IT
01BE CBBC               RES     7,H             ;MASK DMA BIT
01C0 3AD711             LD      A,(MMSTUS)      ;CHECK I/O MEM MAP FLAG
01C3 FE00               CP      0               ;IF 0 THEN I/O
01C5 280B               JR      Z,DMAFIL
01C7 0650               LD      B,80            ;LOAD COUNTER WITH 80
01C9 3620       SPALP:  LD      (HL),' '        ;PUT SPACES IN LAST LINE
01CB 23                 INC     HL              ;ADVANCE POINTER
01CC 10FB               DJNZ    SPALP           ;LOOP TILL DONE
01CE 0610               LD      B,16            ;LOAD COUNTER FOR ENDDMA
01D0 1802               JR      BLKLP
01D2 0660       DMAFIL: LD      B,96            ;LOAD COUNTER FOR ENDDMA
01D4 36F1       BLKLP:  LD      (HL),ENDDMA     ;PUT ENDDMA IN LAST LINE
01D6 23                 INC     HL              ;ADVANCE POINTER
01D7 10FB               DJNZ    BLKLP           ;LOOP TILL LINE FILLED
01D9 FB                 EI                      ;ENABLE INTERRUPTS
01DA E1                 POP     HL              ;RESTORE REGISTERS
01DB D1                 POP     DE
01DC C1                 POP     BC
01DD 0D                 DEC     C               ;PRESET CURSOR LINE
01DE C9                 RET
01DF            ;
01DF            ;       CALCULATES NEW LINE ADDRESS
01DF            ;
01DF            ;       This routine calculates the refresh memory
01DF            ;       address of the first character in the line
01DF            ;       designated in register C
01DF            ;
01DF C5         LNADD:  PUSH    BC              ;SAVE REGISTER
01E0 D5                 PUSH    DE
01E1 0600               LD      B,0             ;CLEAR B FOR DOUBLE ADD
01E3 CB01               RLC     C               ;MULTIPLY X2
01E5 210012             LD      HL,SCRTBL       ;LOAD TOP OF TABLE
01E8 09                 ADD     HL,BC           ;ADD OFFSET
01E9 5E                 LD      E,(HL)          ;GET POINTER ADDR
01EA 23                 INC     HL
01EB 56                 LD      D,(HL)
01EC EB                 EX      DE,HL           ;PUT ADDR IN HL
01ED CBBC               RES     7,H             ;REMOVE CRTC CONTROL BIT
01EF D1                 POP     DE              ;RESTORE REGISTER
01F0 C1                 POP     BC
```

```
01F1 C9                 RET
01F2            ;
01F2            ;       FILL ROUTINE
01F2            ;
01F2            ;       This routine removes all of the end of line
01F2            ;       code from in front of the cursor. This is done
01F2            ;       so that new characters will be visable on that
01F2            ;       line.  This routine also counts the invisable
01F2            ;       attributes and adjusts the pointer address.
01F2            ;
01F2 D5         FILL:   PUSH    DE              ;SAVE REGISTERS
01F3 E5                 PUSH    HL
01F4 0600               LD      B,0             ;ZERO ATTRIBUTE COUNTER
01F6 7E         FLOOP:  LD      A,(HL)          ;GET CHARATER
01F7 E6C0               AND     0C0H            ;STRIP 4 LSB
01F9 FEC0               CP      0C0H            ;IS IT A DMA CONTROL
01FB 2002               JR      NZ,NODMA
01FD 3620               LD      (HL),' '        ;YES -  PUT SPACE
01FF 1D         NODMA:  DEC     E               ;DEC COUNTER
0200 FA0C02             JP      M,NOFILL        ;IF NEG ALL DONE
0203 FEB0               CP      80H             ;IS IT AN ATTRIBUTE
0205 2002               JR      NZ,NOTATT
0207 04                 INC     B               ;INC ATTRIBUTE COUNTER
0208 1C                 INC     E               ;INC CHARACTER COUNTER
0209 23         NOTATT: INC     HL              ;ADVANCE POINTER
020A 18EA               JR      FLOOP           ;ANY MORE?
020C E1         NOFILL: POP     HL              ;RESTORE HL
020D 58                 LD      E,B             ;MOVE COUNTER FOR ADD
020E 19                 ADD     HL,DE           ;ADD ATTRIBUTE OFFSET
020F D1                 POP     DE              ;RESTORE DE
0210 C9                 RET
0211            ;
0211            ;       INTIALIZE ALL CRT FIRMWARE
0211            ;
0211            ;       This routine sets-up the CRT controller
0211            ;       prepares the RAM area and initializes the
0211            ;       cursor.
0211            ;
0211            ;
0211            ;       INITIALIZE STATUS LINE & SCREEN
0211            ;
0211 213011     INTST:  LD      HL,TOPSCR+24*98 ;LOAD BEGINNING OF STATUS LINE
0214 0660               LD      B,96            ;LOAD LINE COUNT
0216 3620       STINLP: LD      (HL),' '        ;PUT SPACES
0218 23                 INC     HL              ;ADVANCE POINTER
0219 10FB               DJNZ    STINLP
021B 36F0               LD      (HL),ELCODE     ;LOAD EOL CODE
021D 216B11             LD      HL,TOPSCR+(24*98)+79-(PARTBL-LOGON)
0220 117602             LD      DE,LOGON
0223 1A         MSG:    LD      A,(DE)          ;PUT LOGON MESSAGE
0224 FE00               CP      0               ;IF ZERO STOP
0226 2805               JR      Z,INTCRT
0228 77                 LD      (HL),A          ;PUT ON SCREEN
0229 23                 INC     HL              ;ADVANCE POINTERS
022A 13                 INC     DE
022B 18F6               JR      MSG
022D            ;
022D 3EA0       INTCRT: LD      A,CRTEI         ;ENABLE CRT INTERRUPT
022F D341               OUT     (CRTC),A
0231 FD210012           LD      IY,SCRTBL       ;LOAD TOP OF TABLE
0235            ;
```

```
0235 01D0F6   CLS:    LD      BC,-98*24       ;LOAD COUNTER
0238 210008           LD      HL,TOPSCR       ;LOAD TOP A
023B 36F1     INTLP:  LD      (HL),ENDDMA     ;FILL MEMORY WITH END DMA
023D 23               INC     HL
023E 03               INC     BC              ;COUNTER EQ ZERO?
023F CB78             BIT     7,B
0241 20F8             JR      NZ,INTLP
0243 0618             LD      B,24            ;LOAD COUNTER
0245 216008           LD      HL,TOPSCR+96    ;POINT AT END OF LINE
0248 116200           LD      DE,98           ;LOAD OFFSET PER LINE
024B 36F0     ATTLOP: LD      (HL),ELCODE     ;DENOTES END OF LINE
024D 19               ADD     HL,DE           ;ADD OFFSET FOR NEXT LINE
024E 10FB             DJNZ    ATTLOP          ;LOOP TILL 24 LINES DONE
0250          ;
0250 013800           LD      BC,DLEND-ISCTBL ;INITIALIZE SCRTBL
0253 110012           LD      DE.SCRTBL
0256 218E02           LD      HL,ISCTBL
0259 EDB0             LDIR                    ;MOVE IT TO RAM
025B          ;
025B 2A0012           LD      HL,(SCRTBL)     ;GET FIRST LINE ADDRESS
025E CBBC             RES     7,H             ;RESET DMA BIT
0260 3E80             LD      A,80H           ;CLEAR ATTRIBUTE BYTE
0262 32D811           LD      (ATTBUF),A
0265          ;
0265 0E00             LD      C,0             ;INITIALIZE REGISTERS
0267 51               LD      D,C
0268 41               LD      B.C
0269 59               LD      E.C
026A CDC602           CALL    CURSOR          ;INITIALIZE CURSOR
026D          ;
026D 3AD711           LD      A,(MMSTUS)      ;IF MEM MAP ON FILL SPACES
0270 FE00             CP      0
0272 C4F706           CALL    NZ.MMFILL       ;YES - PUT SPACES
0275 C9               RET
0276          ;
0276          ;       CRT SCREEN PARAMETER TABLE
0276          ;
0276 81       LOGON:  DEFB 81H
0277 5644422D41322056657220342E30
                      DEFM 'VDB-A2 Ver 4.0'
0285 8020F32000
                      DEFB 80H,20H,0F3H,20H,00H
028A 4F       PARTBL: DEFB HCR                ;SCREEN CONTROL PARAMETERS
028B 58               DEFB VVVR
028C 79               DEFB ULL
028D 09               DEFB MFCH
028E          ;
028E 0088     ISCTBL: DEFW TOPSC8             ;LINE TABLE FOR SCSEEN REFRESH
0290 6288             DEFW TOPSC8+98
0292 C488             DEFW TOPSC8+98*2
0294 2689             DEFW TOPSC8+98*3
0296 8889             DEFW TOPSC8+98*4
0298 EA89             DEFW TOPSC8+98*5
029A 4C8A             DEFW TOPSC8+98*6
029C AE8A             DEFW TOPSC8+98*7
029E 108B             DEFW TOPSC8+98*8
02A0 728B             DEFW TOPSC8+98*9
02A2 D48B             DEFW TOPSC8+98*10
02A4 368C             DEFW TOPSC8+98*11
02A6 988C             DEFW TOPSC8+98*12
02A8 FA8C             DEFW TOPSC8+98*13
```

```
02AA 5C8D             DEFW TOPSC8+98*14
02AC BE8D             DEFW TOPSC8+98*15
02AE 208E             DEFW TOPSC8+98*16
02B0 828E             DEFW TOPSC8+98*17
02B2 E48E             DEFW TOPSC8+98*18
02B4 468F             DEFW TOPSC8+98*19
02B6 A88F             DEFW TOPSC8+98*20
02B8 0A90             DEFW TOPSC8+98*21
02BA 6C90             DEFW TOPSC8+98*22
02BC CE90             DEFW TOPSC8+98*23
02BE 3091             DEFW TOPSC8+98*24
02C0 0000             DEFW 0000
02C2 220000   WDMOVE: LD      (DUMMY).HL      ;SWAP LINES IN TABLE
02C5 C9               RET                     ; TO PREVENT FLASHING
02C6          DLEND:
02C6          RWDMOV  EQU     WDMOVE-ISCTBL+SCRTBL ;ADDRESS OF WORD LOAD
02C6          ;
02C6          ;       MOVE CURSOR ROUTINE
02C6          ;
02C6          ;       This routine moves the cursor on the srceen
02C6          ;       based on the data in registers C & E.
02C6          ;
02C6 3E80     CURSOR: LD      A,CURPOS        ;SEND CONTROL CHAR
02C8 D341             OUT     (CRTC),A
02CA 7B               LD      A,E             ;SEND COLUMN
02CB D340             OUT     (CRTD),A
02CD 79               LD      A,C             ;SEND LINE
02CE D340             OUT     (CRTD),A
02D0 7E               LD      A,(HL)          ;SEE IF DMA
02D1 E6F0             AND     0F0H            ;STRIP LSB'S
02D3 FEF0             CP      0F0H
02D5 C0               RET     NZ              ;NO RETURN
02D6 3620             LD      (HL),' '        ;PUT SPACE SO CURSAR WILL APPEAR
02D8 C9               RET
02D9          ;
02D9          ;       KEYBOARD BUFFER TO S100 SERVICE ROUTINE
02D9          ;
02D9          ;       This routine checks the S100 keyboard port
02D9          ;       status and return if busy or loads the port
02D9          ;       with the next character in the keyboard
02D9          ;       buffer.
02D9          ;
02D9 DB20     KEYOUT: IN      A,(KYS100)      ;CHECK STATUS
02DB 1F               RRA                     ;NOT READY RETURN
02DC D8               RET     C
02DD E5               PUSH    HL              ;SAVE HL
02DE 2AD511           LD      HL,(KYOTA)      ;GET OUTPUT BUFFER ADDR
02E1 DD3500           DEC     (IX+0)          ;DEC COUNTER
02E4 7D               LD      A,L             ;CHECK ADDR AT BOTTOM
02E5 FED0             CP      BOTKEY
02E7 2003             JR      NZ,NOBOT1       ;NO THEN SKIP
02E9 21A011           LD      HL,TOPKEY       ;POINT TO TOP
02EC 7E       NOBOT1: LD      A,(HL)          ;GET CHARACTER
02ED D320             OUT     (KYS100),A      ;SEND TO HOST
02EF 23               INC     HL              ;INC POINTER
02F0 22D511           LD      (KYOTA).HL      ;SAVE POINTER
02F3 E1               POP     HL              ;RESTORE HL
02F4 C9               RET
02F5          ;
02F5          ;       ATTRIBUTE CONTROL ROUTINES
02F5          ;
```

D3
20

```
02F5                ;
02F5                ;       HOME ROUTINE
02F5                ;
02F5                ;       This routine clears the screen and puts the
02F5                ;       cursor at the home postion.
02F5                ;
02F5 E1        HOME:    POP     HL              ;RESTORE REGISTER
02F6 C33502             JP      CLS             ;CLEAR SCREEN
02F9                ;
02F9                ;       SET CURSOR POSITION ROUTINE
02F9                ;
02F9                ;       This routine takes the next two byte from the
02F9                ;       S100 video port and moves the cursor according
02F9                ;       to the values.
02F9                ;
02F9 E1        STCUR:   POP     HL              ;RESTORE REGISTER
02FA D5                 PUSH    DE              ;SAVE OLD CURSOR POSITION
02FB C5                 PUSH    BC
02FC CDEC00             CALL    STATUS          ;WAIT FOR CURSOR LINE
02FF DB10               IN      A,(VDS100)      ;GET CURSOR LINE
0301 E67F               AND     7FH             ;STRIP PARITY
0303 D620               SUB     OFFSET          ;REMOVE OFFSET
0305 FE19               CP      25              ;IF > MAX,MAKE MAX
0307 0E18               LD      C,24
0309 F20D03             JP      P,CURNXT
030C 4F                 LD      C,A             ;STORE IN C
030D CDEC00    CURNXT:  CALL    STATUS          ;WAIT FOR CURSOR COLUMN
0310 DB10               IN      A,(VDS100)      ;GET CURSOR COLUMN
0312 E67F               AND     7FH             ;STRIP PARITY
0314 D620               SUB     OFFSET          ;REMOVE OFFSET
0316 FE50               CP      80              ;IF > MAX MAKE MAX
0318 1E4F               LD      E,79
031A F21E03             JP      P,CKCUR
031D 5F                 LD      E,A             ;STORE IN E
031E 79        CKCUR:   LD      A,C             ;IF STATUS LINE SAVE OLD POSITION
031F FE18               CP      24
0321 200C               JR      NZ,NO25         ;NO - DO NOTHING
0323 E1                 POP     HL              ;GET OLD LINE
0324 7D                 LD      A,L             ;SAVE IT
0325 32DD11             LD      (SAVLN),A
0328 E1                 POP     HL              ;GET OLD COLUMN
0329 7D                 LD      A,L
032A 32DE11             LD      (SAVCOL),A      ;SAVE IT
032D 1802               JR      PLCUR
032F E1        NO25:    POP     HL              ;WASTE OLD CURSAR POSITION
0330 E1                 POP     HL
0331 CDDF01    PLCUR:   CALL    LNADD           ;CALC LINE ADDRESS
0334 CDF201             CALL    FILL            ;REMOVE DMA CONTROLS
0337 19                 ADD     HL,DE           ;ADD COLUMN TO ADDRESS
0338 3E01               LD      A,1             ;SET ATTRIBUTE REQUEST FLAG
033A 32D911             LD      (ATTFLG),A
033D CDC602             CALL    CURSOR          ;MOVE CURSOR
0340 C9                 RET
0341                ;
0341                ;       CURSOR UP ROUTINE
0341                ;
0341                ;       This routine moves the cursor up one line.
0341                ;       if the line is more than the minimum value.
0341                ;
0341 E1        CURUP:   POP     HL              ;RESTORE REGISTER
0342 79                 LD      A,C             ;AT FIRST LINE?
```

```
0343 FE00               CP      0
0345 C8                 RET     Z               ;YES - DO NOTHING
0346 0D                 DEC     C               ;SUBTRACT ONE FROM LINE
0347 18E8               JR      PLCUR           ;PLACE CURSOR
0349                ;
0349                ;       CURSOR DOWN ROUTINE
0349                ;
0349                ;       This routine moves the cursor down one line
0349                ;       if the line is less than maximum value.
0349                ;
0349 E1        CURDN:   POP     HL              ;RESTORE REGISTER
034A 79                 LD      A,C             ;AT BOTTOM LINE
034B FE18               CP      24
034D C8                 RET     Z               ;YES - DO NOTHING
034E FE17               CP      23
0350 C8                 RET     Z
0351 0C                 INC     C               ;ADD ONE TO LINE
0352 18DD               JR      PLCUR           ;PLACE CURSOR
0354                ;
0354                ;       CURSOR RIGHT ROUTINE
0354                ;
0354                ;       This routine moves the cursor right one
0354                ;       column if the column is less than the
0354                ;       maximum value.
0354                ;
0354 E1        CURRT:   POP     HL              ;RESTORE REGISTER
0355 7B                 LD      A,E             ;AT END OF LINE?
0356 FE4F               CP      79
0358 C8                 RET     Z               ;YES - DO NOTHING
0359 1C                 INC     E               ;ADD ONE TO COLUMN
035A 18D5               JR      PLCUR           ;PLACE CURSOR
035C                ;
035C                ;       CURSOR LEFT ROUTINE
035C                ;
035C                ;       This routine move the cursor left one column
035C                ;       if the column is greater than the minimum value.
035C                ;
035C E1        CURLT:   POP     HL              ;RESTORE REGISTER
035D 7B                 LD      A,E             ;AT BEGINING OF LINE?
035E FE00               CP      0
0360 C8                 RET     Z               ;YES DO NOTHING
0361 1D                 DEC     E               ;SUBTRACT ONE FROM COLUMN
0362 18CD               JR      PLCUR           ;PLACE CURSOR
0364                ;
0364                ;       ERASE TO END OF LINE ROUTINE
0364                ;
0364                ;       This routine erases the line from the cursor postion
0364                ;       to the end of the line.
0364                ;
0364 E1        EOL:     POP     HL              ;RESTORE REGISTER
0365 E5        EOL1:    PUSH    HL              ;SAVE REGISTERS
0366 D5                 PUSH    DE
0367 CD5406             CALL    PRVATT          ;FIND PREVIOUS ATTRIBUTE
036A 47                 LD      B,A             ;TEMP STORE
036B 7E                 LD      A,(HL)          ;GET CHARACTER
036C FEF0               CP      ELCODE          ;AT EOL?
036E 282F               JR      Z,EOLEND        ;YES - DO NOTHING
0370 FE80               CP      80H             ;OFF ATTRIBUTE BYTE
0372 200A               JR      NZ,EOLATT       ;NO - PUT SPACE
0374 23                 INC     HL              ;ADVANCE POINTER
0375 7E                 LD      A,(HL)          ;GET CHARACTER
```

```
0376 FEF0              CP      ELCODE          ;AT EOL?
0378 2825              JR      Z,EOLEND        ;YES - DO NOTHING
037A 3620              LD      (HL),' '        ;PUT SPACE
037C 1817              JR      EOLLP           ;ERASE REST OF LINE
037E 3620     EOLATT:  LD      (HL),' '        ;PUT SPACE
0380 23                INC     HL              ;ADVANCE POINTER
0381 7E                LD      A,(HL)          ;GET CHARACTER
0382 FEF0              CP      ELCODE          ;AT EOL?
0384 2819              JR      Z,EOLEND        ;YES - DO NOTHING
0386 3E80              LD      A,80H           ;LOAD OFF ATTRIBUTE BYTE
0388 B8                CP      B               ;SAME AS PREVIOUS ATTRIBUTE
0389 2002              JR      NZ,EOLSP        ;NO - PUT OFF BYTE
038B 3EF1              LD      A,ENDDMA
038D 77       EOLSP:   LD      (HL),A          ;PUT ON SCREEN
038E 23                INC     HL              ;ADVANCE POINTER
038F FEF0              CP      ELCODE          ;AT EOL?
0391 280C              JR      Z,EOLEND        ;YES - DO NOTHING
0393 3620              LD      (HL),' '        ;PUT SPACE
0395 23       EOLLP:   INC     HL              ;ADVANCE POINTER
0396 7E                LD      A,(HL)          ;GET CHARACTER
0397 FEF0              CP      ELCODE          ;END OF LINE?
0399 2804              JR      Z,EOLEND        ;YES - STOP
039B 36F1     EOLOVR:  LD      (HL),ENDDMA     ;PUT DMA CODE
039D 18F6              JR      EOLLP           ;NEXT CHARACTER
039F 3AD711   EOLEND:  LD      A,(MMSTUS)      ;IF MEM MAP ON FILL WITH SPACES
03A2 FE00              CP      0
03A4 C4F706            CALL    NZ,MMFILL       ;YES - PUT SPACES
03A7 D1                POP     DE              ;RESTORE REGISTERS
03A8 E1                POP     HL
03A9 C9                RET
03AA          ;
03AA          ;        ERASE TO END OF SCREEN ROUTINE
03AA          ;
03AA          ;        This routine erases the screen from the
03AA          ;        current cursor position.
03AA          ;
03AA E1       EOS:     POP     HL              ;RESTORE REGISTER
03AB CD6503            CALL    EOL1            ;ERASE TO END OF LINE
03AE 79                LD      A,C             ;IF STATUS LINE RETURN
03AF FE18              CP      24
03B1 C8                RET     Z
03B2 E5                PUSH    HL              ;SAVE REGISTERS
03B3 D5                PUSH    DE
03B4 C5                PUSH    BC
03B5 0660     ENL:     LD      B,96            ;LOAD LINE COUNT
03B7 0C                INC     C               ;ADVANCE LINE
03B8 CDDF01            CALL    LNADD           ;FIND START OF LINE
03BB 36F1     EOSLP:   LD      (HL),ENDDMA     ;STORE CONTROL CHARACTER
03BD 23                INC     HL              ;INC ADDRESS
03BE 10FB              DJNZ    EOSLP           ;LOOP TILL DONE
03C0 3E17              LD      A,23            ;LAST LINE?
03C2 B9                CP      C
03C3 20F0              JR      NZ,ENL
03C5 3AD711            LD      A,(MMSTUS)      ;IF MEM MAP ON FILL WITH SPACES
03C8 FE00              CP      0
03CA C1                POP     BC              ;GET REGISTER FOR MOVE
03CB C5                PUSH    BC              ;RESAVE
03CC C4F706            CALL    NZ,MMFILL       ;YES - PUT SPACES
03CF C1                POP     BC              ;RESTORE REGISTERS
03D0 D1                POP     DE
03D1 E1                POP     HL
```

```
03D2 C9                RET
03D3          ;
03D3          ;        DELETE LINE
03D3          ;
03D3          ;        this routine deletes the line the cursor is on
03D3          ;        and scrolls all the remaining line under it up one
03D3          ;
03D3 E1       DELLN:   POP     HL                          ;RESTORE REGISTER
03D4 C5                PUSH    BC                          ;SAVE REGISTERS
03D5 D5                PUSH    DE
03D6 E5                PUSH    HL
03D7 3ACE10            LD      A,(TOPSCR+98*23)            ;REMOVE SYNC BYTE
03DA FEF3              CP      STPDMA
03DC 2005              JR      NZ,OVRSTP
03DE 3EF1              LD      A,ENDDMA
03E0 32CE10            LD      (TOPSCR+98*23),A
03E3 210012   OVRSTP:  LD      HL,SCRTBL       ;LOAD TOP OF TABLE
03E6 0600              LD      B,0             ;CLEAR B FOR ADD
03E8 CB01              RLC     C               ;MULTIPLY LINE # BY 2
03EA 09                ADD     HL,BC           ;ADD OFFSET
03EB 5D                LD      E,L             ;MOVE TO DE
03EC 54                LD      D,H
03ED 7E                LD      A,(HL)          ;GET ADDRES OF DELETED LINE
03EE 32DB11            LD      (TEMP2),A       ;STORE IT
03F1 23                INC     HL
03F2 7E                LD      A,(HL)          ;GET SECOND BYTE
03F3 32DC11            LD      (TEMP2+1),A
03F6 23                INC     HL
03F7 3E2E              LD      A,46            ;CALCULATE COUNT
03F9 91                SUB     C               ;SUBSTRACT CURRENT LINE
03FA 2808              JR      Z,DNOMOV        ;IF ZERO DON'T MOVE
03FC 4F                LD      C,A             ;MOVE TO C
03FD          ;
03FD DB41              IN      A,(CRTC)        ;LOCK FOR END OF FRAME
03FF DB41     DCRTWT:  IN      A,(CRTC)        ;GET IRQ BIT
0401 E620              AND     20H             ;MASK FOR IRQ BIT
0403 28FA              JR      Z,DCRTWT        ;NO - WAIT FOR IT
0405          ;
0405 EDB0              LDIR                    ;MOVE TABLE
0407 2ADB11   DNOMOV:  LD      HL,(TEMP2)
040A 222E12            LD      (SCRTBL+46),HL  ;PUT THIS ADDRESS AT BOTTOM
040D CBBC     LNFIL:   RES     7,H             ;MASK DMA BIT
040F 3AD711            LD      A,(MMSTUS)      ;CHECK I/O MEM MAP FLAG
0412 FE00              CP      0
0414 2808              JR      Z,DFIL          ;IF I/O FILL WITH DMA
0416 0650              LD      B,80            ;LOAD COUNT
0418 3620     DSPLP:   LD      (HL),' '        ;PUT SPACE ON SCREEN
041A 23                INC     HL              ;ADVANCE POINTER
041B 10FB              DJNZ    DSPLP           ;LOOP TILL DONE
041D 0610              LD      B,16            ;LOAD CTR FOR DMA BYTES
041F 1802              JR      DBLKLP
0421 0660     DFIL:    LD      B,96            ;LOAD LINE COUNT
0423 36F1     DBLKLP:  LD      (HL),ENDDMA     ;PUT DMA ON SCREEN
0425 23                INC     HL              ;ADVANCE POINTER
0426 10FB              DJNZ    DBLKLP          ;LOOP TILL DONE
0428 E1                POP     HL
0429 D1                POP     DE
042A C1                POP     BC
042B C38101            JP      CALLN
042E          ;
042E          ;        INSERT LINE
```

```
042E            ;
042E            ;       This routine inserts a line at the cursar position
042E            ;       and moves the lines below down one.
042E            ;
042E E1         INLN:   POP     HL              ;RESTORE REGISTER
042F C5                 PUSH    BC              ;SAVE REGISTERS
0430 D5                 PUSH    DE
0431 E5                 PUSH    HL
0432 210012             LD      HL,SCRTBL       ;LOAD TOP OF TABLE
0435 0600               LD      B,0             ;CLEAR B FOR ADD
0437 CD01               RLC     C               ;MULTIPLY LINE # BY 2
0439 09                 ADD     HL,BC           ;ADD OFFSET
043A E5                 PUSH    HL              ;SAVE ADDRESS
043B 2A2F12             LD      HL,(SCRTBL+46)  ;LOAD BOTTOM OF TABLE
043E E5                 PUSH    HL              ;SAVE ADDRESS
043F 3E2E               LD      A,46            ;LOAD TOTAL COUNT
0441 91                 SUB     C               ;GET OFFSET
0442 2811               JR      Z,NOMOVE        ;IF ZERO DON'T MOVE LINES
0444 4F                 LD      C,A             ;LOAD COUNT
0445 212D12             LD      HL,SCRTBL+45    ;LOAD ADRESSES FOR MOVE
0448 112F12             LD      DE,SCRTBL+47
044B            ;
044B DB41               IN      A,(CRTC)        ;LOOK FOR END OF FRAME
044D DB41       ICRTWT: IN      A,(CRTC)        ;GET IRQ BIT
044F E620               AND     20H             ;MASK FOR IRQ BIT
0451 28FA               JR      Z,ICRTWT        ;NO - WAIT FOR IT
0453            ;
0453 EDB8               LDDR                    ;MOVE TABLE
0455 D1         NOMOVE: POP     DE              ;GET ADDRESSES
0456 E1                 POP     HL
0457 73                 LD      (HL),E          ;PUT BACK IN TABLE
0458 23                 INC     HL
0459 72                 LD      (HL),D
045A EB                 EX      DE,HL           ;GET ADDRESS IN HL
045B 18B0               JR      LNFIL           ;FILL IN BLANKED LINE
045D            ;
045D            ;       HOME UP
045D            ;
045D            ;       This routine puts the cursor in the home
045D            ;       postion leaving the screen unchanged.
045D            ;
045D E1         HOMUP:  POP     HL              ;RESTORE REGISTER
045E 0E00               LD      C,0             ;ZERO POSITIONS
0460 59                 LD      E,C
0461 C33103             JP      PLCUR           ;PLACE CURSOR
0464            ;
0464            ;       BELL ROUTINE
0464            ;
0464            ;       This routine activates the bell circuits.
0464            ;
0464 E1         BELL:   POP     HL              ;RESTORE REGISTER
0465 D350               OUT     (SPEAK),A       ;RING BELL
0467 C9                 RET
0468            ;
0468            ;       INTEL 8275 ATTRIBUTE ROUTINES
0468            ;
0468            ;       The 8275 use a byte (10URCCBH) to control
0468            ;       screen attribute functions, where U is
0468            ;       underline, R is reverse video, C is the alternate
0468            ;       character generator, B is blink, and H is highlight.
0468            ;       The routines below toggle the function bits when
0468            ;       called.  There can be only 16 attribute bytes per
0468            ;       line, but consecutive functions are stored in the same
0468            ;       byte.
0468            ;
0468            ;       HIGHLITE ROUTINE
0468            ;
0468 E1         HLITE:  POP     HL              ;RESTORE REGISTER
0469 3AD811             LD      A,(ATTBUF)      ;GET ATTRIBUTE STATE
046C CBC7               SET     0,A             ;SET BIT
046E 1846               JR      ATTFSH          ;PLACE ATTRIBUTE
0470            ;
0470 E1         NHLIT:  POP     HL              ;RESTORE REGISTER
0471 3AD811             LD      A,(ATTBUF)      ;GET ATTRIBUTE STATE
0474 CB87               RES     0,A             ;RESET BIT
0476 183E               JR      ATTFSH          ;PLACE ATTRIBUTE
0478            ;
0478            ;       HIGHLITE TOGGLE
0478            ;
0478 E1         HTOG:   POP     HL              ;RESTORE REGISTER
0479 3AD811             LD      A,(ATTBUF)      ;GET ATTRIBUTE STATE
047C CB47               BIT     0,A             ;TEST BIT
047E 2004               JR      NZ,RESETH
0480 CBC7               SET     0,A             ;SET BIT
0482 1832               JR      ATTFSH          ;PLACE ATTRIBUTE
0484 CB87       RESETH: RES     0,A             ;RESET BIT
0486 182E               JR      ATTFSH          ;PLACE ATTRIBUTE
0488            ;
0488            ;       REVERSE VIDEO ROUTINE
0488            ;
0488 E1         REVID:  POP     HL              ;RESTORE REGISTER
0489 3AD811             LD      A,(ATTBUF)      ;GET ATTRIBUTE STATE
048C CBE7               SET     4,A             ;SET BITS
048E 1826               JR      ATTFSH          ;PLACE ATTRIBUTE
0490            ;
0490 E1         NRVID:  POP     HL              ;RESTORE REGISTER
0491 3AD811             LD      A,(ATTBUF)      ;GET ATTRIBUTE STATE
0494 CBA7               RES     4,A             ;RESET BITS
0496 181E               JR      ATTFSH          ;PLACE ATTRIBUTE
0498            ;
0498            ;       BLINK ROUTINE
0498            ;
0498 E1         BLINK:  POP     HL              ;RESTORE REGISTER
0499 3AD811             LD      A,(ATTBUF)      ;GET ATTRIBUTE STATE
049C CBCF               SET     1,A             ;SET BIT
049E 1816               JR      ATTFSH          ;PLACE ATTRIBUTE
04A0            ;
04A0 E1         NBLNK:  POP     HL              ;RESTORE REGISTER
04A1 3AD811             LD      A,(ATTBUF)      ;GET ATTRIBUTE STATE
04A4 CB8F               RES     1,A             ;RESET BIT
04A6 180E               JR      ATTFSH          ;PLACE ATTRIBUTE
04A8            ;
04A8            ;       CHANGE CHARACTER GENERATOR ROM
04A8            ;
04A8 E1         ALTCH:  POP     HL              ;RESTORE REGISTER
04A9 3AD811             LD      A,(ATTBUF)      ;GET ATTRIBUTE STATE
04AC CB57               BIT     2,A             ;TEST BIT
04AE 2004               JR      NZ,RESETA
04B0 CBD7               SET     2,A             ;SET BIT
04B2 1802               JR      ATTFSH          ;PLACE ATTRIBUTE
04B4 CB97       RESETA: RES     2,A             ;RESET BIT
04B6 32D811     ATTFSH: LD      (ATTBUF),A      ;SAVE CURRENT ATTRIBUTE
```

```
04B9 3E01               LD      A,1
04BB 32D911             LD      (ATTFLG),A      ;SET ATTRIBUTE REQUEST FLAG
04BE C9                 RET
04BF            ;
04BF            ;       PLACE ATTRIBUTE ROUTINE
04BF            ;
04BF            ;       This routine places the attribute byte in
04BF            ;       refresh memory. but merges it with the last
04BF            ;       character if it was an attribute.
04BF            ;       Plus alot of house cleaning.
04BF            ;
04BF AF         PRATT:  XOR     A               ;CLEAR A
04C0 32D911             LD      (ATTFLG),A      ;ZERO ATTRIBUTE REQUEST FLAG
04C3            ;
04C3 CD5406             CALL    PRVATT          ;GET OLD ATTRIBUTE
04C6 32DA11             LD      (TEMP),A        ;STORE IT
04C9            ;
04C9 C5                 PUSH    BC              ;SAVE REGISTER
04CA D5                 PUSH    DE
04CB CD2806             CALL    TMPSCN          ;SWAP LINE ADDRESS
04CE            ;
04CE 2B                 DEC     HL              ;CHECK TO SEE IF LAST
04CF 7E                 LD      A,(HL)          ;GET CHARACTER
04D0 E6C0               AND     0C0H            ;MASK BYTE
04D2 FE80               CP      80H             ;ATTRIBUTE?
04D4 2843               JR      Z,MERGE         ;YES - WRITE OVER
04D6 23                 INC     HL
04D7 7E                 LD      A,(HL)          ;GET CHARACTER
04D8 FEF0               CP      ELCODE          ;AT EOL?
04DA CA8605             JP      Z,ATDONE        ;YES - DO NOTHING
04DD E6C0               AND     0C0H            ;STRIP LOW BITS
04DF FEC0               CP      0C0H            ;DMA CHARACTER?
04E1 2877               JR      Z,ATTSF         ;YES - WRITE OVER
04E3 FE80               CP      80H             ;ATTRIBUTE BYTE?
04E5 2832               JR      Z,MERGE         ;YES - WRITE OVER
04E7 3ADA11             LD      A,(TEMP)        ;GET OLD ATTRIBUTE
04EA 47                 LD      B,A             ;PUT IN B
04EB 3AD811             LD      A,(ATTBUF)      ;IF SAME AS OLD ATTRIBUTE
04EE BB                 CP      B               ; DO NOTHING
04EF CA8605             JP      Z,ATDONE
04F2 CD0A06             CALL    MVCHRT          ;IS CHARACTER ADJUST SCREEN RIGHT
04F5 3AD811             LD      A,(ATTBUF)      ;STORE ATTRIBUTE ON SCREEN
04F8 77                 LD      (HL),A
04F9            ;
04F9 23         ATTBCK: INC     HL              ;POINT AT RIGHT SIDE OF CHARACTER
04FA 7E                 LD      A,(HL)          ;AT EOL?
04FB FEF0               CP      ELCODE
04FD CA8605             JP      Z,ATDONE        ;YES -  STOP
0500 23                 INC     HL
0501 7E                 LD      A,(HL)          ;GET CHARACTER
0502 FEF0               CP      ELCODE          ;AT EOL?
0504 CA1605             JP      Z,ATONE         ;YES - STOP
0507 E6C0               AND     0C0H            ;STRIP LOW BITS
0509 FE80               CP      80H             ;ATTRIBUTE BYTE?
050B 283C               JR      Z,ATTSAM
050D FEC0               CP      0C0H            ;DMA BYTE?
050F 282A               JR      Z,DMA1
0511 CD0A06             CALL    MVCHRT          ;MOVE LINE RIGHT
0514 78                 LD      A,B             ;GET OLD ATTRIBUTE
0515 77                 LD      (HL),A          ;PUT ON SCREEN
0516 2B         ATONE:  DEC     HL              ;POINT AT CHARACTER
0517 186D               JR      ATDONE
0519            ;
0519 7E         MERGE:  LD      A,(HL)          ;GET OLD ATTRIBUTE
051A 47                 LD      B,A             ;PUT IN B
051B 3AD811             LD      A,(ATTBUF)      ;GET NEW ATRIBUTE
051E B8                 CP      B               ;SAME AS OLD?
051F 2817               JR      Z,NOCHG
0521 AF                 XOR     A               ;CLEAR A
0522 BB                 CP      E               ;AT FRONT OF LINE?
0523 280D               JR      Z,NOAT2         ;YES- WRITE OVER ATTRIBUTE
0525 2B                 DEC     HL              ;LOOK FOR NEXT ATTRIBUTE
0526 CD7606             CALL    PRVAT2          ;FIND NEXT PREVIOUS ATTRIBUTE
0529 23                 INC     HL              ;RESTORE POINTER
052A 5F                 LD      E,A             ;PUT IN E
052B 3AD811             LD      A,(ATTBUF)      ;GET NEW ATTRIBUTE
052E BB                 CP      E               ;IS SAME ?
052F CA8605             JP      Z,ATDONE        ;YES - DO NOTHING
0532 3AD811     NOAT2:  LD      A,(ATTBUF)      ;GET NEW ATTRIBUTE
0535 77                 LD      (HL),A          ;WRITE OVER OLD ATTRIBUTE
0536 18C1               JR      ATTBCK          ;LOOK AT BACK SIDE OF CHARACTER
0538 23         NOCHG:  INC     HL              ;POINT TO NEXT CHARACTER
0539 184B               JR      ATDONE
053B            ;
053B 3680       DMA1:   LD      (HL),80H        ;WRITE OVER DMA
053D 23                 INC     HL              ;PLACE SPACE
053E 7E                 LD      A,(HL)          ;GET CHARACTER
053F FEF0               CP      ELCODE          ;AT EOL?
0541 2802               JR      Z,DMAOVR        ;YES - STOP
0543 3620               LD      (HL),' '        ; ON SCREEN
0545 2B         DMAOVR: DEC     HL              ;POINT TO CHARACTER
0546 2B                 DEC     HL
0547 183D               JR      ATDONE
0549            ;
0549 7E         ATTSAM: LD      A,(HL)          ;GET ATTRIBUTE
054A 47                 LD      B,A             ;PUT IN B
054B 3AD811             LD      A,(ATTBUF)      ;GET NEW ATTRIBUTE
054E B8                 CP      B               ;IS IT THE SAME?
054F 2006               JR      NZ,ATTNMT       ;NO - DO NOTHING
0551 23                 INC     HL              ;SET POINTER FOR MOVE
0552 CDEC05             CALL    MOVSCR          ;MOVE LINE LEFT
0555 182F               JR      ATDONE
0557 2B         ATTNMT: DEC     HL              ;POINT TO CHARACTER
0558 182C               JR      ATDONE
055A            ;
055A 3ADA11     ATTSF:  LD      A,(TEMP)        ;GET OLD ATTRIBUTE
055D 47                 LD      B,A             ;PUT IN B
055E 3AD811             LD      A,(ATTBUF)      ;GET NEW ATTRIBUTE
0561 B8                 CP      B               ;IF SAME AS OLD ATTRIBUTE
0562 2822               JR      Z,ATDONE        ; DO NOTHING
0564 77                 LD      (HL),A          ;PUT NEW ATTRIBUTE ON SCREEN
0565 23                 INC     HL              ;MOVE POINTER
0566 7E                 LD      A,(HL)          ;GET CHARACTER
0567 FEF0               CP      ELCODE          ;AT EOL?
0569 281B               JR      Z,ATDONE        ;YES - STOP
056B 3620               LD      (HL),' '        ;WRITE OVER DMA BYTE
056D 3AD811             LD      A,(ATTBUF)      ;GET NEW ATTRIBUTE
0570 FE80               CP      80H             ;IF NEW ATT TURNS ALL OFF
0572 2812               JR      Z,ATDONE        ; STOP
0574 23                 INC     HL              ;MOVE POINTER
0575 7E                 LD      A,(HL)          ;GET CHARACTER
0576 FEF0               CP      ELCODE          ;AT EOL?
```

```
0578 280B           JR     Z,SPOVR        ;YES - STOP
057A 3680           LD     (HL),80H       ;PUT NO ATT BYTE ON SCREEN
057C 23             INC    HL             ;MOVE POINTER
057D 7E             LD     A,(HL)         ;GET CHARACTER
057E FEF0           CP     ELCODE         ;AT END OF LINE?
0580 2802           JR     Z,OVRSP        ;YES - THEN STOP
0582 3620           LD     (HL),' '       ;OR PUT SPACE BEHIND ATTRIBUTE
0584 2B     OVRSP:  DEC    HL             ;POINT AT NEXT CHARACTER
0585 2B     SPOVR:  DEC    HL
0586        ;
0586 E5     ATDONE: PUSH   HL
0587 2ADB11         LD     HL,(TEMP2)     ;GET LINE ADDRESS
058A CD3412         CALL   RWDMOV         ;SWAP SCAN LINE IN TABLE
058D E1             POP    HL             ;RESTORE REGISTERS
058E D1     ATTFIN: POP    DE
058F C1             POP    BC
0590 C9             RET
0591        ;
0591        ;       ADJUST SCREEN
0591        ;       This routine prevents the over write of attributes
0591        ;       already placed on the screen or removes unnecessary
0591        ;       attributes.
0591        ;
0591 3AD811 ADJSCR: LD     A,(ATTBUF)     ;GET LATEST ATTRIBUTE
0594 BE             CP     (HL)           ;SAME AS PRESENT?
0595 200A           JR     NZ,NOTSAM      ;NO - CONTINUE
0597 23             INC    HL             ;SET POINTER
0598 CD2806         CALL   TMPSCN         ;DO TEMP SCAN
059B CDEC05         CALL   MOVSCR         ;WRITE OVER ATTRIBUTE
059E 23             INC    HL             ;RESET POINTER
059F 1833           JR     EDSCN          ;END TEMP SCAN
05A1 23     NOTSAM: INC    HL             ;POINT TO NEXT POSSIBLE ATTRIBUTE
05A2 7E             LD     A,(HL)         ;GET CHARACTER
05A3 FEF0           CP     ELCODE         ;AT EOL?
05A5 2002           JR     NZ,ADJOVR      ;NO - CONTINUE
05A7 2B             DEC    HL             ;RETURN POINTER
05A8 C9             RET
05A9 23     ADJOVR: INC    HL
05AA 7E             LD     A,(HL)         ;GET CHARACTER
05AB FEF0           CP     ELCODE         ;AT EOL?
05AD 282E           JR     Z,ATTONE       ;YES - ALL DONE
05AF E6C0           AND    0C0H           ;STRIP LOW BITS
05B1 FEC0           CP     0C0H           ;IS IT DMA CODE?
05B3 2004           JR     NZ,NOADJ       ;NO CHECK FOR ATTRIBUTE
05B5 3620           LD     (HL),' '       ;WRITE OVER DMA CODE
05B7 1824           JR     ATTONE
05B9 FE80   NOADJ:  CP     80H            ;IS IT AN ATTRIBUTE?
05BB 2020           JR     NZ,ATTONE      ;YES MOVE SCREEN
05BD CD2806         CALL   TMPSCN         ;MOVE LINE FOR TEMP SCAN
05C0 CDEC05         CALL   MOVSCR
05C3 3620           LD     (HL),' '       ;COVER OLD ATTRIBUTE
05C5 23             INC    HL             ;LOOK AT ATTRIBUTE
05C6 3AD811         LD     A,(ATTBUF)     ;GET CURRENT ATTRIBUTE
05C9 BE             CP     (HL)           ;COMPARE
05CA 2B             DEC    HL             ;RESET POINTER
05CB 2007           JR     NZ,EDSCN       ;NO - FINISH
05CD 23             INC    HL             ;SET POINTER
05CE CDEC05         CALL   MOVSCR         ;MOVE LEFT ONE
05D1 23             INC    HL             ;RESET POINTER
05D2 3620           LD     (HL),' '       ;COVER ATTRIBUTE W/SPACE
05D4 E5     EDSCN:  PUSH   HL             ;SAVE REGISTER
```

```
05D5 2ADB11         LD     HL,(TEMP2)     ;GET LINE ADDRESS
05D8 CD3412         CALL   RWDMOV
05DB E1             POP    HL             ;RESTORE REGISTER
05DC C9             RET
05DD 2B     ATTONE: DEC    HL             ;PLACE ATTRIBUTE OVER ONE
05DE 2B             DEC    HL
05DF 7E             LD     A,(HL)         ;GET ATTRIBUTE
05E0 223512         LD     (RWDMOV+1),HL  ;SET UP FOR WORD MOVE
05E3 E5             PUSH   HL             ;SAVE ADDRESS
05E4 67             LD     H,A            ;MOVE ATTRIBUTE
05E5 2E20           LD     L,' '          ;LOAD SPACE IN H
05E7 CD3412         CALL   RWDMOV         ;PUT SPACE ATTRIBUTE ON SCREEN
05EA E1             POP    HL             ;RESTORE ADDRESS
05EB C9             RET
05EC        ;
05EC E5     MOVSCR: PUSH   HL             ;SAVE REGISTERS
05ED D5             PUSH   DE
05EE C5             PUSH   BC
05EF 5D             LD     E,L            ;MOVE HL TO DE
05F0 54             LD     D,H
05F1 2B             DEC    HL             ;POINT FIRST CHARACTER
05F2 1A     NXTMOV: LD     A,(DE)         ;GET CHARACTER TO BE MOVED
05F3 FEF0           CP     ELCODE         ;AT END OF LINE?
05F5 2004           JR     NZ,OVRMOV      ;NO - MOVE CHARACTER
05F7 3620           LD     (HL),' '       ;PUT SPACE
05F9 1809           JR     MOVEND         ;STOP
05FB 77     OVRMOV: LD     (HL),A         ;MOVE CHARACTER
05FC FEF1           CP     0F1H           ;WAS IT A DMA CODE?
05FE 2804           JR     Z,MOVEND       ;YES - STOP
0600 23             INC    HL             ;MOVE ON TO NEXT CHARACTER
0601 13             INC    DE
0602 18EE           JR     NXTMOV
0604 C1     MOVEND: POP    BC             ;RESTORE REGISTERS
0605 D1             POP    DE
0606 E1             POP    HL
0607 2B             DEC    HL             ;MOVE BACK
0608 2B             DEC    HL
0609 C9             RET
060A        ;
060A        ;       MOVE LINE RIGHT
060A        ;       This routine moves a line of characters to the right
060A        ;       one position to make room for an attribute byte.
060A        ;
060A E5     MVCHRT: PUSH   HL             ;SAVE REGISTERS
060B D5             PUSH   DE
060C C5             PUSH   BC
060D 5D             LD     E,L            ;DESTINATION ADDR IN DE
060E 54             LD     D,H
060F 13             INC    DE             ;INC TO RIGHT
0610 7E             LD     A,(HL)         ;GET CHARACTER
0611 4F             LD     C,A            ;TEMP STORE
0612 1A     MOVRGT: LD     A,(DE)         ;GET CHARACTER
0613 47             LD     B,A            ;TEMP STORE
0614 FEF0           CP     ELCODE
0616 280C           JR     Z,MOVDON       ;YES - MOVE DONE
0618 79             LD     A,C            ;GET TEMP CHAR
0619 12             LD     (DE),A         ;MOVE IT
061A 48             LD     C,B            ;MOVE OVER
061B 79             LD     A,C            ;CHECK FOR DMA BYTE
061C E6F0           AND    0F0H
061E FEF0           CP     0F0H
```

```
0620 23                 INC     HL              ;INC POINTERS
0621 13                 INC     DE
0622 20EE               JR      NZ,MOVR3T       ;NEXT CHARACTER
0624 C1         MOVDON: POP     BC              ;RESTORE REGISTERS
0625 D1                 POP     DE
0626 E1                 POP     HL
0627 C9                 RET
0628            ;
0628            ;       TEMPORARY SCAN ROUTINE
0628            ;
0628            ;       This routine move the current line to a
0628            ;       temporary buffer.  The temporary buffer
0628            ;       address is then placed in the refresh
0628            ;       address table.  This prevents flashing on
0628            ;       the screen during attribute updates.
0628            ;       The original address is returned to the
0628            ;       table after the update.
0628            ;
0628 C5         TMPSCN: PUSH    BC              ;SAVE REGISTERS
0629 D5                 PUSH    DE
062A E5                 PUSH    HL
062B 0600               LD      B,0             ;CLEAR B FOR ADD
062D CB01               RLC     C               ;MULTIPLY C X2
062F 210012             LD      HL,SCRTBL       ;LOAD SCREEN ADDRESS TABLE
0632 09                 ADD     HL,BC           ;ADD LINE OFFSET
0633 223512             LD      (RWDMOV+1),HL   ;SAVE TABLE ADDRESS
0636 5E                 LD      E,(HL)          ;GET LINE ADDRESS
0637 23                 INC     HL
0638 56                 LD      D,(HL)
0639 ED53DB11           LD      (TEMP2),DE      ;SAVE LINE ADDRESS
063D            ;
063D EB                 EX      DE,HL           ;GET LINE ADDRESS
063E CBBC               RES     7,H             ;CLEAR DMA BIT
0640 115012             LD      DE,TEMPLN       ;GET TEMPORARY LINE ADDRESS
0643 016000             LD      BC,96           ;GET LINE COUNT
0646 EDB0               LDIR                    ;MOVE LINE TO TEMPORARY
0648 215012             LD      HL,TEMPLN       ;GET TEMPORARY LINE ADDRESS
064B CBFC               SET     7,H             ;SET DMA BIT
064D CD3412             CALL    RWDMOV          ;SWAP LINE IN TABLE
0650 E1                 POP     HL              ;RESTORE REGISTERS
0651 D1                 POP     DE
0652 C1                 POP     BC
0653 C9                 RET
0654            ;
0654            ;       FIND PREVIOUS ATTRIBUTE CODE
0654            ;
0654            ;       This routine finds the last attribute
0654            ;       placed in front of the cursor so the
0654            ;       proper bit may be toggled without dis-
0654            ;       turbing any previously toggled bits.
0654            ;
0654 E5         PRVATT: PUSH    HL              ;SAVE REGISTERS
0655 D5                 PUSH    DE
0656 C5                 PUSH    BC
0657 EB         NOFDAT: EX      DE,HL           ;PUT CURRENT POINTER IN DE
0658 CDDF01             CALL    LNADD           ;GET BEGINING OF LINE
065B EB                 EX      DE,HL           ;PUT ADDRESS IN DE
065C B7         NXTATT: OR      A               ;CLEAR CARRY
065D E5                 PUSH    HL              ;SAVE ADDRESS
065E ED52               SBC     HL,DE           ;AT FRONT OF LINE YET?
0660 E1                 POP     HL              ;RESTORE ADDRESS
```

```
0661 FA7006             JP      M,FDONE         ;NO ATTRIBUTES FOUND
0664 7E                 LD      A,(HL)          ;GET CHARACTER
0665 E6C0               AND     0C0H            ;MASK LSB
0667 FE80               CP      80H             ;ATTRIBUTE?
0669 7E                 LD      A,(HL)          ;RESTORE CHARACTER
066A 2806               JR      Z,FINDAT        ;YES STOP
066C 2B                 DEC     HL              ;DEC POINTER
066D C35C06             JP      NXTATT
0670 3E80       FDONE:  LD      A,80H           ;BLANK ATTRIBUTE CODE
0672 C1         FINDAT: POP     BC
0673 D1                 POP     DE
0674 E1                 POP     HL
0675 C9                 RET
0676            ;
0676 E5         PRVAT2: PUSH    HL              ;SAVE REGISTERS
0677 D5                 PUSH    DE
0678 C5                 PUSH    BC
0679 ED5BDB11           LD      DE,(TEMP2)      ;LOAD LINE ADDRESS
067D CBBA               RES     7,D             ;CLEAR DMA BIT
067F 18DB               JR      NXTATT          ;FIND NEXT ATTRIBUTE
0681            ;
0681            ;       LIGHT PEN ROUTINE
0681            ;
0681            ;       This routine reads the light pen registers
0681            ;       after being set by a light pen.  The data
0681            ;       is then sent to the keyboard port in two
0681            ;       bytes with the column first.  Note - due to
0681            ;       delays in the light pen the column position
0681            ;       may have to be corrected in the host computer
0681            ;       software.
0681            ;
0681 E1         LPEN:   POP     HL              ;RESTORE REGISTER
0682 DB41       LPLP:   IN      A,(CRTC)        ;GET STATUS
0684 E610               AND     10H             ;MASK BITS
0686 28FA               JR      Z,LPLP          ;LOOP TILL READY
0688 3E60               LD      A,LITP          ;GET DATA
068A D341               OUT     (CRTC),A
068C DB40               IN      A,(CRTD)        ;GET COLUMN
068E 47                 LD      B,A
068F CDC906             CALL    DTAOUT          ;SEND IT
0692 DB40               IN      A,(CRTD)        ;GET LINE
0694 47                 LD      B,A
0695 CDC906             CALL    DTAOUT          ;SEND IT
0698 C9                 RET
0699            ;
0699            ;       PUT ANY CHARACTER ON SCREEN
0699            ;
0699            ;       This routine puts any ASCII character
0699            ;       on the screen
0699            ;
0699 F1         GRAF:   POP     HL              ;RESTORE REGISTER
069A CDEC00             CALL    STATUS          ;WAIT FOR NEXT CHARACTER
069D DB10               IN      A,(VDS100)      ;GET CHARACTER
069F E67F               AND     7FH             ;STRIP PARITY
06A1 C31101             JP      PLCHAR          ;PLACE CHARACTER
06A4            ;
06A4            ;       READ CURSOR POSTION
06A4            ;
06A4            ;       This routine sends the cursor postion
0644            ;       to the keyboard port in two bytes with
06A4            ;       the line first.
```

```
06A4            ;
06A4 E1         RDCUR:  POP     HL              ;RESTORE REGISTER
06A5 3E20               LD      A,OFFSET        ;GET OFFSET
06A7 81                 ADD     A,C             ;ADD LINE
06A8 47                 LD      B,A             ;MOVE TEMP
06A9 CDC906             CALL    DTAOUT          ;SEND DATA
06AC 3E20               LD      A,OFFSET        ;GET OFFSET
06AE 83                 ADD     A,E             ;ADD COLUMN
06AF 47                 LD      B,A             ;MOVE TEMP
06B0 CDC906             CALL    DTAOUT          ;SEND DATA
06B3 C9                 RET
06B4            ;
06B4            ;       CHANGE SCROLL SPEED ROUTINE
06B4            ;
06B4            ;       This routine allows the user to control the
06B4            ;       scroll speed of the screen. 0 is the fastest
06B4            ;       (default) and 9 is the slowest.
06B4            ;
06B4 E1         SCLWT:  POP     HL              ;RESTORE REGISTER
06B5 CDEC00             CALL    STATUS          ;GET NUMBER
06B8 DB10               IN      A,(VDS100)
06BA E60F               AND     0FH             ;STRIP ASCII
06BC FE0A               CP      10              ;IF > 9 MAKE 9
06BE FAC306             JP      M,NOT9
06C1 3E09               LD      A,9             ;MAKE NINE
06C3 07         NOT9:   RLCA                    ;MULT X4
06C4 07                 RLCA
06C5 32D111             LD      (WTCTR),A       ;STORE IT
06C8 C9                 RET
06C9            ;
06C9            ;       SEND DATA TO POST COMPUTER
06C9            ;
06C9            ;       This routine send data to the host computer
06C9            ;       via the keyboard port.
06C9            ;
06C9 DB20       DTAOUT: IN      A,(KYS100)      ;CHECK STATUS
06CB 1F                 RRA
06CC 38FB               JR      C,DTAOUT
06CE 78                 LD      A,B             ;GET DATA
06CF D320               OUT     (KYS100),A      ;SEND IT
06D1 C9                 RET
06D2            ;
06D2            ;       RESTORE OLD CURSOR POSITION
06D2            ;
06D2 E1         OLDPN:  POP     HL              ;RESTORE REGISTER
06D3 3ADD11             LD      A,(SAVLN)       ;GET OLD LINE
06D6 4F                 LD      C,A             ;PUT IN C
06D7 3ADE11             LD      A,(SAVCOL)      ;GET OLD COLUMN
06DA 5F                 LD      E,A             ;PUT IN E
06DB C33103             JP      PLCUR           ;PLACE CURSOR
06DE            ;
06DE            ;       MEMORY MAP SOFTWARE
06DE            ;
06DE            ;
06DE            ;       INITIALIZE SCREEN FOR MEMORY MAP
06DE            ;
06DE            ;       This routine removes all end of line codes
06DE            ;       from the screen memory so any poked characters
06DE            ;       will be seen immediately. It also enables
06DE            ;       the memory map circuits.
06DE            ;
```

```
06DE 3AD711     MEMINT: LD      A,(MMSTUS)      ;IS MEM MAP ENABLED
06E1 FE01               CP      1               ;
06E3 C8                 RET     Z               ;BYPASS IF IT IS
06E4 C5                 PUSH    BC              ;SAVE REGISTERS
06E5 D5                 PUSH    DE
06E6 E5                 PUSH    HL
06E7 0E00               LD      C,0             ;LOAD LINE ZERO
06E9 CDF706             CALL    MMFILL          ;FILL MEMORY WITH SPACES
06EC DB73               IN      A,(MMEN)        ;ENABLE MEMORY MAP CKTS
06EE 3E01               LD      A,1             ;SET I/O MEM MAP FLAG
06F0 32D711             LD      (MMSTUS),A
06F3 E1                 POP     HL              ;RESTORE REGISTERS
06F4 D1                 POP     DE
06F5 C1                 POP     BC
06F6 C9                 RET
06F7            ;
06F7            ;       MEMORY MAP FILL ROUTINE
06F7            ;
06F7            ;       This routine fills the display memory with
06F7            ;       spaces.
06F7            ;
06F7 C5         MMFILL: PUSH    BC              ;SAVE REGISTERS
06F8 D5                 PUSH    DE
06F9 E5                 PUSH    HL
06FA 1E4F               LD      E,79            ;LOAD LAST COLUMN
06FC CDDF01     SPCLP:  CALL    LNADD           ;CALC LINE ADDRESS
06FF CDF201             CALL    FILL            ;FILL WITH SPACES
0702 0C                 INC     C               ;INC LINE
0703 3E18               LD      A,MMLINE        ;LOAD MAX LINE
0705 B9                 CP      C               ;AT BOTTOM YET?
0706 20F4               JR      NZ,SPCLP
0708 E1                 POP     HL              ;RESTORE REGISTERS
0709 D1                 POP     DE
070A C1                 POP     BC
070B C9                 RET
070C            ;
070C            ;       MEMORY MAP STATUS ROUTINE
070C            ;
070C            ;       This routine checks the status of the video
070C            ;       I/O port the memory map latches and the keyboard
070C            ;       buffer.  If any are true they are serviced.
070C            ;
070C E1         MEM:    POP     HL              ;RESTORE REGISTER
070D CDDE06             CALL    MEMINT          ;INITIALIZE SCREEN
0710 DB00       MMSTAT: IN      A,(S100ST)      ;CHECK OUT STAUS
0712 1F                 RRA
0713 DCFA00             CALL    C,VIDEO         ;YES, PUT CHARACTER ON SCREEN
0716 3AD211             LD      A,(BCTR)        ;CHECK KEYBOARD BUFFER
0719 FE00               CP      0
071B C4D902             CALL    NZ,KEYOUT       ;SEND CHARACTER TO HOST
071E DB74               IN      A,(MMS)         ;GET MEMORY MAP STATUS
0720 1F                 RRA
0721 D42607             CALL    NC,MMVIDE       ;YES PUT CHARACTER ON SREEEN
0724 18EA               JR      MMSTAT          ;LOOP AGAIN
0726            ;
0726            ;       MEMORY MAP SERVICE ROUTINE
0726            ;
0726            ;       This routine takes the address latched in the memory
0726            ;       map latches and calculates the position on the screen.
0726            ;       The it takes the data from the latches and places it
0726            ;       in refresh memory.
```

```
0726            ;
0726 C5         MMVIDE: PUSH    BC              ;SAVE REGISTERS
0727 D5                 PUSH    DE
0728 E5                 PUSH    HL
0729 DB70               IN      A,(MMH)         ;GET ADDRESS
072B 67                 LD      H,A
072C DB71               IN      A,(MML)
072E 6F                 LD      L,A
072F E60F               AND     0FH             ;STRIP HIGH BITS
0731 F5                 PUSH    AF              ;SAVE OFFSET
0732 CB3C               SRL     H               ;ROTATE HL LEFT FOUR TIMES
0734 CB1D               RR      L
0736 CB3C               SRL     H
0738 CB1D               RR      L
073A CB3C               SRL     H
073C CB1D               RR      L
073E CB3C               SRL     H
0740 CB1D               RR      L
0742 0605               LD      B,5             ;LOAD DIVIDE COUNTER
0744 1650               LD      D,MMCOL         ;LOAD MAX COLUMN
0746 0E00               LD      C,0             ;ZERO C
0748 7D                 LD      A,L             ;MOVE NUMBER TO A
0749 92         MMDIV:  SUB     D               ;SUB DIVISOR
074A F25007             JP      P,MMPOS         ;POSITIVE NUMBER CONTINUE
074D 82                 ADD     A,D             ;RESTORE NUMBER
074E 1802               JR      MMCONT
0750 CBC1       MMPOS:  SET     0,C             ;SET BIT
0752 CB21       MMCONT: SLA     C               ;ROTATE REGISTERS TO DIVIDE
0754 CB27               SLA     A
0756 10F1               DJNZ    MMDIV           ;CONTINUE
0758 CB39               SRL     C               ;RESTORE POSITION
075A CB3F               SRL     A
075C D1                 POP     DE              ;GET REMAINDER
075D 82                 ADD     A,D             ;ADD TO OFFSET
075E 1600               LD      D,0             ;READY FOR ADD
0760 5F                 LD      E,A
0761 3E17               LD      A,MMLINE-1      ;CHECK IF OVER MAX LINE
0763 B9                 CP      C
0764 FA7007             JP      M,MMEND         ;YES - DO NOTHING
0767 CDDF01             CALL    LNADD           ;CALC LINE ADDRESS
076A 19                 ADD     HL,DE           ;ADD COLUMN NUMBER
076B DB72               IN      A,(MMD)         ;GET DATA
076D 77                 LD      (HL),A          ;PUT ON SCREEN
076E 1802               JR      MMEND1
0770 DB72       MMEND:  IN      A,(MMD)         ;WASTE DATA & RESET FLAG
0772 E1         MMEND1: POP     HL              ;RESTORE REGISTERS
0773 D1                 POP     DE
0774 C1                 POP     BC
0775 C9                 RET
0776            ;
0776            ;       MEMORY MAP OFF
0776            ;
0776            ;       This routine turns the memory map circuits off
0776            ;       and returns to I/O map only mode.
0776            ;
0776 E1         MEMO:   POP     HL              ;RESTORE REGISTER
0777 3AD711             LD      A,(MMSTUS)      ;IS MEM MAP DISABLED
077A FE00               CP      0               ;
077C C8                 RET     Z               ;BYPASS IF IT IS
077D DB73               IN      A,(MMEN)        ;RESET MEMORY MAP CKTS
077F AF                 XOR     A               ;RESET I/O MEM MAP FLAG
0780 32D711             LD      (MMSTUS),A
0783 F1                 POP     AF              ;WASTE CALL
0784 C3E400             JP      BDSTAT          ;RETURN TO STATUS ROUTINE
0787            ;
0787            ;       CONTROL CHARACTER TRAP TABLE
0787            ;
0787 1A         CTLTBL: DEFB    CHOME
0788 00                 DEFB    CSTCUR
0789 0B                 DEFB    CCURUP
078A 0A                 DEFB    CCURDN
078B 0C                 DEFB    CCURRT
078C 08                 DEFB    CCURLT
078D 00                 DEFB    CEOL
078E 00                 DEFB    CEOS
078F 1E                 DEFB    CHOMUP
0790 07                 DEFB    CBELL
0791 00                 DEFB    CHLITE
0792 00                 DEFB    CNHLIT
0793 00                 DEFB    CREVID
0794 00                 DEFB    CNRVID
0795 00                 DEFB    CBLINK
0796 00                 DEFB    CNBLNK
0797 00                 DEFB    CLPEN
0798 00                 DEFB    CMEM
0799 00                 DEFB    CMEMO
079A 00                 DEFB    CRDCUR
079B 00                 DEFB    CSCLWT
079C 00                 DEFB    CALTCH
079D 00                 DEFB    CDELLN
079E 00                 DEFB    CINLN
079F 16                 DEFB    CGRAF
07A0 0E                 DEFB    CHTOG
07A1 01                 DEFB    COLDPN
07A2 00                 DEFB    00
07A3            ;
07A3            ;       ESCAPE CHARACTER TRAP TABLE
07A3            ;
07A3 5A         ESCTBL: DEFB    EHOME
07A4 3D                 DEFB    ESTCUR
07A5 00                 DEFB    ECURUP
07A6 00                 DEFB    ECURDN
07A7 00                 DEFB    ECURRT
07A8 00                 DEFB    ECURLT
07A9 54                 DEFB    EEOL
07AA 59                 DEFB    EEOS
07AB 00                 DEFB    EHOMUP
07AC 00                 DEFB    EBELL
07AD 29                 DEFB    EHLITE
07AE 28                 DEFB    ENHLIT
07AF 6A                 DEFB    EREVID
07B0 6B                 DEFB    ENRVID
07B1 5E                 DEFB    EBLINK
07B2 71                 DEFB    ENBLNK
07B3 4C                 DEFB    ELPEN
07B4 4D                 DEFB    EMEM
07B5 4F                 DEFB    EMEMO
07B6 3F                 DEFB    ERDCUR
07B7 53                 DEFB    ESCLWT
07B8 58                 DEFB    EALTCH
07B9 52                 DEFB    EDELLN
07BA 45                 DEFB    EINLN
```

```
07BB 00                 DEFB    EGRAF
07BC 00                 DEFB    EHTOG
07BD 00                 DEFB    EOLDPN
07BE 00                 DEFB    00
07BF             ;
07BF             ;      ATTRIBUTE JUMP TABLE
07BF             ;
07BF D206        ATTTBL: DEFW   OLDPN
07C1 7804               DEFW    HTOG
07C3 9906               DEFW    GRAF
07C5 2E04               DEFW    INLN
07C7 D303               DEFW    DELLN
07C9 A804               DEFW    ALTCH
07CB B406               DEFW    SCLWT
07CD A406               DEFW    RDCUR
07CF 7607               DEFW    MEMO
07D1 0C07               DEFW    MEM
07D3 8106               DEFW    LPEN
07D5 A004               DEFW    NBLNK
07D7 9804               DEFW    BLINK
07D9 9004               DEFW    NRVID
07DB 8804               DEFW    REVID
07DD 7004               DEFW    NHLIT
07DF 6804               DEFW    HLITE
07E1 6404               DEFW    BELL
07E3 5D04               DEFW    HOMUP
07E5 AA03               DEFW    EOS
07E7 6403               DEFW    EOL
07E9 5C03               DEFW    CURLT
07EB 5403               DEFW    CURRT
07ED 4903               DEFW    CURDN
07EF 4103               DEFW    CURUP
07F1 F902               DEFW    STCUR
07F3 F502               DEFW    HOME
0000             THEEND: END


05A9 ADJOVR   0591 ADJSCR   04A8 ALTCH
0586 ATDONE   0516 ATONE    04F9 ATTBCK
11D8 ATTBUF   001B ATTCTR   058E ATTFIN
11D9 ATTFLG   04B6 ATTFSH   024B ATTLOP
0557 ATTNMT   05DD ATTONE   0549 ATTSAM
055A ATTSP    07BF ATTTBL   11D2 BCTR
00E4 BDSTAT   0464 BELL     0498 BLINK
01D4 BLKLP    00D0 BOTKEY   0181 CALLN
0000 CALTCH   0007 CBELL    0000 CBLINK
000A CCURDN   0008 CCURLT   000C CCURRT
000B CCURUP   0000 CDELLN   0000 CEOL
0000 CEOS     0016 CGRAF    0192 CHECK
0198 CHI      0000 CHLITE   001A CHOME
001E CHOMUP   000E CHTOG    0000 CINLN
031E CKCUR    0000 CLPEN    0235 CLS
0000 CMEM     0000 CMEMO    0000 CNBLNK
0000 CNHLIT   0000 CNRVID   0001 COLDPN
00C9 CONT1    000D CR       0000 CRDCUR
0000 CREVID   0169 CRLF     0041 CRTC
0040 CRTD     00A0 CRTEI    01A5 CRTWT
0000 CSCLWT   0000 CSTCUR   0787 CTLTBL
0135 CTRAP    0349 CURDN    035C CURLT
030D CURNXT   0020 CURPOS   0354 CURRT
02C6 CURSOR   0341 CURUP    0423 DBLKLP
03FF DCRTWT   03D3 DELLN    0421 DFIL
02C6 DLEND    053B DMA1     01D2 DMAFIL
0545 DMAOVR   0030 DMARST   0407 DNOMOV
0418 DSPLP    06C9 DTAOUT   0000 DUMMY
0058 EALTCH   0000 EBELL    005E EBLINK
0000 ECURDN   0000 ECURLT   0000 ECURRT
0000 ECURUP   0052 EDELLN   05D4 EDSCN
0054 EEOL     0059 EEOS     0000 EGRAF
0029 EHLITE   005A EHOME    0000 EHOMUP
0000 EHTOG    0045 EINLN    00F0 ELCODE
004C ELPEN    004D EMEM     004F EMEMO
0071 ENBLNK   00F1 ENDDMA   0028 ENHLIT
03B5 ENL      006B ENRVID   0364 EOL
0365 EOL1     037E EOLATT   0000 EOLDPN
039F EOLEND   0395 EOLLP    039B EOLOVR
03BD EOLSP    03AA EOS      03BB EOSLP
003F ERDCUR   006A EREVID   001B ESC
0155 ESCAPE   0053 ESCLWT   07A3 ESCTBL
003D ESTCUR   013B FDMAT    0670 FDONE
01F2 FILL     0672 FINDAT   01F6 FLOOP
0699 GRAF     004F HCR      0468 HLITE
02F5 HOME     045D HOMUP    0478 HTOG
044D ICRTWT   042E INLN     022D INTCRT
023B INTLP   *0211 INTST    028E ISCTBL
0060 KEYBRD   02D9 KEYOUT   11D3 KYINA
11D5 KYOTA    0020 KYS100   000A LF
0177 LIFD     0060 LITP     01DF LNADD
040D LNFIL    0276 LOGON    0681 LPEN
0682 LPLP     070C MEM      06DE MEMINT
0776 MEMO     0519 MERGE    0009 MFCH
0050 MMCOL    0752 MMCONT   0072 MMD
0749 MMDIV    0073 MMEN     0770 MMEND
0772 MMEND1   06F7 MMFILL   0070 MMH
0071 MML      0018 MMLINE   0750 MMPOS
0074 MMS      0710 MMSTAT   11D7 MMSTUS
0726 MMVIDE   00CF MORINT   0624 MOVDON
0604 MOVEND   0612 MOVRST   05EC MOVSCR
0223 MSG      060A MVCHRT   04A0 NBLNK
0470 NHLIT    032F NO25     05B9 NOADJ
0532 NOAT2    0045 NOBOT    02EC NOBOT1
0538 NOCHG    01FF NODMA    0657 NOFDAT
020C NOFILL   0151 NOMAT    00B9 NOMORE
0455 NOMOVE   06C3 NOT9     0209 NOTATT
05A1 NOTSAM   019F NOWAIT   0490 NRVID
0000 NULL     065C NXTATT   05F2 NXTMOV
0020 OFFSET   06D2 OLDPN    05FB OVRMOV
0584 OVRSP    03E3 OVRSTP   028A PARTBL
0111 PLCHAR   0331 PLCUR    04BF PRATT
0676 PRVAT2   0654 PRVATT   06A4 RDCUR
04B4 RESETA   0484 RESETH   0488 REVID
1234 RWDMOV   0000 S100ST   11DE SAVCOL
11DD SAVLN    06B4 SCLWT    018F SCROLL
1200 SCRTBL   01C9 SPALP    06FC SPCLP
0050 SPEAK    0585 SPOVR    13A0 STACK
00EC STATUS   02F9 STCUR    0216 STINLP
00F3 STPDMA   0023 STRSCR   11DA TEMP
11DB TEMP2    1250 TEMPLN   07F5 THEEND
0628 TMPSCN   11A0 TOPKEY   8800 TOPSC8
0800 TOPSCR   0079 ULL      0010 VDS100
00FA VIDEO    0058 VVVR     02C2 WDMOVE
0164 WRAP     11D1 WTCTR

00 ERRORS
```