THE BIGBOARD: AN OVERVIEW

The Big Board was designed to fill the needs of the OEM, business, and educational markets for a low cost but high performance general purpose computer. Please note, however, that the Big Board is NOT a finished product. An enclosure, floppy disc drives, power supplies, fan, keyboard, and a video monitor must be added in order to make a complete computer.

A typical user system might include a completed Big Board with two disk drives, an ASCII keyboard for input with a high quality video monitor for output. The user might also have a high speed line printer connected to the optional parallel port. This would require a user written printer driver routine.

In place of the keyboard-monitor combination, some sort of serial terminal such as a LA36 Decwriter or Lear Sigler ADM-3A could be connected to the optional serial I/O port. This terminal can automatically become the system console. In addition a modem could be connected to the other serial channel for communications over the telephone lines with OTHER computers or equipment. The modem would require user written software drivers also. A listing of our PPM system monitor is available to make user written software easier to integrate into the Big Board.

The designers of the Big Board tried to include the right mix of most commonly needed computing features all on one handy sized PC board. These main features include:

64K System RAM----Z-80 CPU----Floppy Disc Controller----

24 Line by 80 Character Video ----ASCII Keyboard Input Port---

Two full serial I/O ports, a Real time clock, and two parallel I/O ports are available as options.

A very powerful System Monitor is included in ROM.

The Big Board kit is fully socketed and includes only the finest quality components.

The Big Board was designed to primarily run the popular CP/M disk operating system. This allows the Big Board to execute the tremendous software base (8080/Z80) that exists to run under CP/M.

As with any design, certain tradeoffs were made in order to produce the most cost effective and reliable machine possible. The Big Board is not expandable beyond its present configuration without extensive modifications. If the 64K of RAM and all of the I/O options of the Big Board are not enough for your application, then we strongly recommend that you implement your
system on either the SS-50 or S-100 buss. In order to maintain the highest possible degree of reliability, we have elected to retain the single density IBM 3740 disk format. This also assures software compatibility between the Big Board and other CP/M based systems.

The Big Board is not a "consumer" or personal computer. It is a serious and powerful machine that requires both hardware and software expertise in its construction and use. If you need a ready to use, no experience needed, take home and RUN computer, then we recommend the many fine systems offered by Radio Shack, Apple, Commodore, Atari, Texas Instruments, etc.

CP/M is a TM of Digital Research (California)
IC AND PARTS CROSS REFERENCE

74LS00=U52,53
74LS02=U87
74LS04=U11,66,77,103,110
7406=U105
74LS08=U37,104
74LS10=U10
74LS14=U101,112,114
74LS20=U56
74LS32=U25,81
7445=U109
74LS74=U9,12,60,108
74LS123=U38,51,106
74LS136=U94
74LS138=U55,84,85,86
74LS151=U75
74LS157=U47,48,49,50
74157=U58,59
74LS161=U24
74164=U76
74LS174=U35,74
74LS193=U98
74LS241=U65,82
74LS242=U99,100
74LS243=U71,72,78,79
74LS283=U34,36
74LS290=U22,97
74LS293=U96
74LS373=U83
74LS393=U21,23
8216=U54,57
2114=U61,62,63,64
2716CHAR-ROM=U73
2716PFM=U67
Z80CPU=U80
Z80PID(MK3881)=U111
FD1771=U102
4116(16KRAM)=U1-8,13-20,26-33,39-46
14.318MHZ=Y1
20.000MHZ=Y2
2N2222=Q1(VIDEO)
2N2907=Q2(CPU)
1N751=VR1
# THEORY OF OPERATION

## BIG BOARD PARTS LIST (BASIC I/O)

### SEMICONDUCTORS

<table>
<thead>
<tr>
<th>Device</th>
<th>Quantity</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>74LS00</td>
<td>65</td>
<td>QUAD 2 INPUT NAND</td>
</tr>
<tr>
<td>74LS02</td>
<td>71</td>
<td>QUAD 2 INPUT NOR</td>
</tr>
<tr>
<td>74LS04</td>
<td>71</td>
<td>HEX INVERTER</td>
</tr>
<tr>
<td>7406</td>
<td>43</td>
<td>HEX O.C. INVERTING BUFFER</td>
</tr>
<tr>
<td>74LS08</td>
<td>102</td>
<td>QUAD 2 INPUT AND</td>
</tr>
<tr>
<td>74LS10</td>
<td>71</td>
<td>TRIPLE 3 INPUT NAND</td>
</tr>
<tr>
<td>74LS14</td>
<td>73</td>
<td>SCHMITT HEX INVERTER</td>
</tr>
<tr>
<td>74LS20</td>
<td>65</td>
<td>DUAL 4 INPUT NAND</td>
</tr>
<tr>
<td>74LS32</td>
<td>71</td>
<td>QUAD 2 INPUT OR</td>
</tr>
<tr>
<td>7445</td>
<td>75</td>
<td>BUFFERED O.C. BCD DECODER</td>
</tr>
<tr>
<td>74LS74</td>
<td>80</td>
<td>DUAL D FLIP FLOP</td>
</tr>
<tr>
<td>74136</td>
<td>727</td>
<td>QUAD 2 INPUT XOR</td>
</tr>
<tr>
<td>74LS123</td>
<td>174</td>
<td>DUAL MONO</td>
</tr>
<tr>
<td>74LS138</td>
<td>127</td>
<td>4-BINARY DECODER</td>
</tr>
<tr>
<td>74LS151</td>
<td>177</td>
<td>1 OF 8 MUX</td>
</tr>
<tr>
<td>74LS157</td>
<td>12</td>
<td>QUAD 2:1 MUX</td>
</tr>
<tr>
<td>74157</td>
<td>12</td>
<td>HI DRIVE VERSION OF ABOVE</td>
</tr>
<tr>
<td>74LS161</td>
<td>171</td>
<td>PRESETTABLE SYNCHRONOUS BINARY COUNTER</td>
</tr>
<tr>
<td>74164</td>
<td>174</td>
<td>8 STAGE SHIFT REG.</td>
</tr>
<tr>
<td>74LS174</td>
<td>127</td>
<td>HEX D LATCH</td>
</tr>
<tr>
<td>74LS193</td>
<td>167</td>
<td>SYNC BINARY U/D COUNTER</td>
</tr>
<tr>
<td>74LS241</td>
<td>267</td>
<td>OCTAL BUFFER</td>
</tr>
<tr>
<td>74LS242</td>
<td>251</td>
<td>4 BIT INVERTING BUS BUFFER</td>
</tr>
<tr>
<td>74LS243</td>
<td>251</td>
<td>4 BIT BUS BUFFER</td>
</tr>
<tr>
<td>74LS283</td>
<td>671</td>
<td>4 BIT FULL ADDER</td>
</tr>
<tr>
<td>74LS290</td>
<td>164</td>
<td>BCD COUNTER</td>
</tr>
<tr>
<td>74LS293</td>
<td>180</td>
<td>BINARY COUNTER</td>
</tr>
<tr>
<td>74LS373</td>
<td>369</td>
<td>OCTAL LATCH</td>
</tr>
<tr>
<td>74LS393</td>
<td>279</td>
<td>DUAL 4 BIT BINARY COUNTER</td>
</tr>
<tr>
<td>8216</td>
<td>2</td>
<td>SIGNETICS/INTEL BUS BUFFER</td>
</tr>
<tr>
<td>2114</td>
<td>4</td>
<td>1K X 4 NMOS STATIC RAM.</td>
</tr>
<tr>
<td>MK2716-6</td>
<td>350</td>
<td>350 NS CHARACTER GENERATOR</td>
</tr>
<tr>
<td>2716</td>
<td>1</td>
<td>PPM MONITOR ROM</td>
</tr>
<tr>
<td>MK3880</td>
<td>1</td>
<td>Z80-CPU</td>
</tr>
<tr>
<td>MK3881</td>
<td>1</td>
<td>Z80-PIO</td>
</tr>
<tr>
<td>FD 1771-B01</td>
<td>1</td>
<td>WESTERN DIGITAL FLOPPY CONTROLLER</td>
</tr>
<tr>
<td>MK4116-4</td>
<td>348</td>
<td>16K DYNAMIC RAM</td>
</tr>
</tbody>
</table>

2N2907 | 62 | PNP TRANSISTOR |
2N2222 | 62 | NPN TRANSISTOR |
1N751  | 16 | 5.1 VOLT 1/2 WATT ZENER

C 1980 J.B. Ferguson
**THEORY OF OPERATION**

### RESISTORS

<table>
<thead>
<tr>
<th>QUANTITY</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>33 OHM 1/4 WATT</td>
</tr>
<tr>
<td>2</td>
<td>820 OHM 1/4 WATT</td>
</tr>
<tr>
<td>5</td>
<td>4.7K OHM 1/4 WATT</td>
</tr>
<tr>
<td>8</td>
<td>10K OHM 1/4 WATT</td>
</tr>
<tr>
<td>1</td>
<td>1.2K OHM 1/4 WATT</td>
</tr>
<tr>
<td>4</td>
<td>220 OHM 1/4 WATT</td>
</tr>
<tr>
<td>1</td>
<td>330 OHM 1/4 WATT</td>
</tr>
<tr>
<td>2</td>
<td>1.0K 1/4 WATT</td>
</tr>
<tr>
<td>2</td>
<td>100K 1/4 WATT</td>
</tr>
<tr>
<td>1</td>
<td>75 OHM 1/4 WATT</td>
</tr>
<tr>
<td>1</td>
<td>1.5K 1/4 WATT</td>
</tr>
<tr>
<td>1</td>
<td>3.9K 1/4 WATT</td>
</tr>
<tr>
<td>1</td>
<td>4.3K 1/4 WATT</td>
</tr>
<tr>
<td>1</td>
<td>39K 1/4 WATT</td>
</tr>
<tr>
<td>1</td>
<td>68K 1/4 WATT</td>
</tr>
<tr>
<td>2</td>
<td>8 PIN 1K SIP, PIN 1 COMMON</td>
</tr>
</tbody>
</table>

### CRYSTALS

<table>
<thead>
<tr>
<th>ICM PART NO.</th>
<th>FREQUENCY</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>433165</td>
<td>14.31818 MHZ</td>
<td>ALL CRYSTALS ARE SERIES RESONANT</td>
</tr>
<tr>
<td>435260</td>
<td>20.000000 MHZ</td>
<td>32 PF LOAD, HC-18 HOLDER</td>
</tr>
<tr>
<td>433165</td>
<td>5.0688 MHZ</td>
<td>(OPTIONAL)</td>
</tr>
</tbody>
</table>

### CAPACITORS

<table>
<thead>
<tr>
<th>QUANTITY</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>105</td>
<td>.01 - .1 UF DISC OR MONOLITHIC 16V OR MORE</td>
</tr>
<tr>
<td>1</td>
<td>.01 UF DISC CERAMIC .25&quot; LEAD SPACING</td>
</tr>
<tr>
<td>3</td>
<td>33 PF DISC</td>
</tr>
<tr>
<td>1</td>
<td>47 PF DISC</td>
</tr>
<tr>
<td>1</td>
<td>.0033 UF DISC</td>
</tr>
<tr>
<td>8</td>
<td>3.2K 2.2 UF (OR GREATER) 20V AXIAL LEAD TANTILUM</td>
</tr>
<tr>
<td>2</td>
<td>.0068 UF 10V RADIAL LEAD ELECTROLYTIC</td>
</tr>
<tr>
<td>1</td>
<td>.10UF 10V RADIAL LEAD ELECTROLYTIC</td>
</tr>
<tr>
<td>1</td>
<td>.001 100 PF DISC</td>
</tr>
</tbody>
</table>

### CONNECTORS

<table>
<thead>
<tr>
<th>PART NUMBER</th>
<th>QUANTITY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>CA-D26SP100-230-430</td>
<td>1</td>
<td>C.A. 26 PIN CONNECTOR</td>
</tr>
<tr>
<td>CA-D50SP100-230-430</td>
<td>1</td>
<td>C.A. 50 PIN CONNECTOR</td>
</tr>
<tr>
<td>27-28</td>
<td>44</td>
<td>14 PIN SOLDERTAIL SOCKETS</td>
</tr>
<tr>
<td>39-64</td>
<td>55</td>
<td>16 PIN SOLDERTAIL SOCKETS</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>18 PIN SOLDERTAIL SOCKETS</td>
</tr>
</tbody>
</table>

---

C 1980 J.B. Ferguson
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>20 PIN SOLDERTAIL SOCKETS</td>
<td>89</td>
</tr>
<tr>
<td>5</td>
<td>24 PIN SOLDERTAIL SOCKETS</td>
<td>1.07</td>
</tr>
<tr>
<td>5</td>
<td>40 PIN SOLDERTAIL SOCKETS</td>
<td>1.76</td>
</tr>
<tr>
<td>1</td>
<td>28 PIN SOLDERTAIL SOCKET</td>
<td>1.24</td>
</tr>
<tr>
<td>1</td>
<td>ELECTROVERT 8 POS. TERMINAL BLOCK</td>
<td>247.04</td>
</tr>
</tbody>
</table>

25-104-0853

---

c 1980 J.B. Ferguson
### RESISTOR VALUES

<table>
<thead>
<tr>
<th>Resistor</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>R-1</td>
<td>820 OHMS</td>
</tr>
<tr>
<td>R-2</td>
<td>820 OHMS</td>
</tr>
<tr>
<td>R-3</td>
<td>68 K (H. SYNC TIMING)</td>
</tr>
<tr>
<td>R-4</td>
<td>100 K (V. SYNC TIMING)</td>
</tr>
<tr>
<td>R-5</td>
<td>39 K (H. SYNC DELAY)</td>
</tr>
<tr>
<td>R-6</td>
<td>33 OHMS</td>
</tr>
<tr>
<td>R-7</td>
<td>33 OHMS</td>
</tr>
<tr>
<td>R-8</td>
<td>33 OHMS</td>
</tr>
<tr>
<td>R-9</td>
<td>33 OHMS</td>
</tr>
<tr>
<td>R-10</td>
<td>33 OHMS</td>
</tr>
<tr>
<td>R-11</td>
<td>33 OHMS</td>
</tr>
<tr>
<td>R-12</td>
<td>33 OHMS</td>
</tr>
<tr>
<td>R-13</td>
<td>33 OHMS</td>
</tr>
<tr>
<td>R-14</td>
<td>4.7 K</td>
</tr>
<tr>
<td>R-15</td>
<td>4.7 K</td>
</tr>
<tr>
<td>R-16</td>
<td>4.7 K</td>
</tr>
<tr>
<td>R-17</td>
<td>4.7 K</td>
</tr>
<tr>
<td>R-18</td>
<td>4.7 K</td>
</tr>
<tr>
<td>R-19</td>
<td>33 OHMS</td>
</tr>
<tr>
<td>R-20</td>
<td>75 OHMS</td>
</tr>
<tr>
<td>R-21</td>
<td>1.5 K</td>
</tr>
<tr>
<td>R-22</td>
<td>10 K</td>
</tr>
<tr>
<td>R-25</td>
<td>3.9 K (NEXT TO R-26)</td>
</tr>
<tr>
<td>R-26</td>
<td>4.3 K</td>
</tr>
<tr>
<td>R-27</td>
<td>10 K (OPTIONAL)</td>
</tr>
<tr>
<td>R-28</td>
<td>10 K (OPTIONAL)</td>
</tr>
<tr>
<td>R-29</td>
<td>10 K (OPTIONAL)</td>
</tr>
<tr>
<td>R-30</td>
<td>10 K (OPTIONAL)</td>
</tr>
<tr>
<td>R-31</td>
<td>10 K (OPTIONAL)</td>
</tr>
<tr>
<td>R-32</td>
<td>10 K (OPTIONAL)</td>
</tr>
<tr>
<td>R-33</td>
<td>10 K (OPTIONAL)</td>
</tr>
<tr>
<td>R-34</td>
<td>10 K (OPTIONAL)</td>
</tr>
<tr>
<td>R-35</td>
<td>10 K</td>
</tr>
<tr>
<td>R-36</td>
<td>10 K</td>
</tr>
<tr>
<td>R-37</td>
<td>1.0 K</td>
</tr>
<tr>
<td>R-38</td>
<td>1.0 K</td>
</tr>
<tr>
<td>R-39</td>
<td>220 OHMS</td>
</tr>
<tr>
<td>R-40</td>
<td>1.2 K</td>
</tr>
<tr>
<td>R-41</td>
<td>10 K</td>
</tr>
<tr>
<td>R-42</td>
<td>10 K</td>
</tr>
<tr>
<td>R-43</td>
<td>10 K</td>
</tr>
<tr>
<td>R-44</td>
<td>100 K</td>
</tr>
<tr>
<td>R-45</td>
<td>4.7 K (P/O SERIAL OPTION)</td>
</tr>
<tr>
<td>R-46</td>
<td>220 OHMS</td>
</tr>
<tr>
<td>R-47</td>
<td>330 OHMS</td>
</tr>
<tr>
<td>R-48</td>
<td>10 K</td>
</tr>
<tr>
<td>R-49</td>
<td>10 K</td>
</tr>
<tr>
<td>R-50</td>
<td>220 OHMS</td>
</tr>
<tr>
<td>R-51</td>
<td>220 OHMS</td>
</tr>
<tr>
<td>R-52</td>
<td>4.7 K (P/O SERIAL OPTION)</td>
</tr>
</tbody>
</table>

© 1980 J.B. Ferguson
ALL CAPACITORS NOT INCLUDED IN THE LIST BELOW ARE NON-CRITICAL AND CAN BE ANY DISK OR MONOLITHIC BETWEEN .01 UP TO .1 UF AT 16 VOlTS OR MORE

CAPACITOR VALUES - CROSS REFERENCE

C-21 = 2.2 UF (OR GREATER) 20 V
C-22 = 2.2 UF (OR GREATER) 20 V
C-23 = .01 UF
C-24 = 33 PF
C-37 = 2.2 UF (OR GREATER) 20 V
C-38 = 2.2 UF (OR GREATER) 20 V
C-51 = 2.2 UF (OR GREATER) 20 V
C-52 = 2.2 UF (OR GREATER) 20 V
xC-53 = 180 PF (H. SYNC TIMING)
xC-54 = 4700 PF (V. SYNC TIMING)
C-67 = 2.2 UF (OR GREATER) 20 V
C-68 = 2.2 UF (OR GREATER) 20 V
C-69 = 470 PF (H. SYNC DELAY)
C-106 = 47 PF
C-110 = 100 PF
C-113 = 33 PF
C-114 = .0033 UF
C-115 = 1.0 UF 10 V
C-124 = 390 PF (OPTIONAL)
C-125 = 390 PF (OPTIONAL)
C-126 = 390 PF (OPTIONAL)
C-127 = 390 PF (OPTIONAL)
C-128 = 390 PF (OPTIONAL)
C-129 = 390 PF (OPTIONAL)
C-130 = 390 PF (OPTIONAL)
C-131 = 390 PF (OPTIONAL)
C-132 = 390 PF (OPTIONAL)
C-133 = 390 PF (OPTIONAL)
C-134 = 390 PF (OPTIONAL)
C-135 = 390 PF (OPTIONAL)
C-136 = 390 PF (OPTIONAL)
C-137 = 390 PF (OPTIONAL)
C-138 = 390 PF (OPTIONAL)
C-139 = 390 PF (OPTIONAL)
xC-140 = 68 UF 10 V
xC-141 = 68 UF 10 V
C-142 = 33 PF (PART OF COMPOSITE VIDEO (NEXT TO U-94))

FOR COMPOSITE VIDEO THE FOLLOWING COMPONENTS WILL ASSUME THE CORRESPONDING VALUES:

R-3 = 68 K
R-4 = 100 K
R-5 = 39 K
C-53 = 180 PF
C-54 = 4700 PF
C-69 = 470 PF

c 1980 J.B. Ferguson
ASSEMBLY INSTRUCTIONS

[] Give the PC board a good visual inspection for any obvious defects or shorts. A few minutes spent here could save hours later.

You will note that we have installed a few components on the PCB in areas where there was a high probability of a short. This will eliminate any potential problems you might have in these areas.

[] Using an ohm meter insure that there are no shorts between the -12, GND, +5, and +12 pads located in the lower left corner of the PCB.

[] When doing the following steps please refer to the printed parts overlay which is among the large schematic sheets.

[] Install and solder 14 pin sockets in locations U9, 10, 11, 12, 21, 22, 23, 25, 37, 60, and 71, 72. Note that pin #1 of all sockets is oriented toward the TOP of the PCB.

[] Install and solder 14 pin sockets in locations U52, 53, 56, 66, 76, 77, 78, 79, 81, and 87.


[] Install and solder 14 pin sockets in locations U90, 91, 92, 93, 94, 95, 112, 114, 115, 116, 117 and 118.

[] Install and solder 16 pin sockets in locations U1-8, 13-20, 26-33, and 39-46.

[] Install and solder 16 pin sockets in locations U54, 55, 57-59.

[] Install and solder 16 pin sockets in locations U24, 34, 35, 36, 38, 47, 48, 49, 50, 51, 74, and 75.

[] Install and solder 16 pin sockets in locations U84, 85, 86, 98, 106, and 109.

[] Install and solder 18 pin sockets in locations U61-64 and U107.
[] Install and solder 20 pin sockets in locations U65, 82, and 83.

[] Install and solder 24 pin sockets in locations U67-70, and U73.

[] Install and solder a 28 pin socket in U88.

[] Install and solder 40 pin sockets in locations U80, 89, 102, 111, and 113.

[] Install and solder 11 of the jumper pins in the holes between U92 and U94 (JB1). Install and solder 3 of these pins at JB6 just to the left of U112.

[] Install and solder the .01 mfd. (or greater) bypass caps in locations C1-C20, C25-C36, C39-C50, C55-C66, C70-C105, C107-C109, C111-C112, and C116-C123.

[] Install and solder the 2.2 mfd. (or greater) axial lead tantalum caps (observing polarity!) in locations C21, 22, 37, 38, 51, 52, 67, 68.

[] Install and solder a 1.0 mfd tantalum at C115.

[] After carefully bending the leads to fit, install and solder two 68 mfd. tantalum caps at C140-141.

[] Double check the polarity of the above tantalums!

[] Install and solder C23, a .01 mfd. disc cap.

[] Install and solder C24, a 33 pf. disc cap.

[] Install and solder C106, a 47 pf. cap.

[] Install and solder C110, a 100 pf. cap.

[] Install and solder C113, a 33 pf. cap.

[] Install and solder C114, a .0033 mfd. cap, which may be marked "332".
Please note that C124-139 (390 or 470 pf) are part of the serial I/O option and are not supplied with the basic it.

The following caps are supplied for use only with composite video output (the most commonly used). If you wish to use SPLIT video and sync, refer to the Theory of Operation under "Sync Generation" to calculate these values.

Install and solder C142, a 33 pf cap located next to U94, (Used only for composite video)

Install and solder C53, a 180 pf cap.

Install and solder C54, a 4700 pf cap.

Install and solder C69, a 470 pf cap.

Install and solder R1-R2, 820 ohm resistors.

Install and solder R6-R13, and R19. These are 33 ohms, or any value from 27 to 33 ohms.

Install and solder R14-18, 4.7K ohm resistors. R45 and 52 are also 4.7K but are part of the serial option.

Install and solder R22, 35-36, 41-43, and 48-49 which are 10K ohm resistors. R27-34 are 10K, but are part of the parallel I/O option.

Install and solder R37-38 which are 1K ohm resistors.

Install and solder R39, 46, 50, and 51 which are 220 ohm resistors.

Install and solder R40, which is a 1.2K ohm resistor.

Install and solder R44, which is a 100K ohm resistor.

Install and solder R47, which is a 330 ohm resistor.

Install and solder R20, which is a 75 ohm resistor used for composite video ONLY.
Install and solder R21, which is a 1.5K ohm resistor used for composite video ONLY.

Install and solder R25, which is a 3.9K ohm resistor used for composite video ONLY. R25 is located next to R26.

Install and solder R26, which is a 4.3K ohm resistor. This is used for composite video ONLY.

The next 3 resistor values listed are for composite video (sync timing). For SPLIT video refer to the Theory of Operation for calculation of these values.

Install and solder R3, which is a 68K ohm resistor.

Install and solder R4, which is a 100K ohm resistor.

Install and solder R5, which is a 39K ohm resistor.

Install and solder Q1, a 2N2222, located near U74.

Install and solder Q2, a 2N2907, located near U80.

Install and solder Y1, a 14.318 mhz crystal.

Install and solder Y2, a 20.000 mhz crystal.

Y3, a 5.0688 mhz crystal, is part of the serial option.

Install and solder a 26 pin connector in location J2. Note that J3, J4 are also 26 pin conn, but are part of the serial I/O option. The side with the SHORTER pins is soldered to the PCB.

All 40 pin connectors are for options. These are located in JB4, JB5, (serial), and J5 (parallel).

Install and solder the 50 pin connector in location J1. This is the floppy disk connector.

Install and solder two 1K 8pin resistor packs in
locations RP1-2. Pin 1 is denoted by a dot on the PCB and should be matched with the dot or indentation on the resistor pack. Install the packs tilted at a slight angle away from connector J1 to allow for clearance with the mating floppy disk connector.

[] Install and solder TB1, the power connector for the Big Board.

[] Install and solder VR1, a 1N751 zener, located near U98.

[] Install and solder the 10 pin connector at location J6. This is the video connector.

[] Note that the 8 pin connector at JB2 is part of the CTC option. The 16 pin connector at JB3 is part of the parallel I/O option.

[] Before inserting any IC's in their sockets, apply +5, +12, -12, and ground to TB1. With an accurate voltmeter, check the voltage on C21. It should be -5VDC (within 5%). When measuring voltages, be very careful not to short your probes to any of the grounds on the PCB, this would short out your power supplies.

[] Check the voltage across C37, it should be +12VDC (within 5%).

[] Check the voltage across C91, it should be +5VDC (within 5%).

[] Check the voltage between pins 23 and 2 on connector J2, it should measure -12VDC (within 5%).

[] The power supplies used to run the Big Board MUST be of the highest quality and should have Over Voltage Protection (OVP) along with current limiting. A cheap power supply could become very expensive if you have to replace all of the IC's on a Big Board!!!

[] DISCONNECT ALL DC POWER NOW !!!

[] Install 74LS00's in locations U52,53. Note that all pin #1's are indicated by a notch or indentation in the device. All pin #1's are toward the TOP of the PCB.
[] Install the 74LS02 in location U87.

[] Install 74LS04's in locations 11, 66, 77, 103, 110.

[] Install a 7406 in location U105.

[] Install 74LS08's in locations 37, and 104.

[] Install a 74LS10 in location U10.

[] Install a 74LS14 in locations U101, 112, and 114.

[] Install a 74LS20 in U56.

[] Install 74LS32's in U25, 81.

[] Install a 7445 in U109.

[] Install 74LS74's in U9, 12, 60, 108.

[] Install 74LS123's in U38, 51, and 106.

[] Install a 74LS136 at U94. (For SPLIT video use a LS86).

[] Install 74LS138's in U55, 84, 85, and 86.

[] Install a 74LS151 in U75.

[] Install 74LS157's in U47, 48, 49, and 50.

[] Install 74157's in U58 and 59.

[] Install a 74LS161 in U24.

[] Install a 74164 in U76.

[] Install 74LS174 in U35, and 74.

[] Install a 74LS193 in U98.
Install 74LS241's in U65 and 82.

Install 74LS242's in U99 and 100.

Install 74LS243's in U71, 72, 78, 79.

Install 74LS283's in U34 and 36.

Install 74LS290's in U22 and 97.

Install a 74LS293 in U96.

Install a 74LS373 in U83. (can be 74S373).

Install 74LS393 in U21 and 23.

Install 8216's in U54 and 57.

Install 2114's (250 NS or better) at U61-U64.

Install the CHAR ROM at U73.

Install the PFM Rom at U67.

Install the Z80 at U80.

Install the PIO at U111.

Install the WD1771 at U102.

Install the 32 4116's at U1-8, 13-20, 26-33, 39-46. Must be 300 NS or better.

DO NOT continue until you have double checked that all the IC's are in the correct sockets!

Use the cable and 26 pin connector assembly provided to connect an ASCII encoded keyboard to the Big Board. Refer to the "Keyboard Connector Pin Assignments" (J2) in the connector
assignment section of this manual. Note that either polarity of strobe is available at JB6 as described under the "Keyboard Strobe Polarity Strapping Option". The needed DC power is also available at J2.

[ ] Connect the jumpers for JB7 "Video Strapping Options" as shown for composite video. JB7 is located between U92 and U94.

[ ] Connect a composite type video monitor (must be high bandwidth, i.e. NOT a converted TV) to J6 pins 9 and 10. We recommend 75 ohm coax such as RG174. Note that pin 9 is GND.

[ ] Connect a normally open push button switch to TB1 pins 5 and 6. This is the system reset.

[ ] We are about to apply final power! This is the last chance you will have to RECHECK YOUR WORK!

[ ] Reconnect the power to TB1. ALL power supplies MUST be turned on AT THE SAME TIME.

[ ] Press the RESET button. The video monitor should be blank.

[ ] Now hit the RETURN key on your keyboard. PFM system monitor should sign on. Now is the time to familiarize yourself with PFM. Carefully read the PFM users manual.

[ ] We strongly recommend that you now use PFM to test your memory for at least 4 to 8 hours. The more you test the memory, the more reliable your system will be. The PFM memory test may seem slow, but this is due to the exhaustive nature of the test procedure. We have found that most Big Board problems are nothing more than memory problems. This is so, even though we provide the best quality memories available to us. Remember, that it only takes one bad bit out of the total 524,288 bits to cause the system to malfunction. Note that because PFM is executing out of RAM, test only from 0000 to EFFF hex so that you will not kill PFM.

[ ] After testing the 60K as indicated above, power down the system, exchange U1-8 with U39-46. And retest memory. This way ALL rams are fully tested.

[ ] IF the video does not clear, the trouble can be either in the CPU section or the video section. IF PFM does not sign on and the video appears to be working, the problem is more likely RAM related or improper keyboard hook up. IF characters DO appear, but are not sharp and whole, then look for problems in
the video divider chain. The video section is independent. The only thing it shares with other parts of the board is the +5VDC, GND, and 2114 memories.

[] Assuming that the computer is operating correctly, it is now time to connect the floppy disc drives.

[] Assuming that a Shugart SA800 compatible drive is being used, connect power to the drives. Using the 50 conductor cable supplied, attach a crimp on (IDC) connector that matches your drive to the bare end of the cable. Plus this end onto the floppy disk drive (noting pin #1). The other end of the cable is connected to J1 on the Big Board. Please be careful to observe that pin #1 is connected properly. All odd numbered pins are grounded, so a connector connected backwards could be trouble!

[] If you are connecting more than one drive, make sure that the drive select option (on the drive PCB) is jumpered correctly. Most new drives are received jumpered as drive #0. There should be only ONE drive #0, #1, etc. Refer to the drive data manual for additional information.

[] If you do not have a Shugart SA800 compatible drive, then you will have to carefully examine your drive documentation along with the Big Board schematics in order to correctly interconnect the two. Usually the incompatibility lies in the fact that some of the non-standard drive signals are located on different pins.

[] Now apply DC power to the drives and Big Board along with AC power to the drives (if required).

[] Hit the RETURN key to sign on PFM. Exercise PFM some to insure that the computer is STILL operating properly.

[] Insert in drive 0 an IBM compatible diskette that you consider EXPENDABLE (just in case something goes wrong). DO NOT use your Big Board CP/M diskette. Use PFM command R,00,01,01 to see if data can be read from the diskette and that the head on the drive did load. If a disk error is indicated, refer to the PFM User Manual for an explanation.

[] Using PFM again, type R00,40,01. This command will cause the head to seek to track 40 hex. You should be able to hear this head seeking action. If the head seeks, and the data is read, then the disk controller is probably working correctly.

[] We recommend that you find another IBM compatible disk
system (a friend or a computer store) that can be used to make a 
Back Up copy of your CP/M system diskette. This copy MUST be a 
true byte for byte copy.

[ ] Insert a copy of the CP/M system diskette (Big Board 
version) in drive 0. Type B (and a RETURN) in PFM to Boot in 
CP/M. CP/M should sign on now.

[ ] If CP/M does not come up, then make sure that you have 
indeed inserted a CP/M (2.2) with a Big Board BIOS.

[ ] Also, if CP/M does not come up, you can use the PFM Read 
command to verify that track 0 sector 1 does contain the start 
of CP/M (a loader).

[ ] After CP/M is up, refer to their manuals in order to 
learn more about the CP/M commands.
BIG BOARD OPTIONS

REAL TIME CLOCK (CTC)

[ ] Install and solder an 8 pin connector at JB2.

[ ] Install and solder a 28 pin socket at U88.

[ ] Install the CTC chip at U88.

[ ] Refer to "CTC Strapping and I/O Assignments" in the theory of operation.

SERIAL I/O OPTIONS

[ ] Install and solder a 40 pin socket at U113.

[ ] Install and solder four 14 pin sockets at U115-118.

[ ] Install and solder 4.7K ohm resistors at R45, 52.

[ ] Install and solder sixteen 390 or 470 pf caps at C124-139.

[ ] Install and solder 2 40 pin connectors at JB4, 5.

[ ] Install and solder two 26 pin connectors are J3, 4.

[ ] Install two MC1488's at U117, 118.

[ ] Install two MC1489 at U115, 116.

[ ] Install the S10/0 at U113.

[ ] Refer to the Jumpers shown under "Serial I/O Strapping Options". Also refer to "Serial I/O Connector Pin Assignments" and the theory of operation.

[ ] A typical application would be to connect a three wire serial device (such as an ADM-3A) to serial channel B. This would require jumpers between pins 5-6 and 9-10 on JB5. This properly directs data to and from the S10 and terminal through the RS232 buffers.
Now that a serial device is connected, refer to the PFM users’ manual for information as to assigning this device as the console I/O.

PARALLEL I/O OPTION

[ ] Install and solder R27-34, which are 10K ohm resistors.

[ ] Install and solder a 40 pin socket at U89.

[ ] Install and solder five 14 pin sockets at U90-93, and 95.

[ ] Install and solder a 40 pin connector at J5.

[ ] Install and solder a 16 pin connector at JB3.

[ ] Install a 74LS86 in location U95.

[ ] Install four 74LS243’s in U90-93.

[ ] Install the PIO at U89.

[ ] Refer to the "General Purpose PIO Strapping and Connector Pin Assignments" for necessary information. Also read the theory of operation for additional data.

SPLIT VIDEO OPERATION

[ ] Refer to "Video Strapping Option" and the theory of operation for additional information. Also read the theory of operation regarding the video section.
THEORY OF OPERATION

CENTRAL PROCESSOR (SHEET 1)
--------------------------------

CLOCK GENERATOR:

All the system clocks with the exception of the baud clock and the
video dot clock are generated from a master oscillator operating at 20
Mhz. Part of inverter U-77, biased into the linear region by 2000 ohms of
feedback, serves as the active element of the oscillator. Another
inverter, in the same package, buffers the output to prevent undesirable
feedback and serves to lower the output impedance of the oscillator.

The 20 Mhz clock is scaled by the divide-by-5 section of decade
counter U-97 to provide 4 Mhz for use in the floppy disk data separator.
The 2 Mhz clock for the disk controller is generated from the 4 Mhz clock
by the remaining divide by two section of U-97.

The 2.5 Mhz processor clock is generated by dividing the master 20
Mhz clock by 8 with binary counter U-96. The output of the third stage is
buffered by inverter U-77 and transistor Q-1 to provide the rail to rail
clock required by the Z-80.

The column address strobe "CAS", and the address multiplexer
control "MUXC", are derived from the 20 Mhz clock. When memory request
"MREQ" is low and refresh "RFISH" is high, generation of "CAS" and "MUXC"
is enabled. "RFISH" disables the generation of "CAS" and "MUXC" by holding
shift register U-76 reset. This is done to take advantage of the low
power row address strobe "RAS" only refresh mode of the 16 K dynamic RAMs.

RESET CONTROLLER:

Two types of reset take place on the board. Power on reset is
detected and conditioned by part of hex schmitt inverter U-101. The
pushbutton reset is also conditioned by a part of hex schmitt inverter
U-101. The "D" type flip flop U-108 synchronizes the pushbutton reset
with machine cycle one "M1" from the processor. The output of the flip
flop triggers a 12 microsecond one shot U-106. Power on reset and
pushbutton reset are or'ed together by U-81 and inverted by U-103 for use
by the processor. The reset pulse is negative or'ed with "M1" by U-104 to
generate a reset for the Z-80 family programmable I/O devices. A low on
"M1" with read "RD" high will reset a PIO.

BUS BUFFERING:

Octal buffer U-82 buffers the control signals generated by the
processor for use throughout the system. Quad transceivers U-78 and U-79
mediate data transfers to and from memory. Two portions of U-87 control
the direction of the data bus transceivers. During a memory read the data
transceivers allow data from memory through to the processor, otherwise
the processor always drives memory. Octal buffer U-65 drives the lower 8

© 1980 J.B. Ferguson
bits of the address bus. The octal latch U-83 serves a dual function. As well as buffering the upper 8 bits of the address bus, the latch holds the address bus stable during the active portion of the memory request cycle "MREQ". During the latter portion of the "MREQ" cycle the Z-80 allows the address bus to change. This could generate a false "RAS" to the dynamic RAM if it were not held stable.

READ ONLY MEMORY:

The board can accommodate up to 8 K of 2716's.

U-67 RESIDES FROM 0000 HEX TO 07FF HEX
U-68 RESIDES FROM 0800 HEX TO 0FFF HEX
U-69 RESIDES FROM 1000 HEX TO 17FF HEX
U-70 RESIDES FROM 1800 HEX TO 1FFF HEX

The description of the bank switching technique will be covered with the 64 K RAM theory of operation.

PORT ADDRESS DECODING:

Octal decoder U-85 is used to select the appropriate I/O device based on the binary value of the address bits A2, A3, & A4. When A7 is low and "MIR" is high, a low on "IORQ" will cause the appropriate output of the decoder to go low, selecting the I/O device for a read or write operation.

PORT 0-3 = CHANNEL A BAUD RATE (WRITE ONLY)
PORT 4 = SIO CHANNEL A DATA
PORT 5 = SIO CHANNEL B DATA
PORT 6 = SIO CHANNEL A CONTROL
PORT 7 = SIO CHANNEL B CONTROL
PORT 8 = GP PIO PORT A DATA
PORT 9 = GP PIO PORT A CONTROL
PORT A = GP PIO PORT B DATA
PORT B = GP PIO PORT B CONTROL
PORT C-F = CHANNEL B BAUD RATE (WRITE ONLY)
PORT 10 = 1771 STATUS/COMMAND REGISTER
PORT 11 = 1771 TRACK REGISTER
PORT 12 = 1771 SECTOR REGISTER
PORT 13 = 1771 DATA REGISTER
PORT 14-17 = CRT SCROLL REGISTER (WRITE ONLY)
PORT 18 = CTC CHANNEL 0
PORT 19 = CTC CHANNEL 1
PORT 1A = CTC CHANNEL 2
PORT 1B = CTC CHANNEL 3
PORT 1C = SYSTEM DATA PORT
PORT 1D = SYSTEM CONTROL PORT
PORT 1E = KEYBOARD DATA PORT
PORT 1F = KEYBOARD CONTROL PORT

DISK TRANSFER SYNCHRONIZATION:

In order to successfully execute the high speed data transfers between the processor and the disk controller, the fast Z-80 non maskable

© 1980 J.B. Ferguson
interrupt "NMI" response was employed. During reads and writes to and from the disk controller, the data at memory location 66 hex is retrieved and stored. This location is overwritten with a RETURN instruction. After this setup is accomplished the processor executes a HALT instruction. When the processor is in a HALT condition, a DATA REQUEST (DRQ) or an INTERRUPT REQUEST (IRQ) from the disk controller will cause a non-maskable interrupt to be generated. The processor then executes the RETURN instruction at 66 hex and returns to transfer the data to or from the disk controller. When the 128 byte transfer is complete the old data is restored and the processor resumes normal operation. This hardware assistance obviated the necessity for a DMA device by eliminating the disk controller "DRQ" status test.

CRT DISPLAY CONTROLLER (SHEET 2)

VIDEO CLOCK GENERATION:

Three inverters from U-11 are used to generate the video dot clock. The 14.31818 Mhz dot clock is divided by 7 to develop the character clock. Synchronous binary counter U-24 is preloaded with a binary 9 at each top count to accomplish the divide by 7 function. The character clock is divided by 128 by the 8 bit binary counter U-23 to develop the scan clock. In the process of developing the scan clock the intermediate outputs of U-23 develop part of the character address for the video RAM. Decade counter U-22 divides the scan clock by 10, simultaneously developing the line clock and the vertical component of the character matrix address. U-21 and part of U-9 work in conjunction to generate the frame clock and the line address for the video RAM. The two devices divide the line clock by 26 to generate the 60 hz frame clock. The second half of U-21 divides the frame clock by 16 to develop the 4 hz blink clock.

VIDEO SCROLLING:

In order to eliminate the delay associated with software scrolling, hardware assistance was employed. For ease of understanding, the CRT RAM resides from 3000 hex to 3FFF hex. Writing into the scroll register adds an offset to the line address developed by the line counter. The net effect is similar to the rotation of a cylinder whose axis is horizontal and perpendicular to the line of sight. The amount of rotation is determined by the magnitude of the number contained in the scroll register. For instance, an offset of zero puts the data at location 3000 hex (of the CRT memory) at the bottom of the screen. If the offset was one, the data at 3000 hex would be displayed on the line next to the bottom. An offset of seventeen hex (23 decimal) puts the data at location 3000 hex at the top of the screen. U-35 is the scroll register. It's contents are added, modulo 24, to the line address by adders U-36 and U-34 and part of and gate U-37.

VIDEO RAM ADDRESSING:

Multiplexers U-47, U-48, and U-49 select the source of the addresses for the video RAM. If the processor is doing a read or write to
video RAM "CRTCE" (CRT memory access enable) will go low. When "CRTCE" goes low, the address from the processor is selected instead of the address generated by the counter chain. This gives the processor access to the video RAM for read or write operations. U-50 maps the 12 bit address developed by the counter chain into the 2 K byte video RAM.

SYNC GENERATION:

Horizontal sync is generated by decoding the 80th count of the character counter U-23. The detection of this count triggers the horizontal delay one shot U-51. The period of the delay is determined by the formula:

\[
(Tw = 0.45 \times Rt \times Cext)
\]

Rt is in K Ohms, Cext is in Pf, and Tw is in ns.

the limits on Rt are 250 K > Rt > 5 K.

When the horizontal delay one shot times out, the H sync one shot is triggered. The formula for determining the timing components for the H sync one shot are the same as those for the horizontal delay. The vertical sync is generated between counts 24 and 26 of the line counter. The decode of line count 24 triggers one shot U-38. The duration of the vertical sync pulse is also user selectable via proper selection of R-4 and C-54. The formula for selection of these components is the same as the one used for the horizontal delay.

CPU ACCESS OF VIDEO RAM:

During read or write operations involving the video RAM and the CPU, "CRTCE" will go low. When "CRTCE" goes low the processor address bus is selected by multiplexers U-47 - U-49 as the address source for the video RAM. A low on "CRTCE" is also used as a term in the direction control logic for data bus access. Decoder U-86 controls the direction and activity of transceivers U-71 and U-72. During a processor read operation, data from the video RAM at the specified address is allowed onto the processor data bus. During a processor write operation, data from the processor is written to the video RAM at the specified address.

VIDEO GENERATION:

While in the display mode, ASCII data from the video RAM and scan address data from decade counter U-22 are used to select the proper dot patterns from the character generator U-73. The dot information from the character generator is sampled by hex "D" flip flop U-74 at the next character time. While the next character is being accessed, the previous dot pattern is multiplexed out of U-74 by multiplexer U-75. Multiplexer U-75 feeds the video driver U-94.

DISPLAY BLANKING:

The display is blanked during horizontal retrace, vertical retrace, CPU access, and decode of scan counts 8 & 9. Blanking is accomplished by disabling the character generator.
64 K RAM AND BANK SWITCHING (SHEET 3)
--------------------------------------

RAM ADDRESS MULTIPLEXING:

The address from the processor is multiplexed to the RAM array by multiplexers U-58 and U-59. During a memory access the row address is presented to the array first. After the row address is stable the decode of A15B and A14B gated by "MREQ", generates the proper row address strobe. The decode of A15B and A14B is accomplished by octal decoder U-84. Nand gate package U-52 gates the decoder outputs with "MREQ" to generate the "RAS" for the appropriate 16 K block. After the proper setup and hold times for the row address have been met, "MUXC" goes high. A high on "MUXC" switches the column address on to the RAM array. After the setup time is met "CAS" goes low and latches the column address into the 16 K block that received the "RAS". If the memory is being read, the data from the RAMs will be gated onto the data bus by transceivers U-54 and U-57. If the memory is being written to, data is routed from the processor's data bus to the RAM array.

REFRESH:

During the refresh cycle, the Z-80 places the refresh address on the lower 7 bits of the address bus. When this address is stable in the RAM array, the "RFSH" pin on the Z-80 goes low. The active low "RFSH" generates a "RAS" on all RAMs via nand gate packages U-52 and U-53. An active "RFSH" disables the generation of both "CAS" and "MUXC".

BANK SWITCHING:

Bit 7 of port 1C hex is the bank switch control. When the output is high, the ROMs and the CRT display appear in the lower 16K block. When bit 7 of port 1C hex is low, all the 64K RAM is available to the processor.

FLOPPY DISK CONTROLLER, SYSTEM PIO, AND CTC (SHEET 4)
--------------------------------------------------------

FLOPPY DISK CONTROLLER:

The 1771 (U-102) performs all the control functions required to interface to a floppy disk drive. The only support required by the 1771 is external data separation, inverting data bus transceivers, head load timer, and buffering to and from the drive(s).

DATA SEPARATOR:

Presettable counter U-98 is used as a digital monostable with the timing reference developed by the system clock. Raw data coming from the disk drive is used to preload the counter. If the counter does not receive a data bit between clocks the counter in effect times out and

C 1980 J.B. Ferguson
presents the controller with a logic zero. If the counter receives data between clocks, the controller will see a logic one on its data input.

HEAD LOAD TIMING:

When the 1771 activates the head load output, monostable U-106 is triggered. The 1771 samples the "HLT" until a logic one is detected. At this time the head is assumed to be loaded and stable. This time can be altered to accomodate the timing characteristics of different drives. The formula for determining this time is the same as that used for the horizontal delay mono in the CRT controller.

DATA BUS BUFFERING:

Inverting transceivers U-99 and U-100 adapt the 1771 to the non-inverted Z-80 data bus. During a read operation, data from the 1771 is allowed onto the processor's data bus. Otherwise the processor's data bus always drives the 1771's data inputs.

CONTROL BUS BUFFERING:

U-105, part of U-82, and U-101 buffer the control, status and data to and from the 1771. In addition to buffering and isolation, U-101 and U-82 provide schmitt trigger characteristics for noise rejection. Refer to 1771 data sheet for detailed programming information.

CTC:

The CTC resides at ports 18 hex through 1B hex. All the inputs and outputs associated with the CTC are available to the user at JB-2. Refer to the strapping option section for pin assignments. Refer to the CTC data sheet for detailed programming information.

SYSTEM PIO:

The system PIO resides at ports 1C hex through 1F hex. The "A" side of the system PIO controls the floppy disk drive select, bank switching, disk power switching, sensing keyboard data available (for polled keyboard applications), and an uncommitted user definable I/O bit. The bit allocations are as follows:

BITS 0, 1, AND 2 CARRY THE DRIVE NUMBER IN BINARY
BIT 3 IS USED FOR KEYBOARD DATA AVAILABLE
BIT 4 IS USER DEFINABLE
BIT 5 IS AN OPTIONAL 'BEEPER' OUTPUT
BIT 6 CONTROLS A SOLID STATE RELAY FOR DRIVE A.C.
BIT 7 CONTROLS THE BANK SWITCHING (0=RAM)

The drive select information should be presented to the system PIO in the following manner:

read port 1C hex to maintain system status data
mask off the lower 3 bits
"or" in the desired drive number in binary format
write the modified data to port 1C hex

The "B" side of the system PIO is devoted to the keyboard. The keyboard port is seven bits wide and is fully buffered by schmitt inverters. After any reset, the CPU examines bit eight of the keyboard port (J-2 PIN 15) to see if a keyboard is present. If bit eight of the keyboard connector (J-2) is at ground potential, the keyboard will be assigned as the console device. If no keyboard is present, the SIO channel 'B' will be assigned as the console. If the console device is assigned to the serial channel, depressing the carriage return key after reset will automatically set the baud rate.

The keyboard strobe polarity is user selectable via JB-6 strapping option. Refer to the strapping option sheet for application information.

GENERAL PURPOSE PIO AND SIO (SHEET 5)
---------------------------------------

The G.P. PIO provides the user with 16 bits of user definable input or output or a mix of input and output on nibble boundaries. The G.P. PIO resides at ports 08 hex - 0B hex. The PIO will support all modes of interrupt supported by the Z-80. For detailed programming information refer to the Z-80 PIO data sheet. For applications information, refer to the strapping option section.

SIO:

The Z-80 SIO supports two full channels of serial I/O with the capability of supporting full RS-232 protocol on both channels. In addition, the A side of the SIO can provide clocks to synchronous modems or receive clocks from the modem. Both channels of the SIO can be configured to interface to a modem or a terminal. Refer to the strapping option sheets for detailed instructions. Refer to the SIO data sheet for programming information. After programming the SIO to perform synchronous data communication with one of the involved protocols, you are eligible for an honorary degree in high computereese. Otherwise custom programming services will be available at standard consulting rates from:

J.B. FERGUSON
1705 HOMEMAKER HILLS DR.
ARLINGTON, TEXAS  76010

The "SYNC" pins on both channels of the SIO have been connected to the Rx data pins to facilitate baud rate selection. Utilizing this feature, the start bit duration can be timed, and the baud rate can be set accordingly. If you wish to use these pins for their origional intentions, the straps can be cut with no ill effect.

BAUD RATE GENERATOR:

The COM 8116 provides the user with two programmable baud rate generators. Channel A baud rate resides at port 00 hex and is write only.
Channel B baud rate resides at port 0C hex and is also write only. The programming procedure is as follows:

00 hex = 50 Baud
01 hex = 75 Baud
02 hex = 110 Baud
03 hex = 134.5 Baud
04 hex = 150 Baud
05 hex = 300 Baud
06 hex = 600 Baud
07 hex = 1200 Baud
08 hex = 1800 Baud
09 hex = 2000 Baud
0A hex = 2400 Baud
0B hex = 3600 Baud
0C hex = 4800 Baud
0D hex = 7200 Baud
0E hex = 9600 Baud
0F hex = 19.2 Kbaud

INTERRUPT STRUCTURE:

All the Z-80 family devices on this board are capable of supporting mode 0, 1, and 2 interrupts. Mode 2 interrupts are used in the monitor delivered with the system. The I register in a unmodified system is loaded with OFF hex. The priority chain is organized high to low as follows:

SIO CHANNEL A
SIO CHANNEL B
SYSTEM PIO PORT A
SYSTEM PIO PORT B
GP PIO PORT A
GP PIO PORT B
CTC CHANNEL 0
CTC CHANNEL 1
CTC CHANNEL 2
CTC CHANNEL 3

POWER SUPPLY REQUIREMENTS:

The board requires three power supplies:

+5.0 volts at 3 amps
+12.0 volts at .25 amps
-12.0 volts at 0.20 amps

It is STRONGLY recommended that quality power supplies be used. It is also recommended that over voltage protection be employed. The OVP should be set at 1.25 times the rated voltage. for instance the 5V OVP should be set at 6.2V.
### GENERAL PURPOSE PIO STRAPPING (JB4) AND CONNECTOR PIN ASSIGNMENTS (J5)

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>all odd numbered</td>
<td>o o</td>
<td>port A STROBE</td>
</tr>
<tr>
<td>pins are grounded</td>
<td>o o</td>
<td>port A READY</td>
</tr>
<tr>
<td></td>
<td>o o</td>
<td>port A bit 0</td>
</tr>
<tr>
<td></td>
<td>o o</td>
<td>port A bit 1</td>
</tr>
<tr>
<td></td>
<td>o o</td>
<td>port A bit 2</td>
</tr>
<tr>
<td></td>
<td>o o</td>
<td>port A bit 3</td>
</tr>
<tr>
<td></td>
<td>o o</td>
<td>port A bit 4</td>
</tr>
<tr>
<td></td>
<td>o o</td>
<td>port A bit 5</td>
</tr>
<tr>
<td></td>
<td>o o</td>
<td>port A bit 6</td>
</tr>
<tr>
<td></td>
<td>o o</td>
<td>port A bit 7</td>
</tr>
<tr>
<td></td>
<td>o o</td>
<td>port B READY</td>
</tr>
<tr>
<td></td>
<td>o o</td>
<td>port B STROBE</td>
</tr>
<tr>
<td></td>
<td>o o</td>
<td>port B bit 0</td>
</tr>
<tr>
<td></td>
<td>o o</td>
<td>port B bit 1</td>
</tr>
<tr>
<td></td>
<td>o o</td>
<td>port B bit 2</td>
</tr>
<tr>
<td></td>
<td>o o</td>
<td>port B bit 3</td>
</tr>
<tr>
<td></td>
<td>o o</td>
<td>port B bit 4</td>
</tr>
<tr>
<td></td>
<td>o o</td>
<td>port B bit 5</td>
</tr>
<tr>
<td></td>
<td>o o</td>
<td>port B bit 6</td>
</tr>
<tr>
<td></td>
<td>o o</td>
<td>port B bit 7</td>
</tr>
<tr>
<td>15 16</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

---all ground---

If the "port A STROBE polarity" control pin (JB-4 pin 7) is not connected to JB-4 pin 8, a logic 1 on the "port A STROBE" input (J-5 pin 2) will produce a logic 0 on the PIO "PASTB" input (U-89 pin 16). If JB-4 pin 7 is connected to JB-4 pin 8 no inversion will occur. JB-4 pin 4 and JB-4 pin 3 affect "PBSTB" in exactly the same manner.

If the "port A READY polarity" control pin (JB-4 pin 2) is not connected to JB-4 pin 1, a logic "0" on the PIO "PARDY" output (U-89 pin 18) will produce a logic "1" on the "PARDY" output at J-5 pin 4. If JB-4 pin 1 is connected to JB-4 pin 2 no inversion will occur. JB-4 pin 5 and JB-4 pin 6 affect "PBDRDY" in exactly the same manner.

If the "port A upper direction" control pin (JB-4 pin 15) is not connected to JB-4 pin 16, J-5 pins 14, 16, 18, and 20 will all be inputs. The data on these pins will be applied to PIO pins 10, 9, 8, and 7 respectively. If U-90, U-91, U-92, or U-93 is an 74LS243, the data will not be inverted. If inversion is desired, a 74LS242 should be used.

If the "port A upper direction" control pin (JB-4 pin 15) is connected to JB-4 pin 16, J-5 pins 14, 16, 18, and 20 will all be outputs. The data at PIO pins 10, 9, 8, and 7 will be reflected at J-5 pins 14, 16, 18, and 20 respectively. All direction control strapping options operate in the same manner.

If the PIO is to be used as an output device, the appropriate direction options must be exercised. On power on, and all successive resets, the PIO will revert to the "input" mode. Refer to the PIO data sheet for programming procedure.

© 1980 J.B. Ferguson
CTC STRAPPING AND I/O ASSIGNMENTS (JB2)
------------------------------------------

2 1
SYSTEM CLOCK o o CLOCK/TRIGGER 0
ZC-T00 o o CLOCK/TRIGGER 1
ZC-T01 o o CLOCK/TRIGGER 2
ZC-T02 o o CLOCK/TRIGGER 3
 8 7

VIDEO OUTPUT CONNECTOR PIN ASSIGNMENTS (J6)
------------------------------------------

all ground
9  o  o  o  o  o  1
10 o  o  o  o  2
---separate video
---separate h.sync
---separate v.sync
------n.c.
----------composite video

VIDEO STRAPPING OPTIONS (JB1)
------------------------------------------

1  o  ground
2  o  h.sync inversion control
3  o  h.sync output
4  o  composite video strap
5  o  composite video strap
6  o  v.sync output
7  o  ground
8  o  v.sync inversion control
9  o  ground
10 o  video inversion control
11 o  ground

with no strapping options exercised:
    inverted video
    negative h.sync
    negative v.sync
for positive h.sync connect pin 1 to pin 2
for positive v.sync connect pin 8 to pin 9
for positive video connect pin 10 to pin 11
for composite video:
    connect pin 10 to pin 11
    connect pin 3 to pin 4
    connect pin 5 to pin 6

c 1980 J.B. Ferguson
install R-21, R-25, R-26, Q-1, and R-20
install a 33 pf capacitor next to R-21

SERIAL I/O CONNECTOR PIN ASSIGNMENTS (J3 = B CHAN, J4 = A CHAN)

----------
<table>
<thead>
<tr>
<th>DCD</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROTECTIVE GROUND</td>
</tr>
<tr>
<td>DSR</td>
</tr>
<tr>
<td>CTS</td>
</tr>
<tr>
<td>RTS</td>
</tr>
<tr>
<td>RX DATA</td>
</tr>
<tr>
<td>TX DATA</td>
</tr>
<tr>
<td>PROTECTIVE GROUND</td>
</tr>
</tbody>
</table>
13 o o o o o o o o o o o o o o 1
26 o o o o o o o o o o o o 14
----------
| TX CLOCK |
| RX CLOCK |
| DTR     |

SERIAL I/O STRAPPING OPTIONS (JB4 = A CHAN, JB5 = B CHAN)

ONLY CHANNEL A IS CAPABLE OF UTILIZING BAUD CLOCKS FROM AN EXTERNAL DEVICE OR OF PROVIDING BAUD CLOCKS TO AN EXTERNAL DEVICE. WHEN PROVIDING THE BAUD CLOCK TO THE EXTERNAL DEVICE, THE SIO MUST USE THE SAME CLOCK SOURCE.

----------
| BAUD RATE GENERATOR SUPPLIES MODEM WITH TX CLOCK |
| MODEM SUPPLIES SIO WITH TX CLOCK |
| MODEM SUPPLIES SIO WITH RX CLOCK |
| BAUD RATE GENERATOR SUPPLIES SIO WITH TX CLOCK |
| BAUD RATE GENERATOR SUPPLIES SIO WITH RX CLOCK |
39 o o o o o o o o o o o o o o o o o o o o 1
40 o o o o o o o o o o o o o o o o o o 2
----------
| SIO CHAN A |
| ONLY |

----------
| (M) TXD TO PIN 3 |
| (T) TXD TO PIN 2 |
| (M) RXD FROM PIN 2 |
| (T) RXD FROM PIN 3 |
| (M) CTS TO PIN 5 |
| (T) RTS TO PIN 4 |
| (M) RTS FROM PIN 4 |
| (T) CTS FROM PIN 5 |
| (M) DCD TO PIN 8 |
| (T) DTR TO PIN 20 |
| (M) DTR FROM PIN 20 |
| (T) DCD FROM PIN 8 |

DATA SET READY IS ACTIVE ON BOTH CHANNELS

C 1980 J.B. Ferguson
LEGEND

(M) INDICATES MODEM (DATA COMMUNICATIONS EQUIPMENT) FUNCTION
(T) INDICATES TERMINAL (DATA TERMINAL EQUIPMENT) FUNCTION

FOR INSTANCE, EXERCISING THE (T) STRAP OPTIONS WILL ALLOW COMMUNICATION WITH A MODEM. EXERCISING THE (M) STRAP OPTIONS WOULD ALLOW COMMUNICATION WITH A TERMINAL.

TXD=TRANSMITTED DATA
RXD=RECEIVED DATA
RTS=REQUEST TO SEND
CTS=CLEAR TO SEND
DTR=DATA TERMINAL READY
DCD=DATA CARRIER DETECT

KEYBOARD STROBE POLARITY STRAPPING OPTION
---------------------------------------------

○ ○ ○ ○
JB6

JB6 IS LOCATED DIRECTLY BELOW U111 (KEYBOARD PIO)
STRAP CENTER PAD TO LEFT PAD FOR POSITIVE STROBE
STRAP CENTER PAD TO RIGHT PAD FOR NEGATIVE STROBE

KEYBOARD CONNECTOR PIN ASSIGNMENTS (J2)
----------------------------------------

<table>
<thead>
<tr>
<th>BIT 7</th>
<th>BIT 8</th>
<th>STROBE</th>
<th>+12V</th>
<th>-12V</th>
<th>+5V</th>
<th>25</th>
<th>26</th>
<th>PINS 2, 4, 6, 8, 10, 12, 14, 16</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ARE ALL GROUND</td>
</tr>
</tbody>
</table>

C 1980 J.B. Ferguson
THEORY OF OPERATION

DISK CONNECTOR PIN ASSIGNMENTS (J1)
--------------------------------------
SELECT 2-----------------------------SELECT 3
SELECT 1---------------------------STEP IN
SELECT 0---------------------------STEP
READY-----------------------------WRITE DATA
INDEX-------------------------------WRITE
HEAD LOAD------------------------TRACK 0
LOW CURRENT----------------------WRITE PROTECT
                                   50L RAW DATA
                                   49

ALL ODD NUMBERED PINS ARE GROUNDED

POWER CONNECTOR PIN ASSIGNMENTS (TB1)
--------------------------------------
-12V----------------- 1  2  3  4  5  6  7  8
GND------------------ DISK AC CONTROL
+5V------------------RESET RETURN
+12V------------------RESET

C 1980 J.B. Ferguson
PFM-80 USERS MANUAL

--INTRODUCTION--

The PFM-80 monitor is the basic control program for the BIGBOARD
single-board computer. It begins execution when the computer is first
turned on, or whenever the reset button is pressed, and resides in the top
4K of memory.

The monitor provides two essential functions for the system. It
is the "software front panel" of the computer and it contains the routines
that initialize and control all the basic system input/output resources.
The "front panel" functions of the monitor include commands to display and
alter the contents of memory and I/O ports, to begin execution at a given
address, and to bootstrap programs from disk. The basic I/O functions of
PFM-80 provide driving routines for the built-in CRT display and keyboard
input, the external RS-232 serial terminal, the floppy disk controller,
and the real-time clock. In this capacity the monitor is always active,
even when application programs like the CP/M * disk operating system have
control of the CPU.

The following sections of this manual will explain how to use the
console monitor commands, what facilities are provided by the resident I/O
handlers, and how to interface applications programs to the monitor.

CONSOLE MONITOR COMMAND SUMMARY

The PFM-80 monitor enters the command mode after it has
initialized the system following a power-on or pushbutton reset. The
following sign-on message is displayed on the console output device as an
indication that the monitor is ready to accept commands.

.. system reset ..

* *

Commands consist of a single character command name and zero to
three hexadecimal numeric parameters separated by commas or spaces. The
command line may be entered using upper case or lower case letters. A
carriage return is used as the terminator. Errors within a line can be
corrected by typing control-H to delete the last character or control-X to
delete the entire line. If a line is entered with an unknown command
name, an invalid number or parameters or an out-of-range parameter, an
error message will be displayed and the command will not be executed.

The user may wish to halt long running commands like the memory
dump before they are finished. This can be done by typing carriage return
while the command is doing output. Output can also be frozen temporarially
and then re-started by typing repeatedly on the space bar.

The following table summarizes the monitor's command set. The
items enclosed in angle brackets represent the numeric parameters expected

c 1980 Russel Smith
by the command. A detailed description of each command is provided in the following pages.

<table>
<thead>
<tr>
<th>command</th>
<th>format</th>
</tr>
</thead>
<tbody>
<tr>
<td>d(ump)</td>
<td>D &lt;start&gt;,&lt;end&gt;</td>
</tr>
<tr>
<td>m(emory)</td>
<td>M &lt;address&gt;</td>
</tr>
<tr>
<td>t(est)</td>
<td>T &lt;start&gt;,&lt;end&gt;</td>
</tr>
<tr>
<td>f(ill)</td>
<td>F &lt;start&gt;,&lt;end&gt;,&lt;constant&gt;</td>
</tr>
<tr>
<td>c(opy)</td>
<td>C &lt;source_start&gt;,&lt;source_end&gt;,&lt;dest_start&gt;</td>
</tr>
<tr>
<td>v(erify)</td>
<td>V &lt;source_start&gt;,&lt;source_end&gt;,&lt;dest_start&gt;</td>
</tr>
<tr>
<td>g(oto)</td>
<td>G &lt;address&gt;</td>
</tr>
<tr>
<td>r(ead)</td>
<td>R &lt;unit&gt;,&lt;track&gt;,&lt;sector&gt;</td>
</tr>
<tr>
<td>b(oot)</td>
<td>B</td>
</tr>
<tr>
<td>i(nput)</td>
<td>I &lt;port&gt;</td>
</tr>
<tr>
<td>o(utput)</td>
<td>O &lt;port&gt;,&lt;data&gt;</td>
</tr>
</tbody>
</table>

1) DUMP COMMAND

The dump command outputs a tabular display of the contents of memory in hexadecimal and ASCII representation. Each display line has the following format:

```
AAAA DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD CCCCCCCCCCCCCCCCC
```

where AAAA is the starting memory address of the line in hexadecimal, the DD’s are the hex values of the 16 bytes of data starting at location AAAA, and the C’s are the ASCII characters equivalent to each data byte. Bytes less than 20 hex are replaced in the ASCII portion of the dump by periods.

The dump command accepts zero, one or two address parameters. If two addresses are specified, the block of memory between those two locations will be displayed. Entering only one address will display 256 bytes of memory starting at the specified location. Typing ‘D’ with no parameters will cause the routine to display the 256 byte block of memory starting at the last address displayed by the dump command.

2) MEMORY COMMAND

The memory examine/change command allows the contents of individual memory locations to be read from and written into using the monitor. This command accepts one parameter representing the memory address at which to begin examining data. The display format is as follows:

```
AAAA DD _
```

where AAAA is the current memory address and DD is the hexadecimal value of the data in that location. After displaying the contents of a memory location, the routine waits for one of the following items to be input from the console.

c 1980 Russel Smith
- Typing a carriage return will cause the routine to display the data at the next memory location, with no modification of content.

- Typing a minus sign will have a similar effect, except the address is decremented instead of incremented.

- Typing a two digit hexadecimal number will cause that number to be stored at the displayed address. The new data is stored as soon as the second digit is entered, with no terminating character required.

- Typing any character other than carriage return, a minus sign or a hexadecimal digit will cause the command to terminate.

3) TEST COMMAND

This command allows the user to test memory for errors caused by defective 16K memory chips, solder bridges and various other problems. Any portion of memory may be tested except the area reserved for the monitor (F000 to FFFF hex). Two parameters are required from the user; the starting address and ending address of the memory block to be tested. Only the high order 8 bits of the addresses entered are actually used however, due to a characteristic of the test algorithm being employed. If no errors occur, the test routine will output a plus sign every time a test pass is done. A total of 256 plus signs must be output for all possible test patterns to have been tried. When errors are detected an error line will be output in the following format:

AAAA DD should=XX

where AAAA is the address of a location that fails to test, DD is the data read back from that location, and XX is the test pattern that was written there.

4) FILL COMMAND

The fill command allows blocks of memory to be filled with a fixed data constant. Three parameters are required in the command line; a starting memory address, an ending address and a fill constant. Each location in the specified block of memory has the constant written into it and then read back again to check for memory errors. An error line like the one described for the 'T' command is printed for any locations that fail to verify.

5) COPY COMMAND

The copy command allows blocks of data to be moved around in memory. Three parameters are required in the command line; a starting memory address, an ending address, and a destination address. The contents of the block of memory bounded by the first two addresses copied to the block starting at the third address. As with the fill command, a test is made to verify that each byte of the destination block, when read back, is the same as the corresponding byte in source block.

c 1980 Russel Smith
6) VERIFY COMMAND

The verify command allows the contents of two blocks of memory to be compared with each other byte by byte. This command has the same syntax as the copy command. Each byte in the source block is compared with the corresponding byte in the destination block. Any locations that are not the same will cause a memory error message line to be displayed. If both blocks are identical nothing is output and control simply returns to the monitor.

7) GOTO COMMAND

The goto command allows control of the CPU to be passed to another program by the monitor. This command requires a single parameter from the user representing the address at which to begin execution. The monitor actually passes control to the specified location by executing a CALL instruction. This makes it possible for the external routine to return to the monitor by doing a RET, assuming it does not re-load the stack pointer and loose the return address to the monitor.

8) READ COMMAND

The read command allows individual disk sectors to be read into memory and displayed on the console. Three parameters are required; a drive unit number (range 0 to 3), a track number (range 0 to 4C) and a sector number (range 1 to 1A). The command routine performs a drive select, track seek and sector read sequence using the supplied parameters. If no errors occur, the contents of the input buffer will be dumped out in the 'D' command format. In the event of a disk error, a diagnostic message will be printed in the following format:

disk error XX UAA TBB SCC

Where XX represents the 1771 disk controller error status code, AA is the unit number, BB is the track number, and CC is the sector number. The error code is composed of eight bits of status information as described in the tables below.

<table>
<thead>
<tr>
<th>bit</th>
<th>read/write</th>
<th>seek/restore/select</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>drive not ready</td>
<td>drive not ready</td>
</tr>
<tr>
<td>6</td>
<td>write protected</td>
<td>unused</td>
</tr>
<tr>
<td>5</td>
<td>write fault</td>
<td>unused</td>
</tr>
<tr>
<td>4</td>
<td>record not found</td>
<td>seek error</td>
</tr>
<tr>
<td>3</td>
<td>crc error</td>
<td>crc error</td>
</tr>
<tr>
<td>2</td>
<td>lost data</td>
<td>cannot restore</td>
</tr>
<tr>
<td>1</td>
<td>unused</td>
<td>unused</td>
</tr>
<tr>
<td>0</td>
<td>always=1</td>
<td>always=0</td>
</tr>
</tbody>
</table>

The least significant bit of the error code indicates which of the above sets of error conditions is applicable. If the LSB=1 the disk error was generated by a read or write operation, otherwise it was caused by a seek, restore, or select operation.

c 1980 Russel Smith
9) BOOT COMMAND

The boot command is used to load and begin execution of a one sector long bootstrap loader from the first sector on drive unit zero. The most common use of this command will be to boot up the CP/M * disk operating system, although it is not necessarily restricted to this purpose only.

The boot works by reading the contents of track 0, sector 1 into memory at location 80 hex and then jumping to that address to start execution of the code just read in. Normally the routine on sector 1 will be a small loader that in turn reads in a larger program such as the operating system. This two level bootstrap process makes the boot command more application independent. The only requirements are that the first sector of the boot diskette be reserved for a loader and that the bottom 256 bytes of memory are not written over by the program being loaded.

10) INPUT COMMAND

This command allows the contents of input ports to be read from using the monitor. It operates very much like the memory examine command, except that input ports are being examined instead of memory locations. A single parameter representing a port number is expected in the command line. The contents of adjacent ports can then be examined by typing carriage return or a minus sign as in the 'M' command. Typing any other key will cause the routine to terminate.

11) OUTPUT COMMAND

The output command is provided to allow output ports to be written to using the monitor. Two parameters are expected in the command line; a port number and a data byte to be output to that port. Both parameters should be between 0 and FF hex. After outputting the specified data to the port, this routine simply returns to the monitor instead of stepping to the next location like the input command. This makes it possible to use the output command to initialize Z-80 peripheral devices like the SIO, PIO and CTC.

c 1980 Russel Smith
MONITOR RESIDENT I/O DRIVER FUNCTIONS
-------------------------------------

This section describes the facilities available in the PFM-80 monitor for controlling the input/output resources of the BIGBOARD single-board computer.

1) INTERRUPT PROCESSING

The PFM-80 monitor takes advantage of the powerful interrupt handling capabilities of the Z-80 microprocessor. Interrupts are utilized in the I/O drivers for the console keyboard input, the real-time clock and the floppy disk controller. All necessary initialization tasks and interrupt service routines for these devices are contained in the monitor.

For the most part, the operation of the interrupt mechanism should be transparent to most applications programs that will run under PFM-80. A few precautions must be taken however, to insure that user written software does not adversely effect the operation of the system. The following list describes the major hazards to the interrupt system;

- Interrupts should not be disabled permanently by user code, as this will lock-up the console input and real-time-clock routines.

- The Z-80 'I' register should never be altered. Doing so is GUARANTEED to crash the system.

- The CPU operates in Z-80 interrupt mode 2 and should not be switched to either of the other two interrupt modes.

- Adequate stack space must be reserved in user programs to allow at least one level of stack for interrupt return addresses. Use of the stack pointer for 'trick' programming purposes is highly discouraged for the same reason.

The monitor initializes the Z-80 'I' register to point to the system interrupt vector table at location FF00 to FF1F hex. This table contains pre-assigned vector locations for all the peripheral devices on BIGBOARD, including those that are not used by any built-in functions in PFM-80. The devices that are not currently used include SIO channel A, the general purpose PIO and CTC channels 0 and 1. These ports can be initialized and used as needed without affecting the overall system operation. Consult the monitor variables table at the end of this manual for the vector addresses.

2) MEMORY MAPPED VIDEO DISPLAY

The BIGBOARD single-board computer is equipped with a built-in 80 character by 24 line CRT display controller, for use with an external video monitor as the system console output device. The refresh memory for the CRT is bank switch-able from the system's 64K byte memory space and includes a hardware address translation circuit for high speed scrolling.
The PFM-80 monitor contains an output driver routine for the CRT that emulates the characteristics of a typical stand-alone video terminal. The control character set recognized by this routine is upwardly compatible with that of the well known Lear Siegler ADM3-A. An operational summary of the CRT driver is given below.

- All character codes between 20 and 7F hex are directly displayable on the screen. Characters are formed in a 5*7 dot matrix.

- New characters are stored on the screen at the location occupied by the cursor. The cursor is then moved one place to the right.

- If the cursor must appear at a screen position occupied by a non-blank character, the presence of the cursor will be indicated by making the overlayed character blink on and off.

- If a displayable character is output when the cursor is in the right-most column of the screen, an automatic carriage return and linefeed is generated afterwards.

- If a linefeed is output when the cursor is on the bottom line of the screen, the entire display is scrolled up one line and a new blank line is created on the bottom.

- All character codes between 00 and 1F hex are interpreted as control characters. Only 14 of these codes have some effect on the CRT display, and are described in the table below. The remaining 18 are treated as nulls.

<table>
<thead>
<tr>
<th>CODE</th>
<th>KEY</th>
<th>NAME</th>
<th>EFFECT</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>^@</td>
<td>NUL</td>
<td>NONE</td>
</tr>
<tr>
<td>01</td>
<td>^A</td>
<td>SOH</td>
<td>NONE</td>
</tr>
<tr>
<td>02</td>
<td>^B</td>
<td>STX</td>
<td>NONE</td>
</tr>
<tr>
<td>03</td>
<td>^C</td>
<td>ETX</td>
<td>NONE</td>
</tr>
<tr>
<td>04</td>
<td>^D</td>
<td>EOT</td>
<td>NONE</td>
</tr>
<tr>
<td>05</td>
<td>^E</td>
<td>ENQ</td>
<td>NONE</td>
</tr>
<tr>
<td>06</td>
<td>^F</td>
<td>ACK</td>
<td>NONE</td>
</tr>
<tr>
<td>07</td>
<td>^G</td>
<td>BEL</td>
<td>AUDIBLE BELL</td>
</tr>
<tr>
<td>08</td>
<td>^H</td>
<td>BS</td>
<td>BACKSPACE OR CURSOR LEFT</td>
</tr>
<tr>
<td>09</td>
<td>^I</td>
<td>HT</td>
<td>HORIZONTAL TAB</td>
</tr>
<tr>
<td>0A</td>
<td>^J</td>
<td>LF</td>
<td>LINEFEED OR CURSOR DOWN</td>
</tr>
<tr>
<td>0B</td>
<td>^K</td>
<td>VT</td>
<td>CURSOR UP</td>
</tr>
<tr>
<td>0C</td>
<td>^L</td>
<td>FF</td>
<td>CURSOR RIGHT</td>
</tr>
<tr>
<td>0D</td>
<td>^M</td>
<td>CR</td>
<td>CARRIAGE RETURN</td>
</tr>
<tr>
<td>0E</td>
<td>^N</td>
<td>SO</td>
<td>NONE</td>
</tr>
<tr>
<td>0F</td>
<td>^O</td>
<td>SI</td>
<td>NONE</td>
</tr>
<tr>
<td>10</td>
<td>^P</td>
<td>DLE</td>
<td>NONE</td>
</tr>
<tr>
<td>11</td>
<td>^Q</td>
<td>DC1</td>
<td>NONE</td>
</tr>
<tr>
<td>12</td>
<td>^R</td>
<td>DC2</td>
<td>NONE</td>
</tr>
<tr>
<td>13</td>
<td>^S</td>
<td>DC3</td>
<td>NONE</td>
</tr>
<tr>
<td>14</td>
<td>^T</td>
<td>DC4</td>
<td>NONE</td>
</tr>
<tr>
<td>15</td>
<td>^U</td>
<td>NAK</td>
<td>NONE</td>
</tr>
<tr>
<td>16</td>
<td>^V</td>
<td>SYN</td>
<td>NONE</td>
</tr>
</tbody>
</table>

© 1980 Russel Smith
Bell:

Generates a 10 microsecond pulse which can be used with external hardware to produce an audible 'BELL' sound.

Backspace:

Moves the cursor to the next column to the left. If the cursor is in the leftmost column of the screen, this character has no effect.

Tab:

Moves the cursor right to the next tab stop. The tab stops are fixed at every eighth column, starting from the left.

Linefeed:

Moves the cursor down one line on the screen. If the cursor is on the bottom-most line, the screen is scrolled up and a blank line is created on the bottom. The top line is lost.

Cursor Up:

Moves the cursor up one line on the screen. If the cursor is on the top of the screen it rolls around to the bottom.

Cursor Right:

Moves the cursor to the next column to the right. If the cursor is in the rightmost column, there is no effect.

Carriage Return:

Moves the cursor to the left-most column of the screen.

Clear To Eos:

Clears the contents of the screen from the current cursor position to the end of the bottom line.
Clear To Eol:

Clears the contents of the line the cursor is on, from the cursor position to the end of the line.

Clear Screen:

Clears the entire screen regardless of the current cursor position and places the cursor in the top-left corner of the screen. Also re-initializes the CRT driver subroutine.

Escape:

Used to initiate an XY cursor positioning sequence. The cursor can be moved to an arbitrary location on the screen by outputting a 4 character sequence composed of 1) escape, 2) equals sign, 3) row# character and 4) column# character. The row/column number characters are formed by taking the desired row# (range=0 to 23) or column# (range=0 to 79) and adding 20 hex to it.

Home Cursor:

Moves the cursor to the top-left corner of the screen, without altering any characters on the display.

Display CC's:

Functions as a prefix character to force the output of symbols in the character generator that correspond to ASCII control characters. The next character output to the CRT after outputting a 1F hex will be displayed on the screen regardless of its numeric value.

3) PARALLEL KEYBOARD INPUT

A parallel keyboard interface is provided on BIGBOARD for systems that will use the built-in keyboard and CRT display as the console I/O device. This interface is designed to connect to an ASCII encoded keyboard with 7 bits of parallel data and a key-pressed strobe.

The PFM-80 monitor contains an interrupt driven input handler for the keyboard that maintains a 16 character deep FIFO buffer for input data. This makes it possible to do a considerable amount of typing ahead without any input characters being lost. If characters are typed while disk access is going on, they may be lost because the disk routines lock out all lower priority interrupts. Any characters received when the FIFO is full will also be lost, in which case the interrupt handler will output a bell (07 hex) to the console output device as a warning.

The keyboard input handler contains a software shift lock mechanism as a convenience for users whose keyboards may not have a hardware shift lock. The keyboard input routine utilizes a user defined character (set to 00 hex on reset) as the shift lock. When this character is input from the keyboard it is not stored in the FIFO, but is used to
cause a software flag to be toggled on and off. Any time a character is received when this flag is turned on, the character will be shifted to the opposite case. Characters with values lower than 40 hex are not affected by the shift lock mechanism.

4) FLOPPY DISK CONTROLLER

The system is equipped with an on-board floppy disk interface capable of controlling up to four Shugart compatible 8 inch drives. The interface hardware is based on a Western Digital 1771 disk controller chip, along with extra TTL support circuitry to provide buffering, drive select, head load timing and data separator functions. An optional drive motor on/off control is also available.

The PFM-80 monitor contains a complete I/O driver package for the disk controller. Linkage to the disk I/O routines in the monitor is provided by a set of subroutine entry points described later in this manual. The basic functions available are; drive select, restore, seek track, read sector, and write sector. The user can also specify the track-to-track seek stepping rate, and the sector record length.

All disk functions are verified upon completion, with the final status being returned in the A register. If the command was executed successfully, then A will contain all zeros on return, otherwise it will contain an error status byte as described above under the console monitor 'R' command. The disk drive routines will attempt to recover from any disk I/O errors that occur, so it is generally not necessary for user written programs to try to re-execute commands that fail the first time.

5) SERIAL INPUT/OUTPUT OPTION

The BIGBOARD single-board computer has provisions for an optional serial I/O capability if the Z-80 SIO and its related support chips are installed. These components provide two completely independent RS-232 serial ports that can be used to interface to printers, CRT terminals and data communications equipment. Although the SIO is optional, it can be used with an external terminal to replace the built-in keyboard and CRT display as the system console I/O device.

On power-on reset the PFM-80 monitor determines which type of console I/O has been selected and configures the system accordingly. If serial I/O is selected, the monitor initializes SIO port B for asynchronous operation and waits for a carriage return to be typed in order to measure the baud rate of the terminal being used. Once the baud rate is set, the system behaves just as it would with the built-in console. Received characters generate interrupts and are buffered in a FIFO as described for the parallel keyboard.

If the built-in console I/O is being used, the SIO is not required to be installed in the system, but it is still initialized by the monitor for optional use as a printer port. In this case the SIO is also

C 1980 Russel Smith
programmed for asynchronous operation, but the baud rate is set to 300 and the interrupts are not enabled. A set of entry points for doing simple polled I/O with the SIO is provided in the monitor for use in this configuration. In both of the configurations mentioned above, the SIO is programmed to transmit and receive 7 bit words, with odd parity, and 1 stop bit.

6) REAL TIME CLOCK OPTION

The BIGBOARD single board computer has provisions an optional Z-80 CTC device that can be used to generate the timebase for interrupt driven timers, real-time clocks, and other time keeping functions. If a CTC is installed on the board, the monitor will initialize CTC channels 2 and 3 to interrupt the processor once a second. Channels 0 and 1 of the CTC are not initialized and can be used for other purposes.

The one second interrupt from the CTC is utilized by the the monitor's disk I/O routines to implement the optional disk motor turn-off function. If 30 seconds elapse with no disk I/O activity, the monitor will turn on an output bit that is connected to the pin marked 'DAC' on BIGBOARD's power supply connector. This can be wired to an optically isolated AC relay that controls the power to the disk drives. Power will be turned back on automatically when the next disk command is issued.

7) PARALLEL I/O OPTION

An optional Z-80 PIO chip has been included on BIGBOARD for general purpose I/O interfacing. This device is not required to be installed in the system and is completely unused by any built-in functions. The PIO contains two independent 8 bit parallel I/O ports that can be used to interface to printers, ROM programmers, analog converters, other computers, or just about anything else imaginable. Those interested in using the PIO should consult the schematic drawings for any needed hardware interfacing details. Data about programming the PIO can be found in most Z-80 applications manuals.

c 1980 Russel Smith
USER ACCESSIBLE MONITOR Routines AND VARIABLES

This section gives the locations and calling sequences of the user accessible I/O routines in the PFM-80 monitor. It also describes a number of important monitor variables that may need to be accessed by user written programs.

PFM-80 subroutines are accessed via a table of JUMP instructions beginning at memory location F000 hex. All monitor calls should be made to these entry points, since the actual addresses of the routines inside PFM-80 will vary between different releases. Parameter passing conventions for the monitor fall into one of two groups. The character oriented I/O routines all pass data using the A register, while the disk routines pass parameters in C and HL and return status information in A.

Storage for the monitor's stack and working variables occupies the top 256 bytes of memory, from FF00 to FFFF hex. The mode 2 interrupt vector table takes up the first 32 bytes of this block and the stack starts at the very top. In between, are variables used by the monitor resident I/O drivers and interrupt service routines, some of which are described below. Programs should not attempt to write into any locations in this block that are not specifically mentioned below.

1) PFM-80 SUBROUTINE ENTRY POINTS

<table>
<thead>
<tr>
<th>Location</th>
<th>Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>F000</td>
<td>ENT0:</td>
<td>JP INIT; PFM-80 cold start entry</td>
</tr>
<tr>
<td>F003</td>
<td>ENT1:</td>
<td>JP PROMPT; PFM-80 warm start entry</td>
</tr>
<tr>
<td>F006</td>
<td>ENT2:</td>
<td>JP CONST; console input status test</td>
</tr>
<tr>
<td>F009</td>
<td>ENT3:</td>
<td>JP CONIN; console input</td>
</tr>
<tr>
<td>F00C</td>
<td>ENT4:</td>
<td>JP CONOUT; console output</td>
</tr>
<tr>
<td>F00F</td>
<td>ENT5:</td>
<td>JP CRTOUT; memory-mapped CRT output</td>
</tr>
<tr>
<td>F012</td>
<td>ENT6:</td>
<td>JP SIOST; SIO port B input status test</td>
</tr>
<tr>
<td>F015</td>
<td>ENT7:</td>
<td>JP SIOIN; SIO port B input</td>
</tr>
<tr>
<td>F018</td>
<td>ENT8:</td>
<td>JP SIOOUT; SIO port B output</td>
</tr>
<tr>
<td>F01B</td>
<td>ENT9:</td>
<td>JP SELECT; disk drive select</td>
</tr>
<tr>
<td>F01E</td>
<td>ENT10:</td>
<td>JP HOME; restore to track 0</td>
</tr>
<tr>
<td>F021</td>
<td>ENT11:</td>
<td>JP SEEK; seek track</td>
</tr>
<tr>
<td>F024</td>
<td>ENT12:</td>
<td>JP READ; read sector into memory</td>
</tr>
<tr>
<td>F027</td>
<td>ENT13:</td>
<td>JP WRITE; write sector from memory</td>
</tr>
</tbody>
</table>

© 1980 Russel Smith
<table>
<thead>
<tr>
<th>FUNCTION</th>
<th>PARAMETERS</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>COLD</td>
<td>IN: none</td>
<td>Perform cold start initialization of PFM-80 monitor and enter command mode. OUT: does not return</td>
</tr>
<tr>
<td>WARM</td>
<td>IN: none</td>
<td>Enter PFM-80 monitor command mode with no re-initialization. OUT: does not return</td>
</tr>
<tr>
<td>CONST</td>
<td>IN: none</td>
<td>Test for data ready in console input FIFO and return status in A. If data is available then A=FF hex. OUT: status in a</td>
</tr>
<tr>
<td>CONIN</td>
<td>IN: none</td>
<td>Return character from console input FIFO in A. If FIFO is empty then loop until character is input. OUT: character in A</td>
</tr>
<tr>
<td>CONOUT</td>
<td>IN: character in A</td>
<td>Output character passed in A to the console output device. OUT: none</td>
</tr>
<tr>
<td>CRTOUT</td>
<td>IN: character in A</td>
<td>Output character passed in A to the memory-mapped CRT display. OUT: none</td>
</tr>
<tr>
<td>SIOST</td>
<td>IN: none</td>
<td>Test for received data available from SIO channel B and return status in A. if data is available then A=0, else A=FF hex. OUT: status in A</td>
</tr>
<tr>
<td>SIOIN</td>
<td>IN: none</td>
<td>Return received data from SIO channel B in A. Loop until data is received if none is available on entry. OUT: character in A</td>
</tr>
<tr>
<td>SIOOUT</td>
<td>IN: character in A</td>
<td>Output character passed in A to SIO channel B transmit register. OUT: none</td>
</tr>
<tr>
<td>SELECT</td>
<td>IN: unit number in C</td>
<td>Select specified drive for future restore, seek, read or write command. If the drive is not ready, then the currently selected drive is left on. OUT: status in A</td>
</tr>
<tr>
<td>HOME</td>
<td>IN: none</td>
<td>Move read/write head to home position at track 0 and verify if it got there. OUT: status in A</td>
</tr>
<tr>
<td>SEEK</td>
<td>IN: track number in C</td>
<td>Move read/write head to specified track and verify if it got there. OUT: status in A</td>
</tr>
<tr>
<td>READ</td>
<td>IN: sector number in C</td>
<td>Read specified sector on current buffer pointer in HL track into memory data buffer. OUT: status in A</td>
</tr>
<tr>
<td>WRITE</td>
<td>IN: sector number in C</td>
<td>Write specified sector on current buffer pointer in HL track from memory data buffer. OUT: status in A</td>
</tr>
</tbody>
</table>
2) STORAGE ALLOCATION FOR PFM-80 VARIABLES

{ INTERRUPT VECTOR TABLE }

| FF00 | SIOV0:   | DEFS 2 | ;SIO port B xmit buffer empty |
| FF02 | SIOV1:   | DEFS 2 | ;SIO port B external/status change |
| FF04 | SIOV2:   | DEFS 2 | ;SIO port B recieve data available |
| FF06 | SIOV3:   | DEFS 2 | ;SIO port B special recieve condition |
| FF08 | SIOV4:   | DEFS 2 | ;SIO port A xmit buffer empty |
| FF0A | SIOV5:   | DEFS 2 | ;SIO port A external/status change |
| FF0C | SIOV6:   | DEFS 2 | ;SIO port A recieve data available |
| FF0E | SIOV7:   | DEFS 2 | ;SIO port A special recieve condition |

| FF10 | CTCV0:   | DEFS 2 | ;CTC channel 0 interrupt |
| FF12 | CTCV1:   | DEFS 2 | ;CTC channel 1 interrupt |
| FF14 | CTCV2:   | DEFS 2 | ;CTC channel 2 interrupt |
| FF16 | CTCV3:   | DEFS 2 | ;CTC channel 3 interrupt |

| FF18 | SYSVA:   | DEFS 2 | ;system PIO port A interrupt |
| FF1A | SYSVB:   | DEFS 2 | ;system PIO port B interrupt |

| FF1C | GENVA:   | DEFS 2 | ;general purpose PIO port A interrupt |
| FF1E | GENVB:   | DEFS 2 | ;general purpose PIO port B interrupt |

{ CONSOLE KEYBOARD INPUT VARIABLES }

| FF20 | FIFO:    | DEFS 16 | ;input data fifo buffer |
| FF30 | FIFCNT:  | DEFS 1  | ;number of characters in FIFO |
| FF33 | LOCK:    | DEFS 1  | ;character used for software shift loc |

{ REAL-TIME-CLOCK VARIABLES }

| FF5D | TIKCNT:  | DEFS 2  | ;binary one second interrupt counter |
| FF5F | DAY:     | DEFS 1  | ;calendar day |
| FF60 | MONTH:   | DEFS 1  | ;calendar month |
| FF61 | YEAR:    | DEFS 1  | ;calendar year |
| FF61 | HRS:     | DEFS 1  | ;clock hours |
| FF63 | MINS:    | DEFS 1  | ;clock minutes |
| FF64 | SECS:    | DEFS 1  | ;clock seconds |

{ DISK INPUT/OUTPUT DRIVER VARIABLES }

| FF6A | SPEED:   | DEFS 1  | ;track to track stepping rate |
| FF6B | RECLLEN: | DEFS 1  | ;disk record length for read/write |
| FF6C | MOTOR:   | DEFS 1  | ;drive motor activity timer |

{ CRT OUTPUT DRIVER VARIABLES }

| FF76 | CSRCHR:  | DEFS 1  | ;character used for cursor indicator |
| FF77 | BASE:    | DEFS 1  | ;current content of scroll register |
| FF79 | NULLS:   | DEFS 1  | ;null count for SIO control character |
|      |          |         | ;output padding |

© 1980 Russel Smith
PFM Monitor Listing

0001: ******************************************************
0002: *
0003: * BIGBOARD MONITOR ROM, NON-RELOCATABLE VERSION *
0004: * Russell Smith 2-August-1980 *
0005: *
0006: ******************************************************
0007: *
0008: *
0009: PSECT  ABS
0010: F000   ROM  EQU  0F000H ;START OF 2K ROM
0011: FF00   RAM  EQU  0FF00H ;START OF 256 BYTE RAM
0012: 3000   CRTMEM EQU  3000H  ;BASE OF 4K CRT MEMORY
0013: *
0014: *
0015: ORG  ROM
0016: INCLUDE INIT.ASM
0017: ******************************************************
0018: *
0019: ; COLD START INITIALIZATION ROUTINE FOR
0020: ; CONFIGURING THE SYSTEM AFTER A POWER-ON
0021: ; OR PUSHBUTTON RESET.
0022: 18-Oct-80
0023: *
0024: ******************************************************
0025: *
0026: *
0027: ; --- MONITOR ENTRY POINT TABLE ---
0028: *
0029: F000 C32AF0 0029 COLD:  JP  INIT  ;MONITOR COLD ENTRY POINT
0030: F003 C32BF1 0030 WARM:  JP  PROMPT ;MONITOR WARM ENTRY POINT
0031: F006 C331F4 0031 CONST: JP  KBDST ;CONSOLE STATUS VECTOR
0032: F009 C339F4 0032 CONIN: JP  KBGIN ;CONSOLE INPUT VECTOR
0033: F00C C32OF5 0033 CONOUT: JP  CRTOUT ;CONSOLE OUTPUT VECTOR
0034: F00F C320F5 0034 JP  CRTOUT ;CRT OUTPUT VECTOR
0035: F012 C3E8F4 0035 JP  SIDST ;SIO CHANEL B STATUS VECTOR
0036: F015 C3F0F4 0036 JP  SIOIN ;SIO CHANEL B INPUT VECTOR
0037: F018 C3F3F4 0037 JP  SIOOUT ;SIO CHANEL B OUTPUT VECTOR
0038: F01B C3B1F6 0038 JP  SELECT ;DISK DRIVE SELECT
0039: F01E C3E9F6 0039 JP  HOME  ;HOME R/W HEAD
0040: F021 C3F8F6 0040 JP  SEEK ;SEEK TO TRACK
0041: F024 C32AF7 0041 JP  READ  ;READ SECTOR
0042: F027 C31FF7 0042 JP  WRITE  ;WRITE SECTOR
0043: ;
0044: ; DO A SHORT POST-RESET TIME DELAY. ALSO INITIALIZES THE
0045: ; STACK POINTER AND FILLS THE MONITOR SCRATCH RAM WITH ZEROS
0046: ;
0047: F02A F3 0049 INIT:  DI
0048: F02B 2100FF 0050 LD  HL,RAM ;POINT TO START OF MONITOR RAM
0049: F02E 3600 0051 INIT1: LD  (HL),O ;FILL 256 BYTE SPACE WITH ZEROS
0050: F030 F9 0052 LD  SF,HL ;SOMETHING USEFUL TO ADD DELAY
0051: F031 2C 0053 INC  L
0052: F032 20FA 0054 JR  N2,INIT1-# ;LOOP TAKES 4 MILLISECONDS
0053: 0056: INITIALIZE THE Z-80 FOR INTERRUPT MODE #2
0057: 0058: 0059:
F035 ED47 0059  LD  I,A  ;LOAD I REG W/ MSB OF VECTOR TABLE
F037 ED5E 0060  IM  2  ;AND SELECT INTERRUPT MODE 2
F039 CDECF5 0061  CALL  CLRSSCN  ;FILL THE CRT MEMORY WTH BLANKS
F041 21D3F0 0062  ;STORE ANY NON-ZERO VALUES FOR VARIABLES IN MEMORY
F043 0600 0064  INIT2: LD  HL,INTAB  ;POINT TO DEFAULT VAR TABLE
F045 4E 0066  LD  C,(HL)  ;BC=DATA BLOCK BYTECOUNT
F047 23 0068  INC  HL  ;DE=DESTINATION FOR DATA
F049 5E 0069  LD  E,(HL)
F04B 56 0070  INC  HL
F04D 28F2 0071  ;COPY DATA @ HL TO VAR @ DE
F04F B7E 0072  BIT  7,(HL)  ;LOOP AGAIN IF NOT END OF TABLE
F051 0073  JR  2,INIT2-$  ;INITIALIZE THE PROGRAMMABLE I/O DEVICES
F053 23 0074  INC  HL  ;POINT TO I/O INIT DATA TABLE
F055 4E 0075  LDIR  7,(HL)  ;B=INIT LOOP BYTECOUNT
F057 23 0076  INC  HL  ;C=DEVICE CONTROL PORT#
F059 B7E 0077  BIT  7,(HL)  ;SEND DATA @ HL TO PORT @ C
F05B 28F6 0078  JR  2,INIT3-$  ;TEST FOR TABLE END MARKER
F05D 0079  ;LOOP AGAIN IF NOT AT END
F05F E78 007A  IN  A,(C)  ;Determine if console I/O configuration will be for the
F061 EE6 007B  CP  000000110B  ;externaL I/O device, or an external serial terminal.
F063 C12 007C  JR  NZ,PARALL-$  ;IF THE SIO IS INSTALLED
F065 DB1E 007D  IN  A,(KBDAT)  ;SKIP CONFIG TEST IF NO SIO
F067 0610 007E  LD  B,00000000B  ;MAKE SURE KBD PIO 'READY' RESET
F069 0610 007F  ;RESET SIO INIT STATUS COMMAND
F06B 27E 0080  DECIDE: OUT  (C),B  ;TEST FOR ARRIVAL OF A SERIAL
F06D 0081  IN  A,(C)  ;INPUT CHAR START BIT
F071 CB67 0082  BIT  4,A  ;INPUT CHAR START BIT
F073 2000 0083  JR  NZ,BAUD-#  ;EXIT LOOP IF START BIT DETECTED
F075 DB1C 0084  IN  A,(BIDAT)
F077 CB5F 0085  BIT  3,A  ;TEST FOR DATA RDY STROBE FROM
F079 20F2 0086  JR  NZ,DECIDE-$  ;PARALLEL KBD, LOOP IF INACTIVE
F081 DB1E 0087  PARALL: IN  A,(KBDAT)  ;DISCARD FIRST KEYBOARD CHAR
F083 08B3 0088  LD  A,10000011B  ;PARALLEL KEYBOARD CHAR
F085 D31F 0089  OUT  (KBDCTL),A  ;ENABLE INTERRUPTS FROM KBD PIO
F087 182D 008A  JR  SIGNON-$  ;SIGN ON W/ BUILT-IN CONSOLE I/O
F089 008B  ;AUTOMATIC BAUD RATE SETTING ROUTINE FOR SIO
F08B 008C  ;1111
F08D AF 008D  BAUD: XOR  A
F08F ED41 008E  BAUD1: OUT  (C),B
F091 ED50 008F  IN  D,(C)  ;READ SIO STATUS REGISTER
F093 CB62 0090  BIT  4,D  ;TEST THE SYNC/HUNT BIT
F095 28F8 0091  JR  2,BAUD1-$  ;LOOP UNTIL IT CHANGES STATE
F097 3C 0092  Baud2: INC  A
F099 ED41 0093  OUT  (C),B  ;RESET REGISTER #0 FLAGS AGAIN
F09B ED50 0094  IN  D,(C)  ;& LOOP TIMING THE SYNC/HUNT BIT
F09D CB62 0095  BIT  4,D
F09F 20F7 0096  JR  NZ,BAUD2-$  ;REPEAT UNTIL BIT CHANGES AGAIN
F08A  21CAF0  012  LD  HL,RATES-1
F08B  23    015  Baud:  INC  HL  ;INDEX INTO BAUD RATE TABLE
F08C  17    0124  RLA  ;USING COUNT DERIVED IN A
F08D  30FC  0125  JR  NC.BAUDZ-$
F08E  7E    0126  LD  A,(HL)  ;GET BAUD RATE CONTROL BYTE FROM
F08F  D30C  0127  OUT  (BAUDB),A  ;TBL & OUTPUT TO COM-8116 TIMER
F090  CDF0F4  0128  CALL  SIOIN  ;DISCARD 1ST SERIAL INPUT CHAR
F091  3E01  0129  LD  A, 1
F092  D307  0130  OUT  (SI0CPB),A  ;RE-PROGRAM SIO B TO GENERATE
F093  3E1C  0131  LD  A,00011100B;INTERRUPTS ON RECEIVED DATA,
F094  D307  0132  OUT  (SI0CPB),A  ;PARITY DOES NOT AFFECT VECTOR
F095  21FEF4  0133  LD  HL,SIOOUT
F096  220DF0  0134  LD  (CONDOUT+1),HL;RE-DIRECT CONSOLE OUTPUT
             TO SIO
F097  0135    0136  ;PRINT SIGNON MESSAGE
F098  ;0137    ;
F099  0FB    0138  SIGNON; EI
F09A  06CF3  0139  CALL  PNext
F09B  000A  0140  DEFB  CR,LF
F09C  2E2E2E20  0141  DEFM  '"... system monitor 3.3 ..."'
F09D  73797374  0142
F09E  656D2060  0143
F09F  6F6E6974  0144
F0A0  6F722033  0145
F0A1  2E33202E  0146
F0A2  2E2E    0147
F0A3  000A  0148  DEFB  CR,LF
F0A4  04    0149  DEFB  EOT
F0A5  C303F0  0150  JP  WARM  ;GO ENTER MONITOR
F0A6  0151    0145 ;
F0A7  0152    0146 ;
F0A8  0153    0147 ;
F0A9  0154    0148 ;BAUD RATE CONSTANTS FOR COM 8116 BAUD RATE GENERATOR
F0AA  0155    0149 ;
F0AB  0156    0150 ;
F0AC  0157    0151 ;
F0AD  0152    0152 ;
F0AE  0153    0153 ;
F0AF  0154    0154 ;
F0B0  0155    0155 ;
F0B1  0156    0156 ;
F0B2  0157    0157 ;
F0B3  0158    0158 ;
F0B4  0159    0159 ;
F0B5  0160    0160 ;INTAB EGU $  ;INITIALIZATION DATA TABLES
F0B6  0161    0161 ;
F0B7  0162    0162 ;
F0B8  0163    0163 ;INITIALIZE THE Z-80 'I' REGISTER INTERRUPT VECTOR TABLE
F0B9  0164    0164 ;
F0BA  02    0165 ;F0D3  DEFB  2
F0BB  1AFF  0166 ;F0D4  DEFW  SYSVEC+2
F0BC  8CF4  0167 ;F0D5  DEFW  KEYSRV  ;PARALLEL KBD INTERRUPT VECTOR
F0BD  02    0168
F0BE  16FF  0169 ;F0D6  DEFB  2
F0BF  9FF4  0170 ;F0D7  DEFW  CTCVEC+6
F0C0  09    0171 ;F0D8  DEFW  TIMER  ;1 SEC TIMER INTERRUPT VECTOR
F0C1  04    0172
F0C2  0AFF  0173 ;F0D9  DEFB  4
F0C3  8CF4  0174 ;F0DB  DEFW  SIOVEC+4
F0C4  02    0175 ;F0DC  DEFW  SIOINT  ;SIO RECEIVE INTERRUPT VECTOR
F0C5  1AFF  0176 ;F0DD  DEFW  SIOERR  ;SIO PARITY, OVERRUN & FRAMING
F0C6  8CF4  0177 ;ERROR
F0C7  04    0178 ;F0DE  DEFB  8
F0C8  0AFF  0179 ;F0DF  DEFB  B
F0C9  ;F0E0  DEFW  SIOVECTOR
F0CA  8CF4  0180 ;F0E4  DEFW  SIOERROR
DEFW UNIT
DEFB 255 ;FLAG ALL DRIVES AS DE-SELECTED
DEFB 255,255,255,255 ;CLEAR HEAD POSITION TABLE
DEFB 00000000B ;SELECT FASTEST SEEK SPEED
DEFB 128 ;SELECT 128 BYTE SECTOR LENGTH
DEFB 30 ;SET MOTOR TURN-OFF TIMER

;INITIALIZE THE CRT DISPLAY CURSOR

DEFB 2
DEFW CHRSAV
DEFB 1
DEFB 1
;USE NON-BLINKING UNDERSCORE

;SET DEFAULT 'SOFTWARE' INTERRUPT VECTORS

DEFB 6
DEFW TIKVEC
DEFW DSKTMR ;POINT TIKVEC TO DISK MTR TIMER
DEFW STASH ;POINT PINVEC TO FIFO INPUT
DEFW STASH ;POINT SINVEC TO FIFO INPUT

;SET FREE MEMORY POINTER

DEFB 2
DEFW FREPTR ;POINT TO 1ST LOC AFTER MONITOR
DEFW ROMEND

;END OF VARIABLE INIT TABLE

BAUDA EQU 00H ;CHANNEL A BAUD RATE GENERATOR
SID EQU 04H ;DUAL SERIAL I/O
GENPIO EQU 08H ;GENERAL PURPOSE PARALLEL I/O
BAUDB EQU 0CH ;CHANNEL B BAUD RATE GENERATOR
WD1771 EQU 10H ;WEST DIGITAL DISK CONTROLLER
SCROLL EQU 14H ;CRT SCROLL MEM SCROLL REGISTER
CTC EQU 18H ;QUAD COUNTER/TIMER CIRCUIT
SYSPIO EQU 1CH ;SYSTEM PARALLEL I/O

INITIALIZE SYSTEM PIO FOR USE AS BANK-SWITCH,

BITDAT EQU SYSPIO+0
BITCTL EQU SYSPIO+1
KBDDAT EQU SYSPIO+2
KBDCALL EQU SYSPIO+3

FUT SYSTEM PIO IN BIT MODE
MAKE BITS 4 AND 3 BE INPUTS
DISABLE INTERRUPTS

DE-SELECT ROMS, ENABLE DRIVE O

PUT KBD PORT IN INPUT MODE
LOAD KEYBOARD INTERRUPT VECTOR
;INITIALIZE CHANNELS 2 AND 3 OF THE CTC
;TO GENERATE ONE SECOND INTERRUPTS FROM CTC3

>0018 CTC0 EQU CTC+0 ;CTC CHAN 0 PORT#
>0019 CTC1 EQU CTC+1 ;CTC CHAN 1
>001A CTC2 EQU CTC+2 ;CTC CHAN 2
>001B CTC3 EQU CTC+3 ;CTC CHAN 3

0246 ;

0247 F10F 011B DEFB 1,CTC0 ;BASE INTERRUPT VECTOR FOR CTC
0248 F111 10 DEFB CTCVEC

0249 ;

0250 F112 021A DEFB 2,CTC2 ;PUT CTC2 IN TIMER MODE
0251 F114 27 DEFB 00100111B ;CTC2 PERIOD=105*256*400 NS
0252 F115 69 DEFB 105
0253 F116 021B DEFB 2,CTC3 ;PUT CTC3 IN COUNTER MODE
0254 F118 C7 DEFB 11000111B ;CTC3 PERIOD=999936 uS
0255 F119 5D DEFB 93

0256 ;

0257 ;INITIALIZE SIO CHANEL B FOR ASYNCHRONOUS SERIAL
;INTERFACE TO PRINTER OR TERMINAL

0258 ;

>0004 SIODPA EQU SIO+0 ;SIO DATA PORT A
>0005 SIOPB EQU SIO+1 ;SIO DATA PORT B
>0006 SICPA EQU SIO+2 ;SIO CONTROL/STATUS PORT A
>0007 SICPB EQU SIO+3 ;SIO CONTROL/STATUS PORT B

0259 ;

0260 F11A 010C DEFB 1,BAUDB ;SET COM B116 TO 300 BD DEFAULT
0261 F11C 05 DEFB 0101B
0262 F11D 0B07 DEFB 11,SIOPB ;SELECT REGISTER #4
0263 F11E 0E DEFB 4
0264 F120 45 DEFB 0100101B ;16X CLK, 1 STOP BIT, ODD PARITY
0265 F121 01 DEFB 1 ;SELECT REGISTER #1
0266 F122 04 DEFB 0000100B ;STATUS AFFECTS VECTOR,

0267 ;NO INTERRUPTS

0268 F123 03 DEFB 3 ;SELECT REGISTER #3
0269 F124 41 DEFB 0100001B ;7 BITS/RX CHAR
0270 F125 05 DEFB 5 ;SELECT REGISTER #5
0271 F126 AA DEFB 1010101B ;7 BITS/TX CHAR, ASSERT DTR
0272 F127 02 DEFB 2 ;SELECT REGISTER #2
0273 F128 00 DEFB SIOPB ;LOAD INTERRUPT VECTOR BASE
0274 F129 02 DEFB 2 ;SELECT READ REG#2 FOR SIO TEST
0275 F12A FF DEFB -1 ;END-OF-TABLE

0276 ;

0277 ;INIT DONE
0278 ;

;INCLUDE MONITOR.ASM

;**************************************************************************
0279 ;**************************************************************************
0280
0281
0282
0283
0284
0285
0286
0287
0288
0289
0290
0291
0292
0293
0294
0295
0296
0297
0298
0299
0300
0301
0302
0303
F12B CDECF3 0304 PROMPT: CALL PNEXT
F12E 000A 0305 DEFB CR.LF
F130 2A20 0306 DEFM ' * '
F132 04 0307 DEFB EOT
F133 2188FF 0308 LD HL,LINBUF
F136 0820 0309 LD C,32
F138 CD38F3 0310 CALL GETLIN ; INPUT A BUFFERED CONSOLE LINE
F13B 3B35 0311 JR C,WHAT-$ ; PRINT 'WHAT?' IF INPUT ERROR
F13D AF 0313 XOR A
F13E 3284FF 0314 LD (ESCFLG),A
F141 CDFFFC3 0315 CALL CRLFS
F144 3A88FF 0316 LD A,(LINBUF) ; GET FIRST CHAR IN LINE
F147 FE0D 0317 CP CR
F149 28E0 0318 JR Z,FROMPT-$ ; JUMP IF A NULL LINE
F14B 2182F1 0319 LD HL,CMDTAB ; SEARCH FOR A MATCHING CHAR
F14E 010000 0320 LD BC,CMDSIZ/3 ; IN COMMAND SEARCH TABLE
F151 CD60F3 0321 CALL SEARCH
F154 201C 0322 JR NZ,WHAT-$ ; TRY AGAIN IF SEARCH FAILS
F156 C5 0323 PUSH BC
F157 FD2189FF 0324 LD IY,LINBUF+1
F15B CD6AF3 0325 CALL PARAMS ; INPUT NUMERIC PARAMETERS FROM
F15E DDE1 0326 POP IX ; LINE BUFFER AND TEST IF ERROR
F160 3B10 0327 JR C,WHAT-$
F162 2A7CFF 0328 LD HL,(PARAM1)
F165 ED5BEFF 0329 LD DE,(PARAM2)
F169 ED4BB0FF 0330 LD BC,(PARAM3)
F16D CD80F1 0331 CALL CALLX ; CALL SUBROUTINE & IX
F170 30B9 0332 JR NC,FROMPT-$ ; GO BACK TO PROMPT IF NO ERRORS
F172 CDECF3 0334 WHAT: CALL PNEXT
F175 20778861 0335 DEFM ' what ?'
F177 7A203F
F17C 07 0336 DEFB 'G'-64 ; SAY 'what?' AND BEEP THE BELL
F17D 04 0337 DEFB EOT
F17E 18AB 0338 JR PROMPT-$
F180 DDE9 0341 CALLX: JP (IX) ; CALL SUBROUTINE & IX
F180 0342 0343 0344 0345 0346 0347 0348 0349 034A 034B 034C 034D 034E 034F 0350 0351 0352 0353 0354 0355 0356 0357 0358 0359 035A 035B 035C 035D 035E 035F 0360 0361 0362 0363 0364 0365 0366 0367 0368 0369 036A 036B 036C 036D 036E 036F
F182 52 0345 CMDTAB: DEFB 'R'
F183 4F 0346 DEFB 'O'
F184 49 0347 DEFB 'I'
F185 47 0348 DEFB 'G'
F186 54 0349 DEFB 'T'
F187 4F 0350 DEFB 'F'
F188 4D 0351 DEFB 'M'
F189 43 0352 DEFB 'C'
F18A 42 0353 DEFB 'B'
F18B 44 0354 DEFB 'D'
F18C 53 0355 DEFB 'S'
F18D 29F3 0357 DEFW SWITCH ; SWITCH CONSOLE OUTPUT VECTOR
F18F 05F2 0358 DEFW MENDMP ; DUMP MEMORY IN HEX/ASCII
F191 A3F1 0359 DEFW BOOT ; BOOT UP CP/M
F193 E6F2 0360 DEFW BLOCK ; MEMORY BLOCK MOVE
F195 57F2 0361 DEFW VIEW ; MEMORY EXAMINE/CHANGE
F197 DBF2 0362 DEFW FILL ; FILL MEMORY
F199 BCF2 0363 DEFW TEST ; RAM DIAGNOSTIC
F19B B1F2 0364 DEFW GOTO ; JUMP TO MEMORY LOCATION
F19D FEF2 0365 DEFW INCMD ; READ FROM INPUT PORT
0021 CMDSIZ EQU $-CMDBR

; MONITOR COMMAND ACTION ROUTINES PACKAGE

; DISK BOOT LOADER COMMAND
LD C,0 ; SELECT DRIVE 0 FOR BOOT LOAD
CALL SELECT
JR NZ,DSKERR-$
CALL HOME ; HOME HEAD TO TRACK 0
JR NZ,DSKERR-$$ERROR IF NOT READY OR AT TR0
LD HL,0080H ; POINT TO CP/M READ BUFFER
LD C,1 ; SELECT SECTOR 1
CALL READ ; READ TRACK 0/ SECTOR 1
JR NZ,DSKERR-$$
PDP AF ; CLEAN UP STACK
JP 0080H ; GO EXECUTE LOADER

; DISK SECTOR READ COMMAND
DSKCMD: CP 3 ; CHECK PARAMETER COUNT
SCF
RET NZ
LD C,L ; USE FIRST ARG AS UNIT#
LD C,(HL) ; USE SECOND ARG AS TRACK#
CALL SEEK
JR NZ,DSKERR-$$
LD HL,PARAM3 ; USE THIRD ARG AS SECTOR#
LD C,(HL) ; MARK ERROR BYTE AS DUE TO READ
CALL READ
JR NZ,DSKERR-$$
LD HL,0080H
LD DE,B
CALL DUMP ; DUMP DISK READ BUFFER & RETURN
JP 0018 ;

; SAVE 1771 STATUS
CALL PNEST
DEFM 'disk error'

; PRINT 1771 ERROR BYTE IN BIN
LD B,B
LD A,'0'
XOR B
LD A,'0'
CALL DSKR2-$$ ; REPEAT FOR 8 BITS
0432 ;
0433 ;
0434 ;
0435 ;-- MEMORY DUMP COMMAND --
0436 ;
F205 3D 0437 MEMDMP: DEC A ;CHECK PARAMETER COUNT
F206 2806 0438 JR Z, MDMP2-$
F208 3D 0439 DEC A
F209 280B 0440 JR Z, MDMP3-$
F20B 2AB6FF 0441 MDMP1: LD HL, (LAST)
F20E 110000 0442 MDMP2: LD DE, 16
F211 180D 0443 JR MDMP3B-$
F213 EB 0444
F214 ED52 0445 MDMP3: EX DE, HL ;DERIVE BYTECNT FOR DUMP RANGE
F216 0604 0446 SBC HL, DE
F218 CB3C 0447 LD B, 4
F21A CB1D 0448 MDMP3A: SRL H ;DIVIDE BYTECOUNT BY 16
F21C 10FA 0449 RR L
F21E 23 0450 DJNZ MDMP3A-$
F21F EB 0451 INC HL
F220 CD2F2 0452 EX DE, HL
F223 22B6FF 0453 MDMP3B: CALL DUMP ;DUMP DE+$16 BYTES STARTING AT HL
F226 C9 0454 LD (LAST), HL
F227 ES 0455 RET
F228 CD3F3 0456 DUMP: PUSH HL ;SAVE STARTING ADDRESS
F228 CD3F3 0457 CALL PUT4HS ;PRINT STARTING ADDRESS IN HEX
F22B CD3F4 0458 CALL SPACE
F22E 0610 0459 DUMP2: LD B, 16
F230 7E 0460 DUMP2: LD A, (HL) ;GET A DATA BYTE @ HL
F231 23 0461 INC HL
F232 CD3F3 0462 CALL PUT2HS ;PRINT THE DATA IN HEX
F235 10F9 0463 DJNZ DUMP2-$
F237 E1 0464 POP HL ;REPEAT 16 TIMES
F23B 0610 0465 DUMP2A: LD B, 16
F23D 7E 0466 DUMP2A: LD A, (HL) ;GET BACK DATA BYTE @ HL
F23E 23 0467 INC HL
F240 CBBF 0468 RES 7, A
F243 FE20 0469 CP 20H
F240 3804 0470 JR C, DUMP4-$
F242 FE7F 0471 CP 7FH
F244 3B02 0472 JR C, DUMP5-$
F246 3EE2 0473 DUMP4: LD A, ',' ;PRINT DOT IF DATA < 20 OR > 7F
F248 CD15F4 0474 DUMP5: CALL OUTPUT ;PRINT ASCII CHARACTER IN A
F24B 10ED 0475 DUMP5: CALL DUMP3-$
F24D CDBF3 0476 CALL CRLF
F250 C0 0477 RET NZ ;EXIT IF ESCAPE REQ INDICATED
F251 1B 0478 DEC DE
F252 7A 0480 LD A, D
F253 B3 0481 OR E
F254 20D1 0482 JR NZ, DUMP-$
F256 C9 0483 RET
F257 CDCEF2 0484 ;-- MEMORY EXAMINE COMMAND --
F259 CD9F4 0485 VIEW: CALL MDATA
F25A CD07F4 0486 VIEW: CALL ECHO
F25D FE0D 0487 CP CR
F25F 2B1F 0488 JR Z, VIEW4-$
F261 FE2D 0489 CP ' -'
F263 2B19 048A JR Z, VIEW5-$
F265 CDBF3 048B VIEW2: CALL ASCHEX
F268 3F 0498 CCF
F269 D0 0499 RET NC
F26A 07 0500 RLCA
F26B 07 0501 RLCA
F26C 07 0502 RLCA
F26D 07 0503 RLCA
F26E 4F 0504 LD C,A
F26F CD07F4 0505 CALL ECHO
F272 CDBDF3 0506 CALL ASCHEX
F275 3F 0507 CCF
F276 D0 0508 RET NC
F277 B1 0509 OR C
F278 77 0510 VIEW3: LD (HL),A
F279 CDB9F2 0511 CALL CHECK
F27C 23 0512 VIEW4: INC HL
F27D 23 0513 INC HL
F27E 2B 0514 VIEW5: DEC HL
F27F 18D6 0515 JR VIEW5:
0516 ;
0517 ;
0518 ;
0519 ; -- JUMP TO MEMORY LOCATION COMMAND --
0520 ;
F281 3D 0521 GOTO: DEC A ;CHECK PARAMETER COUNT
F282 37 0522 SCF
F283 C0 0523 RET NZ
F284 E5 0524 PUSH HL
F285 DDE1 0525 POP IX
F287 CDB0F1 0526 CALL CALLX ;CALL ADDRESS PASSED IN HL
F28A B7 0527 OR A ;RETURN IF WE GET BACK AGAIN
0529 ;
0530 ;
0531 ;
0532 ; -- MEMORY READ/WRITE DIAGNOSTIC COMMAND --
0533 ;
F28C FE02 0534 TEST: CP 2 ;CHECK PARAMETER COUNT
F28E 37 0535 SCF
F28F C0 0536 RET NZ
F290 13 0537 INC DE
F291 5A 0538 LD E,D
F292 54 0539 LD D,H
F293 0600 0540 LD B,0 ;INITIALIZE PASS COUNTER
F295 62 0541 TEST1: LD H,D ;POINT HL TO START OF BLOCK
F296 2E00 0542 LD L,0
F298 7D 0543 TEST2: LD A,L ;GENERATE TEST BYTE
F299 AC 0544 XOR H
F29A AB 0545 XOR B
F29B 77 0546 LD (HL),A ;STORE BYTE IN RAM
F29C 23 0547 INC HL
F29D 7C 0548 LD A,H
F29E BB 0549 CP E ;CHECK FOR END OF TEST BLOCK
F29F 20F7 0550 JR NZ,TEST2-$
F2A0 0551 ;NOW READ BACK EACH BYTE & COMPARE
F2A1 62 0552 LD H,D
F2A2 2E00 0553 LD L,0 ;POINT HL BACK TO START
F2A4 7D 0554 TEST3: LD A,L ;RE-GENERATE TEST BYTE DATA
F2A5 AC 0555 XOR H
F2A6 AB 0556 XOR B
F2A7 CDB9F2 0557 CALL CHECK ;VERIFY MEMORY DATA STILL GOOD
F2AA C0 0558 RET NZ ;EXIT IF ESC RED IS INDICATED
F2AB  23  0559  INC  HL    ;ELSE GO TO NEXT BYTE
F2AC  7C  0560  LD  A, H
F2AD  BB  0561  CP  E    ;CHECK FOR END OF BLOCK
F2AE  20F4  0562  JR  NZ, TEST3-$
F2B0  04  0563  INC  B    ;BUMP PASS COUNT
F2B1  3E2B  0564  LD  A, '+'
F2B3  CD15F4  0565  CALL  OUTPUT  ;PRINT '+' AND ALLOW FOR EXIT
F2B6  28DD  0566  JR  Z, TEST1-$
F2BB  C9  0567  RET
F2B9  BE  0571  CHECK: CP  (HL)    ;RETURN IF (HL)=A
F2BA  C8  0572  RET  Z
F2BB  F5  0573  PUSH  AF
F2BC  CDCF2  0574  CALL  MDATA  ;PRINT WHAT WAS ACTUALLY READ
F2BF  CDECF3  0575  CALL  PNEXT
F2C2  73b6F75  0576  DEFM  'should='
F2C9  04  0577  DEFB  EOT
F2CA  F1  0578  POP  AF
F2CB  C3D2F3  0579  JP  PUT2HS  ;PRINT WHAT SHOULD HAVE READ
F2CE  CDCCF3  0582  MDATA: CALL  CRLFS
F2D1  CDCDF3  0583  CALL  PUT4HS
F2D4  7E  0584  LD  A, (HL)
F2D5  C3D2F3  0585  JP  PUT2HS
F2DB  FE03  0591  FILL: CP  3    ;CHECK IF PARAMETER COUNT=3
F2DA  37  0592  SCF
F2DB  C0  0593  RET  NZ
F2DC  71  0594  FILL1: LD  (HL), C
F2DD  E5  0595  PUSH  HL
F2DE  B7  0596  OR  A
F2DF  ED52  0597  SBC  HL, DE    ;COMP HL TO END ADDRESS IN DE
F2E1  E1  0598  POP  HL
F2E2  23  0599  INC  HL    ;ADVANCE POINTER AFTER COMPARE
F2E3  3B77  0600  JR  C, FILL1-$
F2E5  C9  0601  RET
F2E6  FE03  0608  BLOCK: CP  3    ;CHECK IF PARAMETER COUNT=3
F2EB  37  0609  SCF
F2E9  C0  0610  RET  NZ
F2EA  CDF3F2  0611  CALL  BLOCAD
F2ED  79  0612  LD  A, C
F2EE  B0  0613  OR  B
F2EF  CB  0614  RET  2    ;EXIT NOW IF BC=0
F2F0  EDB0  0615  LDIR
F2F2  C9  0616  RET
F2F3  EB  0620  BLOCAD: EX  DE, HL
F2F4  E7  0621  OR  A    ;CLEAR CARRY
F2F5  ED52  0622  SBC  HL, DE    ;GET DIFFERENCE BETWEEN
F2F7  EB  0627  EX  DE,HL  ;HL & DE FOR BYTECOUNT
F2FB  D5  0628  PUSH DE
F2FC  C5  0629  PUSH BC
F2FA  D1  062A  POP DE
F2FB  C1  062B  POP BC
F2FC  03  062C  INC BC
F2FD  C9  062D  RET BC

0630 ;
0631 ;
0632 ;
0633 ;
0634 ; -- READ FROM INPUT PORT COMMAND --
0635 ;
F2FE  3D  0636  INCMD: DEC A  ;CHECK IF PARAMETER COUNT=1
F2FF  37  0637  SCF
F300  C0  0638  RET NZ
F301  4D  0639  LD C,L
F302  CDFCF3  0640  IN1: CALL CRLFS
F305  79  0641  LD A.C
F306  CDD2F3  0642  CALL PUT2HS
F309  ED78  0643  IN A.(C)
F30B  CDF2F3  0644  CALL PUT2HS
F30E  CD07F4  0645  CALL ECHO
F311  FE0D  0646  CP CR
F313  2B06  0647  JR Z,IN2-$
F315  FE2D  0648  CP '-'
F317  2B04  0649  JR Z,IN3-$
F319  B7  0650  OR A
F31A  C9  0651  RET

F31B  0C  0652  IN2: INC C
F31C  0C  0653  INC C
F31D  0D  0654  IN3: DEC C
F31E  18E2  0655  JR IN1-#

0656 ;
0657 ;
0658 ;
0659 ;
0660 ; -- WRITE TO OUTPUT PORT COMMAND --
0661 ;
F320  FE02  0662  OUTCMD: CP 2  ;CHECK IF PARAMETER COUNT=2
F322  37  0663  SCF
F323  C0  0664  RET NZ
F324  4D  0665  LD C.L
F325  ED59  0666  OUT (C),E
F327  B7  0667  OR A
F328  C9  0668  RET

0669 ;
0670 ;
0671 ; -- SWITCH CONSOLE OUTPUT DEVICE COMMAND --
0672 ;
F329  2185FF  0673  SWITCH: LD HL,CFLAG
F32C  34  0674  INC (HL)
F32D  CB46  0675  BIT 0,(HL)
F32F  21FEF4  0676  LD HL,SIDOUT
F332  2B03  0677  JR Z,SWIT2-$
F334  2120F5  0678  LD HL,CRTOUT
F337  220DFF  0679  JR (CONOUT+1),HL
F33A  C9  0680  RET

0681 ;
0682 ;
0683 ;
0684 ;
0685 ;
0686 ;
0687 ;
0688 ;
0689 ;

*******************************************************************************

CONSOLE I/O PACKAGE AND UTILITY ROUTINES
*******************************************************************************
;SAVE MAX LINE LENGTH PARAM IN B
CALL ECHO ;GET A CHAR FROM THE CONSOLE
CR ;CHECK FOR CARRIAGE RETURN
Z,GLIN2-$$ ;CHECK FOR CTRL-H BACKSPACE
Z,GLIN4-$$ ;OTHER CONT CHARACTERS ILLEGAL
(HP),A ;STORE CHARACTER IN BUFFER
HL
C
NZ,GLIN1-$$ ;GET ANOTHER IF MORE ROOM
(HP),A ;RETURN WITH CARRY=1 IF TOO
;MANY CHARACTERS ARE ENTERED
RETURN WITH CARRY BIT=0
DEC HL ;DELETE LAST CHAR FROM BUFFER
PRINT A SPACE TO OVERWRITE THE
LAST CHAR, THEN DO A BACKSPACE
INC C
A,B ;MAKE SURE YOU'RE NOT TRYING TO
<BS> PAST START OF THE LINE
NC,GLIN1-$$ ;SRCH TBL $HL FOR MATCH WITH A
EXIT NOW IF SEARCH FAILS
NZ
ADD HL,BC 
+ RESIDUE FROM CPIR BYTE COUNT
ADD HL,BC 
TO HL 3 TIMES TO GET POINTER
ADD HL,BC 
TO ADDRESS PART OF TABLE ENTRY
INC HL
LD C,(HL)
LD B,(HL)
EXIT WITH Z=1 TO SHOW MATCH

LD BC,0
LD A,(IY+0)
CR ;CHECK IF LINE TERMINATES
NZ,PARA2-$$ ; IMMEDIATELY WITH A RETURN
A ;RET WITH PARAM COUNT=0 IF SO
INC C
INC C
BIT 3,C
<table>
<thead>
<tr>
<th>Address</th>
<th>Opcode</th>
<th>Bytes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>F37B</td>
<td>C0</td>
<td>0745</td>
<td>RET NZ; ERROR IF &gt; 4 NUMBERS ENTERED</td>
</tr>
<tr>
<td>F37C</td>
<td>C5</td>
<td>0746</td>
<td>PARA2; PUSH BC; SAVE PARAMETER COUNT</td>
</tr>
<tr>
<td>F37D</td>
<td>CDFF3</td>
<td>0747</td>
<td>CALL GETHEX; READ A NUMBER FROM LINE BUFFER</td>
</tr>
<tr>
<td>F380</td>
<td>C1</td>
<td>0748</td>
<td>POP BC</td>
</tr>
<tr>
<td>F381</td>
<td>D8</td>
<td>0749</td>
<td>PARA4; RET C; ERROR IF RESULT OVER 16 BITS</td>
</tr>
<tr>
<td>F382</td>
<td>D217CF</td>
<td>0750</td>
<td>LD IX,PARAM1; POINT TO PARAM STORAGE AREA</td>
</tr>
<tr>
<td>F386</td>
<td>DD09</td>
<td>0751</td>
<td>ADD IX,BC; ADD PARAMETER COUNT IN BC</td>
</tr>
<tr>
<td>F38B</td>
<td>DD7500</td>
<td>0752</td>
<td>LD (IX+0),L; STORE DATA RET FROM 'GETHEX'</td>
</tr>
<tr>
<td>F38B</td>
<td>DD7401</td>
<td>0753</td>
<td>LD (IX+1),H; ' '</td>
</tr>
<tr>
<td>F38E</td>
<td>FE20</td>
<td>0754</td>
<td>CP Z,PARA1-0; GET ANOTHER ITEM IF SPACE</td>
</tr>
<tr>
<td>F390</td>
<td>2BE4</td>
<td>0755</td>
<td>JR Z,PARA1-0; ' '</td>
</tr>
<tr>
<td>F392</td>
<td>FE2C</td>
<td>0756</td>
<td>CP Z,PARA1-0; GET ANOTHER ITEM IF COMMA</td>
</tr>
<tr>
<td>F394</td>
<td>2BE0</td>
<td>0757</td>
<td>JR Z,PARA1-0; ' '</td>
</tr>
<tr>
<td>F396</td>
<td>FE0D</td>
<td>0758</td>
<td>CP CR; ELSE CHECK FOR CARRIAGE RETURN</td>
</tr>
<tr>
<td>F398</td>
<td>37</td>
<td>0759</td>
<td>SCF; AND EXIT WITH CY=1 IF NOT</td>
</tr>
<tr>
<td>F399</td>
<td>C0</td>
<td>0760</td>
<td>RET NZ; AND EXIT WITH CY=1 IF NOT</td>
</tr>
<tr>
<td>F39A</td>
<td>79</td>
<td>0761</td>
<td>PAREND; LD A,C; ERROR IF &gt; 4 NUMBERS ENTERED</td>
</tr>
</tbody>
</table>
GETHEX CONVERTS ASCII TO BINARY AND DOES
HIGH LIMIT CHECKS TO LESS THAN 17 BITS.
CARRY SET ON ILLEGAL CONVERSION RESULT
TERMINATING CHARACTER RETURNS IN A.
HL RETURNS WITH 16 BIT BINARY INTEGER

LD HL, 0
JR GNUM3-
LD B, 4
ADD HL, HL
;MULTIPLY RESULT BY 16
RET C
DJNZ GNUM2-
;APPEND NEW LOW ORDER DIGIT
LD D, 0
;AND GET RESULT BACK INTO DE
ADD HL, DE
;RETURN IF OVERFLOW
LD A, (IY+0)
;GET A CHAR FROM LINE INPUT
CALL ASCHEX
;CONVERT ASCII TO NUMERIC
SUB '0'
RET C
CP 10
CCF
RET NC
SUB 7
CP 10
RET C
CP 16
CCF
RET
LD A, H
CALL PUT2HX
LD A, L
CALL PUT2HX
JP SPACE
PUSH AF
RRA
RRA
RRA
RRA
CALL PUTNIB
ADD 00001111B
F3E3 C690 0822 ADD A,90H
F3E5 27 0823 DAA
F3E6 CE40 0824 ADC A,40H
F3EB 27 0825 DAA
F3E9 C315F4 0826 JP OUTPUT

0827 
0828 
0829 ; PMSG PRINTS THE STRING OF ASCII CHARACTERS 
0830 ; POINTED TO BY THE RELATIVE ADDRESS IN DE 
0831 ; UNTIL AN EOT IS ENCOUNTERED IN THE STRING. 
0832 

0004 0833 EOT EQU 04H
000D 0834 CR EQU 0DH
000A 0835 LF EQU 0AH

0836 
0837
F3EC E3 0838 PNEXT: EX (SP),HL
F3ED CDF2F3 0839 CALL PMSG
F3F0 E3 0840 EX (SP),HL
F3F1  C9  0841  RET  
  0842  ;  
F3F2  7E  0843  PMSG:  LD  A,(HL)  
F3F3  23  0844  INC  HL  
F3F4  FE04  0845  CP  EOT  
F3F6  C8  0846  RET  Z  
F3F7  CD15F4  0847  CALL  OUTPUT  
F3FA  1BF6  0848  JR  PMSG-6  
  0849  ;  
  0850  ;  
  0851  ; CRLFS OUTPUTS A RETURN-LINEFEED-SPACE  
  0852  ; TO THE CONSOLE DEVICE  
  0853  ;  
F3FC  CDECF3  0854  CRLFS:  CALL  PNEXT  
F3FF  0D0A04  0855  DEFB  CR,LF,EOT  
F402  3E20  0856  SPACE:  LD  A,' '  
F404  C315F4  0857  JP  OUTPUT  
  0858  ;  
  0859  ;  
  0860  ; ECHO INPUTS ONE CHARACTER FROM THE CONSOLE  
  0861  ; DEVICE, PRINTS IT ON THE CONSOLE OUTPUT AND  
  0862  ; THEN RETURNS IT IN REGISTER A WITH BIT 7 RESET  
  0863  ;  
  0864  ;  
  0865  ; OUTPUT PRINTS THE CHARACTER IN REGISTER A ON  
  0866  ; THE CONSOLE OUTPUT DEVICE AND THEN DOES A CHECK  
  0867  ; FOR CONSOLE INPUT TO FREEZE OR ABORT OUTPUT.  
  0868  ;  
  0869  
F407  CD09F0  0870  ECHO:  CALL  CONIN  ; INPUT A CHARACTER AND ECHO IT  
F40A  F5  0871  PUSH  AF  
F40B  CD0CF0  0872  CALL  CONOUT  
F40E  F1  0873  POP  AF  
F40F  FE5B  0874  CP  'Z'+1  
F411  D8  0875  RET  C  
F412  D620  0876  SUB  32  ; CONVERT UPPER CASE TO LOWER  
F414  C9  0877  RET  
  0878  ;  
  0879  ;  
  0880  ;  
F415  CD0CF0  0881  OUTPUT:  CALL  CONOUT  ; SEE IF CONSOLE INPUT PENDING  
F418  CD06F0  0882  CALL  CONST  ; SEE IF <CR> WAS TYPED  
F418  2B0F  0883  JR  Z,OUTP2-$  
F41D  CD09F0  0884  CALL  CONIN  ; WAIT FOR ANOTHER INPUT CHAR  
F420  FE0D  0885  CP  CR  ; THEN RET TO CALLING ROUTINE  
F422  2B05  0886  JR  Z,OUTP1-$  
F424  CD09F0  0887  CALL  CONIN  ; RETURN CURRENT STATUS OF ESC  
F427  1803  0888  JR  OUTF2-$  
  0889  
F429  3284FF  0890  OUTP1:  LD  (ESCFLG),A  ; SET ESC FLAG TO NON-ZERO VALUE  
F42C  3AB4FF  0891  OUTP2:  LD  A,(ESCFLG)  
F42F  B7  0892  OR  A  ; FLAG TO CALLING ROUTINE  
F430  C9  0893  RET  
  0894  ;  
  0895  ;  
  0896  ;  
  0897  INCLUDE INTSRV.ASM
**INTERRUPT SERVICE Routines for Keyboard**

**Input and Real-Time Clock Functions**

J-Aug-80

```
F431 3A30FF 0909  KBDST:  LD  A,(FIFCNT) ; GET INPUT FIFO BYTECOUNT
F434  B7  0910  OR  A ; TEST IF EQUAL ZERO
F435  C8  0911  RET  Z ; EXIT WITH A=0 IF QUEUE EMPTY
F436  3EFF  0912  LD  A,255
F438  C9  0913  RET ; ELSE A=255 INDICATES DATA RDY
F439  CD31F4  0917  KBDIN:  CALL  KBDST
F43C  2BF8  0918  JR  Z,KBDIN-1 ; LOOP UNTIL KEYBOARD INPUT RDY
F43E  E5  0919  PUSH  HL
F43F  CD6DF4  0920  CALL  REMOVE ; GET CHARACTER FROM INPUT QUEUE
F442  E1  0921  POP  HL
F443  C9  0922  RET
F444  2133FF  0928  STASH:  LD  HL,LOCK ; POINT TO SHIFT LOCK VARIABLES
F447  BE  0929  CP  (HL) ; TEST IF A=SHIFT LOCK CHARACTER
F44B  23  0930  INC  HL ; THEN POINT TO LOCK FLAG
F44F  2002  0931  JR  NZ,STASH2-1 ; JUMP IF NOT SHIFT CHARACTER
F44B  34  0932  INC  (HL) ; ELSE COMPLIMENT THE SHIFT LOCK
F44C  C9  0933  RET ; AND EXIT NOW
F44D  C846  0935  STASH2:  BIT  0,(HL) ; TEST THE SHIFT LOCK FLAG
F44F  280A  0936  JR  Z,STASH3-1 ; JUMP IF SHIFT LOCK NOT SET
F451  FE40  0937  CP  40H ; ELSE CHECK FOR SHIFTABLE CHAR
F453  3B06  0938  JR  C,STASH3-1 ; AND JUMP IF NOT = OR GREATER
F455  FE7F  0939  CP  7FH ; THAN '0' AND LESS THAN RUBOUT
F457  3002  0940  JR  NC,STASH3-1
F459  EE30  0941  XOR  00100000B ; ELSE TOGGLE BIT 5 OF THE CHAR
F45B  4F  0942  STASH3:  LD  C,A
F45C  2130FF  0943  LD  HL,FIFCNT ; BUMP INPUT FIFO CHAR COUNT
F45F  7E  0944  LD  A,(HL)
F460  3C  0945  INC  A
F461  F10  0946  CP  16
F463  D0  0947  RET  NC ; EXIT NOW IF FIFO IS FULL
F464  77  0948  LD  (HL),A ; ELSE INCREMENT FIFO COUNT
F465  2131FF  0949  LD  HL,FIFIN ; POINT HL TO FIFO INPUT OFFSET
F468  CD74F4  0950  CALL  INDEX
F46B  71  0951  LD  (HL),C ; STORE CHARACTER IN FIFO @ HL
F46C  C9  0952  RET
```

F46D  2130FF  0957  REMOVE:  LD  HL,FIFCNT
F470  35    0958  DEC  (HL)
F471  2132FF  0959  LD  HL,FIFOUT ;POINT HL TO FIFO OUTPUT OFFSET
F474  7E    0960  INDEX:  LD  A,(HL)
F475  3C    0961  INC  A
F476  E60F  0962  AND  0000111B ;INCREMENT FIFO POINTER
F47B  77    0963  LD  (HL),A ;MODULO 16 AND REPLACE
F479  2120FF  0964  LD  HL,FIFO
F47C  B5    0965  ADD  A,L ;INDEX INTO FIFO BY OFFSET IN A
F47D  6F    0966  LD  L,A
F47E  7E    0967  LD  A,(HL)
F47F  C9    0968  RET
  0969 ;SOFTWARE DISK MOTOR TURN-OFF TIMER ROUTINE
            0970 ;
F480  216FF  0971  DSKMR:  LD  HL,MOTOR ;DECREMENT DISK TURN-OFF TIMER
F483  35    0972  DEC  (HL)
F484  C0    0973  RET  NZ ;EXIT IF NOT TIMED OUT YET
F485  DB1C  0974  IN  A,(BITDAT)
F487  F644  0975  OR  01000100B ;DISABLE ALL DRIVE SELECTS AND
F489  D31C  0976  OUT  (BITDAT),A ;TURN OFF THE SPINDLE MOTORS
F48B  C9    0977  RET
  0978 ;
F48C  ED735FF  0979  KEYSRV:  LD  (SPSAVE),SP ;SAVE USR STACK POINT AND
F490  3157FF  0980  LD  SP,TMPSTK+32;SWITCH TO LOCAL STACK
F493  E5    0981  PUSH  HL
F494  D5    0982  PUSH  DE
F495  C5    0983  PUSH  BC
F496  F5    0984  PUSH  AF ;SAVE MACHINE STATE
F497  DB1E  0985  IN  A,(KBDAT) ;READ KEYBOARD INPUT PORT
F499  2F    0986  CPL
F49A  2A59FF  0987  LD  HL,(PINVEC);GET KBD INTERRUPT RTN VECTOR
F49D  1822  0988  JR  DSPTCH-$ ;AND JUMP TO DISPATCH POINT
  0989 ;
F49F  ED735FF  0990  TIMER:  LD  (SPSAVE),SP ;SAVE USR STACK POINTER AND
F4A3  3157FF  0991  LD  SP,TMPSTK+32;SWITCH TO LOCAL STACK
F4A6  E5    0992  PUSH  HL
F4A7  D5    0993  PUSH  DE
F4A8  C5    0994  PUSH  BC
F4A9  F5    0995  PUSH  AF ;SAVE MACHINE STATE
F4AA  2A57FF  0996  LD  HL,(TIKVEC);GET CLOCK INTERRUPT RTN VECTOR
F4AD  1812  0997  JR  DSPTCH-$ ;AND JUMP TO DISPATCH POINT
  0998 ;
F4AF  ED735FF  0999  SIOINT:  LD  (SPSAVE),SP ;SAVE USER STACK POINTER AND
F4B3  3157FF  1000  LD  SP,TMPSTK+32;SWITCH TO LOCAL STACK
F4B6  E5    1001  PUSH  HL
F4B7  D5    1002  PUSH  DE
F4B8  C5    1003  PUSH  BC
F4B9  F5    1004  PUSH  AF ;SAVE MACHINE STATE
F4BA  DB05  1005  IN  A,(SIIDPB) ;READ SIO DATA INPUT PORT
F4BC  E67F  1006  AND  0111111B
F4BE  2A5BFF  1007  LD  HL,(SINVEC);GET SERIAL INPUT RTN VECTOR
; CALL SUBROUTINE ADDRESSED BY HL
; RE-ENABLE INTERRUPTS & RETURN

; RX ERROR INTERRUPT SERVICE ROUTINE FOR SIO --
; ARRIVE HERE IF RECEIVE INTERRUPT FROM FRAMING, OVERRUN
; AND PARITY ERRORS. (PARITY CAN BE DISABLED)

; SAVE USER STACK POINTER AND
; SWITCH TO LOCAL STACK

; CLEAR BAD CHARACTER FROM SIO

; OUTPUT A CTL-G AS A WARNING

; ACC=0 IF NO DATA AVAILABLE

; TEST CONSOLE STATUS
; ELSE SEND CONTROL CHARACTER
; AND THEN SEND NULLS AS PADDING
; THAT COUNT=0 SENDS NO NULLS

; TEST FOR CONTROL CHARACTERS
; JUMP IF PRINTABLE CHARACTER
; SEND CONTROL CHARACTER
; AND THEN SEND NULLS AS PADDING
; LOOP SENDING NULLS TO SIO

; null
GET CHAR OVERLAYERED BY CURSOR
LOAD HL WITH CURSOR POINTER
AND 0000111B ; INSURANCE THAT HL CAN'T
OR CRTBASE ; EVER POINT OUTSIDE CRT MEMORY
LD H,A
LD (HL),B ; RVY CURSOR BY RESTORING CHAR
; PROCESS CHARACTER PASSED IN C
CALL DOUTCH
; NOW STORE A NEW CURSOR CHARACTER AT THE CURSOR LOCATION
LD A,(HL) ; GET CHAR AT NEW CURSOR LOCAT.
LD (CHRSAVE),A ; SAVE FOR NEXT TIME 'COUT' IS CALLED
CP ' ' ; TEST IF CHARACTER IS A SPACE
SET 7,A ; THEN TURN ON ON BIT 7 TO ENABLE BLINK
JR NZ,CRT2-# ; JUMP IF CHARACTER IS NON-BLANK
LD A,(CSTRCHR) ; ELSE GET CHAR USED FOR CURSOR
LD (HL),A ; STORE CHAR IN A AS CURSOR MARK
LD (CURSOR),HL ; SAVE HL AS CURSOR POINTER
LD SP,(SPSAVE)
IN A,(BITDAT)
RES 7,A ;SWITCH BACK LOWER 16K OF RAM
OUT (BITDAT),A
EI
POP BC
;INTERRUPTS ARE SAFE AGAIN
POP DE
POP HL
RET

LD A,(DE) ;GET LEAD-IN SEQUENCE STATE
OR A
JP NZ,MULTI ;JUMP IF IN A LEAD-IN SEQUENCE
LD A,C ;ELSE PROCESS CHARACTER IN C
RET

LD (HL),C ;ELSE STORE DISPLAYABLE CHAR
INC HL ;AND ADV POINTER TO NEXT COLUMN
AND 01111111B ;EXTRACT COLUMN# FROM HL
LD a,l
CALL C ;EXIT IF NOT PAST COLUMN 79
RETURN ;ELSE DO AUTOMATIC <CR>
CALL LFEED ;AND LINEFEED
RET

PUSH HL
LD HL,CTLTAB ;SEARCH FOR CONTROL CHARACTER
CALL SEARCH ;HANDLING SUBROUTINE IN TABLE
CALL BC ,CHARSIZE ;/3
POP HL
RET
NZ ;EXIT IF NOT IMPLEMENTED

DEFB "",-64
DEFB "^",-64
DEFB "[",-64
DEFB "\",-64
DEFB ";",-64
DEFB ",",-64
DEFB ",",-64
DEFB ",&",-64
DEFB ",",-64
DEFB ";",-64
DEFB ",",-64
DEFB ",",-64
DEFB ",",-64
DEFB ",",-64
DEFB ",",-64
DEFB ",",-64
DEFB ",",-64
DEFB ",",-64
DEFB ",",-64
DEFB ",",-64
DEFB "",-64
DEFB "",-64
DEFB "",-64
DEFW BELL ;CTL-G IS THE BELL
DEFW BAKSPC ;CTL-H IS CURSOR LEFT
DEFW TAB ;CTL-I IS TAB
DEFW LFEED ;CTL-J IS CURSOR DOWN
DEFW UPPSR ;CTL-K IS CURSOR UP
DEFW FORSPC ;CTL-L IS CURSOR RIGHT
FSAB E7F5 1209 DEFW RETURN ;CTRL-M IS <CR>
FSAA 11F6 1210 DEFW CLEORES ;CTRL-Q CLEAR TO END-OF-SCREEN
FSAC 03F6 1211 DEFW CLEOREL ;CTRL-X IS CLEAR TO END-OF-LINE
FSAE ECF5 1212 DEFW CLRSCN ;CTRL-Z IS CLEAR SCREEN
FSB0 B6F5 1213 DEFW ESCAPE ;CTRL-[ IS ESCAPE
FSB2 6CF6 1214 DEFW HOMEPS ;CTRL-^ IS HOME UP
FSB4 BAF5 1215 DEFW STUFF ;CTRL-_- IS DISPLAY CONTROL CHARS

>0027 1216
1217 CTLSIZ EQU $-CTLTAB

1218
1219

FSB6 3E01 1220 ESCAPE: LD A,1 ;SET LEAD-IN SEQUENCE STATE
FSBB 12 1221 LD (DE),A ;FOR XY CURSOR POSITIONING MODE
FSB9 C9 1222 RET

1223

FSBA 3E04 1224 STUFF: LD A,4 ;SET LEAD-IN SEQUENCE STATE
FSBC 12 1225 LD (DE),A ;FOR CONTROL CHAR OUTPUT MODE
FSBD C9 1226 RET

1227

FSBE 7D 1228 BAKSC LD A,L ;CHECK FOR LEFT MARGIN
FSBF E67F 1229 AND 01111111B ;ABORT IF IN LEFTMOST COLUMN
FSC1 CB 1230 RET 2 ;BACK UP CURSOR POINTER
FSC2 2B 1231 DEC HL
FSC3 C9 1232 RET

1233

FSC4 7D 1234 FORSC LD A,L ;CHECK FOR RIGHTMOST COLUMN
FSC5 E67F 1235 AND 01111111B
FSC7 FE4F 1236 CP 79
FSC9 D0 1237 RET NC ;DO NOTHING IF ALREADY THERE
FSACA 23 1238 INC HL
FSCB C9 1239 RET ;ELSE ADVANCE CURSOR POINTER

1240

FSCC 110800 1241 TAB: LD DE,B ;TABS ARE EVERY 8 COLUMNS
FSCE 7D 1242 LD A,L ;GET COLUMN COMPONENT OF
FSDO E678 1243 AND 01111100B ;PREVIOUS TAB POSITION
FSDE 83 1244 ADD A,E
FSDF FE50 1245 CP 80 ;EXIT IF NEXT TAB COLUMN WOULD
FSDD D0 1246 RET NC ;BE PAST THE RIGHT MARGIN
FSDE 7D 1247 LD A,L
FSDF E6F8 1248 AND 1111100B ;ELSE INCREMENT THE CURSOR
FSDA 19 1249 LD "L,A ;POINTER FOR REAL
FSDB C9 1250 ADD HL,DE

1251

FSCB DB1C 1252 BELL: IN A,(BITDAT)
FSED CBEF 1253 SET 5,A ;TOGGLE BIT 5 OF SYSTEM PIO TO
FSE0 D31C 1254 OUT (BITDAT),A ;TRIGGER BELL HARDWARE TO SOUND
FSEX CBAF 1255 RES 5,A
FSES D31C 1256 OUT (BITDAT),A
FSEC C9 1257 RET

1258

FSE7 7D 1259 RETURN: LD A,L
FSE8 E680 1260 AND 10000000B ;MOVE CURSOR POINTER BACK
FSEA 6F 1261 LD L,A ;TO START OF LINE
FSEB C9 1262 RET

1263

FSEC 210030 1264 CLRSCN: LD HL,CRTMEM
F5E5 E5 1277 PUSH HL
F5F0 110130 1274 LD DE, CRTMEM + 1
F5F3 01000C 1275 LD BC, 24 * 128
F5F6 3620 1276 LD (HL), ; FILL CRT MEMORY WITH SPACES
F5F8 E6B0 1277 LDIR HL ; POINT TO HOME CURSOR POSITION
F5FA E1 1278 POP HL
F5FB 3E17 1279 LD A, 23
F5FD 3277FF 1280 LD (BASE), A ; MAKE BASE LINE# BE 23 AND
F600 D314 1281 OUT (SCROLL), A ; STORE IN SCROLL REGISTER
F602 C9 1282 RET

F603 E5 1285 CLREOL: PUSH HL ; SAVE CURSOR POINTER
F604 7D 1286 LD A, L
F605 E67F 1287 AND 01111111B ; GET COLUMN# COMPONENT OF
F607 4F 1288 LD C, A ; CURSOR POINTER INTO C
F608 3E50 1289 LD A, 80 ; CALCULATE HOW MANY CHAR
F60A 91 1290 SUB C ; REMAIN ON CURRENT LINE
F60B 47 1291 LD B, A
F60C CD66F6 1292 CALL CLR ; CLEAR REST OF LINE & HL
F60F E1 1293 POP HL
F610 C9 1294 RET

F611 CD03F6 1297 CLREOS: CALL CLREOL ; CLEAR REMAINDER OF CURRENT ROW
F614 E5 1298 PUSH HL
F615 3A77FF 1299 LD A, (BASE)
F618 4F 1300 LD C, A ; COPY BASE SCREEN ROW# TO C
F619 7D 1301 CLRS1: LD A, L ; ROW# COMPONENT OF HL INTO A
F61A 17 1302 RLA
F61B 7C 1303 LD A, H
F61C 17 1304 RLA
F61D E61F 1305 AND 00011111B ; SEE IF HL IS AT BOTTOM ROW
F61F B9 1306 CP C OF SCREEN

F620 280B 1307 JR Z, CLRS2-$ ; AND LEAVE CLEAR LOOP IF SO
F622 CD37F6 1308 CALL DNSCR ; ELSE POINT HL TO NEXT ROW DOWN
F625 CD60F6 1309 CALL CLRLIN ; AND FILL THAT LINE WITH SPACES
F628 18EF 1310 JR CLRS1-$
F62A E1 1312 CLRS2: POP HL ; RESTORE ORIGINAL CURSOR POINTER
F62B C9 1313 RET
F62D 1314 ;
F62C 11B0FF 1316 UPCSR: LD DE, -128 ; SUBTRACT 1 FROM ROW# COMPONENT
F62F 19 1317 ADD HL, DE ; OF CURSOR POINTER IN HL
F630 7C 1318 LD A, H
F631 FE30 1319 CP CRTRAS ; CHECK FOR UNDERFLOW OF POINTER
F633 D0 1320 RET NC
F634 263B 1321 LD H, CRTROP - 1 ; WRAP CURSOR AROUND MODULO 3K
F636 C9 1322 RET
F638 1323 ;
F637 11B000 1325 DNCSR: LD DE, 128 ; ADD 1 TO ROW# COMPONENT
F63A 19 1326 ADD HL, DE ; OF CURSOR POINTER IN HL
F63B 7C 1327 LD A, H
F63C FE3C 1328 CP CRTRAS ; CHECK FOR OVERFLOW OF POINTER
F63E D8 1329 RET C
F63F 2630 1330 LD H, CRTRAS ; RESET POINTER MODULO 128 * 24
F641 C9 1331 RET
F642 7D 1332 ;
F643 17 1336 RLA
F644 7C 1337 LD A, H
F645 1338 ;
F646 7D 1339 LFEED: LD A, L
<table>
<thead>
<tr>
<th>Address</th>
<th>Opcode</th>
<th>Mnemonic</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>F645</td>
<td>17</td>
<td>LDA</td>
<td>1338</td>
</tr>
<tr>
<td>F646</td>
<td>E61F</td>
<td>AND</td>
<td>1339</td>
</tr>
<tr>
<td>F64B</td>
<td>4F</td>
<td>LD</td>
<td>1340</td>
</tr>
<tr>
<td>F649</td>
<td>CD37F6</td>
<td>CALL</td>
<td>1341</td>
</tr>
<tr>
<td>F64C</td>
<td>3A77FF</td>
<td>LD</td>
<td>1342</td>
</tr>
<tr>
<td>F64F</td>
<td>B9</td>
<td>CP</td>
<td>1343</td>
</tr>
<tr>
<td>F650</td>
<td>C0</td>
<td>RET</td>
<td>1344</td>
</tr>
<tr>
<td>F651</td>
<td>E5</td>
<td>APP</td>
<td>1345</td>
</tr>
<tr>
<td>F652</td>
<td>CD60F6</td>
<td>CALL</td>
<td>1346</td>
</tr>
<tr>
<td>F655</td>
<td>29</td>
<td>ADD</td>
<td>1347</td>
</tr>
<tr>
<td>F656</td>
<td>7C</td>
<td>LD</td>
<td>1348</td>
</tr>
<tr>
<td>F657</td>
<td>E61F</td>
<td>AND</td>
<td>1349</td>
</tr>
<tr>
<td>F659</td>
<td>3277FF</td>
<td>LD</td>
<td>1350</td>
</tr>
<tr>
<td>F65C</td>
<td>D314</td>
<td>OUT</td>
<td>1351</td>
</tr>
<tr>
<td>F65E</td>
<td>E1</td>
<td>POP</td>
<td>1352</td>
</tr>
<tr>
<td>F65F</td>
<td>C9</td>
<td>RET</td>
<td>1353</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1354</td>
</tr>
<tr>
<td>F660</td>
<td>7D</td>
<td>LD</td>
<td>1355</td>
</tr>
<tr>
<td>F661</td>
<td>E800</td>
<td>AND</td>
<td>1356</td>
</tr>
<tr>
<td>F663</td>
<td>6F</td>
<td>LD</td>
<td>1357</td>
</tr>
<tr>
<td>F664</td>
<td>0650</td>
<td>LD</td>
<td>1358</td>
</tr>
<tr>
<td>F666</td>
<td>3620</td>
<td>LD</td>
<td>1359</td>
</tr>
<tr>
<td>F66B</td>
<td>23</td>
<td>INC</td>
<td>1360</td>
</tr>
<tr>
<td>F669</td>
<td>10FB</td>
<td>DJNZ</td>
<td>1361</td>
</tr>
<tr>
<td>F66B</td>
<td>C9</td>
<td>RET</td>
<td>1362</td>
</tr>
<tr>
<td>F66C</td>
<td>0E20</td>
<td>LD</td>
<td>1363</td>
</tr>
<tr>
<td>F66E</td>
<td>1817</td>
<td>JR</td>
<td>1364</td>
</tr>
<tr>
<td>F670</td>
<td>EB</td>
<td>EX</td>
<td>1365</td>
</tr>
<tr>
<td>F671</td>
<td>3600</td>
<td>LD</td>
<td>1366</td>
</tr>
<tr>
<td>F673</td>
<td>EB</td>
<td>EX</td>
<td>1367</td>
</tr>
<tr>
<td>F674</td>
<td>FE01</td>
<td>CP</td>
<td>1368</td>
</tr>
<tr>
<td>F676</td>
<td>2008</td>
<td>JR</td>
<td>1369</td>
</tr>
<tr>
<td>F67B</td>
<td>79</td>
<td>LD</td>
<td>1370</td>
</tr>
<tr>
<td>F679</td>
<td>FE3D</td>
<td>CP</td>
<td>1371</td>
</tr>
<tr>
<td>F67B</td>
<td>C0</td>
<td>RET</td>
<td>1372</td>
</tr>
<tr>
<td>F67C</td>
<td>3E02</td>
<td>LD</td>
<td>1373</td>
</tr>
<tr>
<td>F67E</td>
<td>12</td>
<td>LD</td>
<td>1374</td>
</tr>
<tr>
<td>F67F</td>
<td>C9</td>
<td>RET</td>
<td>1375</td>
</tr>
<tr>
<td>F680</td>
<td>FE02</td>
<td>CP</td>
<td>1376</td>
</tr>
<tr>
<td>F682</td>
<td>2019</td>
<td>JR</td>
<td>1377</td>
</tr>
<tr>
<td>F684</td>
<td>3E03</td>
<td>LD</td>
<td>1378</td>
</tr>
<tr>
<td>F686</td>
<td>12</td>
<td>LD</td>
<td>1379</td>
</tr>
<tr>
<td>F687</td>
<td>3A77FF</td>
<td>LD</td>
<td>1380</td>
</tr>
<tr>
<td>F68A</td>
<td>B1</td>
<td>ADD</td>
<td>1381</td>
</tr>
<tr>
<td>F68B</td>
<td>D61F</td>
<td>SUB</td>
<td>1382</td>
</tr>
<tr>
<td>F68D</td>
<td>D61B</td>
<td>SUB</td>
<td>1383</td>
</tr>
<tr>
<td>F68F</td>
<td>30FC</td>
<td>JR</td>
<td>1384</td>
</tr>
<tr>
<td>F691</td>
<td>61B</td>
<td>ADD</td>
<td>1385</td>
</tr>
<tr>
<td>F693</td>
<td>F660</td>
<td>OR</td>
<td>1386</td>
</tr>
<tr>
<td>F695</td>
<td>67</td>
<td>LD</td>
<td>1387</td>
</tr>
<tr>
<td>F696</td>
<td>2E00</td>
<td>LD</td>
<td>1388</td>
</tr>
<tr>
<td>F69B</td>
<td>CB3C</td>
<td>SRL</td>
<td>1389</td>
</tr>
<tr>
<td>F69A</td>
<td>CB1D</td>
<td>RR</td>
<td>1390</td>
</tr>
<tr>
<td>F69C</td>
<td>C9</td>
<td>RET</td>
<td>1391</td>
</tr>
<tr>
<td>F69D</td>
<td>FE03</td>
<td>CP</td>
<td>1392</td>
</tr>
<tr>
<td>F69F</td>
<td>200C</td>
<td>JR</td>
<td>1393</td>
</tr>
</tbody>
</table>

**F645** - Extract row# component of HL
**F646** - Copy row# to C for scroll test
**F64B** - Move cursor to next row down
**F649** - Test if cursor on bottom row
**F64C** - Of screen before moving down
**F650** - Exit if not at bottom

---

**F660** - Get row# part of HL into A
**F661** - Fill new bottom line wth spaces
**F663** - Store new base line
**F664** - Scroll up new blank bottom line

---

**F665** - Point HL to 1st column of row
**F666** - Store ASCII spaces at addr in HL
**F668** - And increment HL
**F669** - Repeat number of times in B

---

**F66A** - Fake-out cursor addr routine
**F66B** - To do HomeUp almost for free

---

**F66C** - Unconditionally reset lead-in state to zero before going on
**F66D** - Get second char of sequence
**F66E** - Abort sequence if not "=
**F66F** - Make leadin=2 next time

---

**F670** - Make leadin=3 next time
**F671** - Arrive here on third char
**F673** - Of ESC="",row,col sequence
**F674** - Make leadin=2 next time
**F676** - Verify row# between 0 and 23

---

**F67A** - Merge in MSB's of CRT Memory
**F67B** - CRTMEM.SHR.7
<table>
<thead>
<tr>
<th>Register</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>F6A1</td>
<td>79</td>
</tr>
<tr>
<td>F6A2</td>
<td>D620</td>
</tr>
<tr>
<td>F6A4</td>
<td>D650</td>
</tr>
<tr>
<td>F6A6</td>
<td>30FC</td>
</tr>
<tr>
<td>F6AB</td>
<td>C650</td>
</tr>
<tr>
<td>F6AA</td>
<td>B5</td>
</tr>
<tr>
<td>F6AB</td>
<td>6F</td>
</tr>
<tr>
<td>F6AC</td>
<td>C9</td>
</tr>
</tbody>
</table>

---

**Disassembled Code:**

```
1402 SETCOL: LD A,C ;arrive here on fourth char
1403 SUB '' ;of ESC,','=','ROW,COL SEQUEN
1404 SETC2: SUB 80 ;MAKE SURE COL# BETWEEN 0 & 79
1405 JR NC,SETC2- ;MERGE IN COL# WITH L
1406 ADD A,80 ;PASSED IN C
1407 DR L
1408 LD L,A
1410 M4TST: CALL DISPLA ;DISPLAY THE CONTROL CHAR
1412 RET

---

**Include File:**

```
1418 ;******************************************************************************
1419 ;******** DISK INPUT/OUTPUT DRIVER SUBROUTINE PACKAGE ********
1420 ; FOR WESTERN DIGITAL 1771 DISK CONTROLLER ********
1422 ; bullet-proof error recovery added 12-APR-80 ********
1425 ;******************************************************************************
1426 ;
1427 ;EQUATES FOR DISK CONTROLLER PORTS AND COMMAND CODES
1429 ;
1430 STSREG EQU WD1771+0 ;STATUS REGISTER
1431 CMDREG EQU WD1771+0 ;COMMAND REGISTER
1432 TRKREG EQU WD1771+1 ;TRACK REGISTER
1433 SECREG EQU WD1771+2 ;SECTOR REGISTER
1434 DATREG EQU WD1771+3 ;DATA REGISTER
1435 ;
1436 RDCMD EQU 10001000B ;READ COMMAND
1437 WRTCMD EQU 10101000B ;WRITE COMMAND
1438 SKCMD EQU 00011100B ;SEEK COMMAND
1439 FICMD EQU 11010000B ;FORCE INTR COMMAND
1440 RSTCMD EQU 00001100B ;RESTORE COMMAND
1441 HLOAD EQU 00000100B ;RD/WRT HEAD LOAD ENABLE
1442 ;
1443 RET EQU 0C9H ;SUBROUTINE RETURN INSTR_OPCODE
1444 NMIVEC EQU 0066H ;THE NON-MASKABLE INTERRUPT IS
1445 ;USED FOR DATA SYNC BETWEEN
1446 ;THE Z-80 AND 1771
1447 ;
1448 ;
1449 ;
```

---

**Notes:**

- The code appears to be written in assembly language, targeting the Western Digital 1771 Disk Controller.
- The assembly instructions include setting control registers, reading data, writing commands, and handling interrupts.
- The comments indicate the purpose of each segment of code, such as checking for maximum valid angle, merging disk unit number, and handling error conditions.
- The use of hexadecimal values indicates the use of specific drive select and command codes.
- The use of `DD`, `DB`, and `DL` registers suggests interaction with disk drives and data transmission.
; AND CONTINUE IF ITS READY
LD A,B
(BITDAT),A ; ELSE PUT BACK OLD DRIVE SELECT
OUT A,10000000B ; AND RETURN DRIVE NOT READY
HL,UNIT
OUT A,(HL)
LD (HL),C
AND STORE NEW UNIT# FROM C
255 ; TEST IF NO DRIVE SELECTED
HL
OUT A,L,A
Z,S,R
; YET & SKIP NEXT SEGMENT IF SO
HL
IN L,A
; POINT HL TO DRIVE SELECT DATA
ADD A,(HL)
LD L,A
GET CURRENT HEAD POSITION
L,A
OUT (HL),A
AND STORE IN TABLE 0 HL
HL,TRKB
LD A,L
INDEX INTO TABLE TO GET
HEAD POSITION OF NEW DRIVE
L,A
OUT A
A,(HL)
; TEST IF NEW DRIVE WAS EVER
Z,HOME
SELECTED AND DO A HOME IF NOT
OUT (TRKREG),A
; OUTPUT DRIVE'S CURRENT HEAD
; POSITION TO THE TRACK REGISTER

; CLEAR DISK CONTROLLER
CALL READY
; EXIT IF DRIVE NOT READY
NZ
ENTRY
XOR A
; SET TRACK# IN MEM TO ZERO
XOR (TRACK),A
; LOAD B WITH A RESTORE COMMAND
B,RSTCMD
; EXECUTE HEAD MOVING OPERATION
; GET TRUE TRACK 0 STATUS
XOR 000000100B
; MASK TO ERROR BITS
XOR 00011100B
; RETURN 1771 STATUS IN A
XOR 0011000B
; CLEAR DISK CONTROLLER
CALL READY
; EXIT IF DRIVE NOT READY
NZ
ENTRY
XOR A
; SET TRACK# DATA FROM C AND
XOR (TRACK),A
; CHECK FOR MAXIMUM VALID#
CP 77
; FORGET IT IF TRACK# > 76
EXIT
; ELSE STORE TRACK# FOR SEEK
OUT (DATREG),A
; OUTPUT TRACK # TO 1771
OUT B,SKCMD
; LOAD B WITH A SEEK COMMAND AND
LD A,C
; GO SEEK WITH PROPER STEP RATE
LD 10011000B
; MASK TO READY,SEEK & CRC ERROR
XOR 0011000B
; BITS AND RETURN IF ALL GOOD
XOR 001000B
; ELSE TRY TO RE-CALIBRATE HEAD
RET A
; ERROR IF WE CAN'T FIND TRACK 0
NZ
ENTRY
XOR A
; OUTPUT TRACK# TO 1771
XOR (DATREG),A
; TRY TO SEEK THE TRACK AGAIN
LD B,SKCMD
LD 10011000B
; RETURN FINAL SEEK STATUS IN A
XOR 001000B
; CLEAR DISK CONTROLLER
CALL READY
; EXIT IF DRIVE NOT READY
NZ
ENTRY
XOR A
; SET TRACK# DATA FROM C AND
XOR (TRACK),A
; CHECK FOR MAXIMUM VALID#
CP 77
; FORGET IT IF TRACK# > 76
EXIT
; ELSE STORE TRACK# FOR SEEK
OUT (DATREG),A
; OUTPUT TRACK # TO 1771
OUT B,SKCMD
; LOAD B WITH A SEEK COMMAND AND
LD A,C
; GO SEEK WITH PROPER STEP RATE
LD 10011000B
; MASK TO READY,SEEK & CRC ERROR
XOR 0011000B
; BITS AND RETURN IF ALL GOOD
XOR 001000B
; ELSE TRY TO RE-CALIBRATE HEAD
RET A
; ERROR IF WE CAN'T FIND TRACK 0
NZ
ENTRY
XOR A
; OUTPUT TRACK# TO 1771
XOR (DATREG),A
; TRY TO SEEK THE TRACK AGAIN
LD B,SKCMD
LD 10011000B
; RETURN FINAL SEEK STATUS IN A
XOR 001000B
; CLEAR DISK CONTROLLER
CALL READY
; EXIT IF DRIVE NOT READY
NZ
ENTRY
XOR A
; SET TRACK# DATA FROM C AND
XOR (TRACK),A
; CHECK FOR MAXIMUM VALID#
CP 77
; FORGET IT IF TRACK# > 76
EXIT
; ELSE STORE TRACK# FOR SEEK
OUT (DATREG),A
; OUTPUT TRACK # TO 1771
OUT B,SKCMD
; LOAD B WITH A SEEK COMMAND AND
LD A,C
; GO SEEK WITH PROPER STEP RATE
LD 10011000B
; MASK TO READY,SEEK & CRC ERROR
XOR 0011000B
; BITS AND RETURN IF ALL GOOD
XOR 001000B
; ELSE TRY TO RE-CALIBRATE HEAD
RET A
; ERROR IF WE CAN'T FIND TRACK 0
NZ
ENTRY
XOR A
; OUTPUT TRACK# TO 1771
XOR (DATREG),A
; TRY TO SEEK THE TRACK AGAIN
LD B,SKCMD
LD 10011000B
; RETURN FINAL SEEK STATUS IN A
XOR 001000B
F72F CDABF7 1525 WRITE: CALL READY ; CLEAR THE DISK CONTROLLER
F722 C0 1526 RET NZ ; EXIST IF DRIVE NOT READY
F723 CB77 1526 BIT 6,A
F725 C0 1527 RET NZ ; EXIT IF DISK WRITE-PROTECTED
F726 068B 1528 LD B,WRDMD
F728 1806 1529 JR RDWRT-$

F72A CDABF7 1531 READY: READ CALL READY ; CLEAR DISK CONTROLLER
F72D C0 1532 RET NZ ; EXIST IF DRIVE NOT READY
F72E 068B 1533 LD B,RCMD
F730 2271FF 1534 RDWRT: LD (IDPTR),HL ; STORE DISK I/O DATA POINTER
F733 216EFF 1535 LD HL,SECTOR
F736 71 1536 LD (HL),C
F737 23 1537 INC HL
F738 70 1538 LD (HL),B
F739 23 1539 INC HL
F73A 3602 1540 LD HL,RELCEN
F73C F3 1541 DI
F73D 216600 1542 LD HL,NMIVEC
F740 56 1543 LD D,(HL) ; SAVE BYTE AT NMI VECTOR LOCAT
F741 36C9 1544 LD (HL),RET
F743 216FF 1545 LD HL,RELCEN
F746 46 1546 LD B,(HL) ; B=NUMBER OF BYTES/SECTOR
F747 0E13 1547 LD C,DATREG
F749 2A71FF 1548 LD HL,(IDPTR) ; HL=DISK R/W DATA POINTER
F74C 3A6EFF 1549 LD A,(SECTOR) ; GET SECTOR NUMBER
F74F D312 1550 OUT (SECREG),A ; OUTPUT SECTOR# TO 1771
F751 CDAEF7 1551 CALL FORCE ; ISSUE FORCE INTERRUPT COMMAND
F754 CB6F 1552 BIT 5,A
F756 3A6FF 1553 LD A,(CMDTYP) ; GET READ OR WRITE COMMAND BYTE
F759 2002 1554 JR NZ,RW2-$ ; JUMP IF HEAD IS ALREADY LOADED
F75B F604 1555 LD HLOAD ; ELSE MERGE IN HLD BIT
F75D CDA2F7 1556 RW2: CALL CMDOUT ; START 1771 DOING IT'S THING
F760 CB6F 1557 BIT 5,A
F762 200D 1558 JR NZ,WLOOP-# ; TEST IF COMMAND IS A R OR W
F764 76 1559 RLOOP: HALT ; AND JUMP TO THE CORRECT LOOP
F765 EDAD 1560 INIT
F767 C264F7 1561 JP BUSY
F768 CD9CF7 1562 CALL 10011100B ; MASK OFF TO READY, NOT FOUND.CRC
F76D E69C 1563 AND RW3-$ ; AND LOST DATA STATUS BITS
F76F 180B 1564 JR NZ,RLOOP ; LOOP UNTIL 1771 COMES UN-BUSY
F771 76 1565 WLOOP: HALT ; N2,WLOOP
F772 EDAD 1566 OUTI ; BUSY
F774 C271F7 1567 JP N2,WLOOP
F777 CD9CF7 1568 CALL 10111100B ; MASK OFF AS ABOVE + WRT FAULT
F778 1570 AND HL,NMIVEC
F779 CB 1571 RW3: LD (HL),D
F77F 72 1572 LD E1
F780 FB 1573 RET
F781 CB 1574 A
F782 2170FF 1575 LD HL,RETRY
F785 35 1576 INC HL
F786 2002 1577 JR NZ,RW4-$ ; DECREMENT RE-TRY COUNT AND
F788 B7 1578 OR A
F789 C9 1579 RTC
F78A 1580
F78C 0801 1581 RW4: LD HL,TRACK
F78D 4E 1582 LD C,(HL)
F78E CDFBF6 1583 CALL SEEK
F791 18A9 1584 JR RW1-$ ; BEFORE READ OR WRITE AGAIN
F795 ;
<table>
<thead>
<tr>
<th>Address</th>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>F793</td>
<td>3A6AFF</td>
<td>1588 STEP: 3</td>
</tr>
<tr>
<td>F796</td>
<td>E603</td>
<td>1589 AND</td>
</tr>
<tr>
<td>F798</td>
<td>B0</td>
<td>1590 OR</td>
</tr>
<tr>
<td>F799</td>
<td>CDA3F7</td>
<td>1591 CALL</td>
</tr>
<tr>
<td>F79C</td>
<td>DB10</td>
<td>1592 IN</td>
</tr>
<tr>
<td>F79E</td>
<td>CB47</td>
<td>1593 BIT</td>
</tr>
<tr>
<td>F7A0</td>
<td>20FA</td>
<td>1594 JR</td>
</tr>
<tr>
<td>F7A2</td>
<td>C9</td>
<td>1595 RET</td>
</tr>
<tr>
<td>F7A3</td>
<td>D310</td>
<td>1596 1598</td>
</tr>
<tr>
<td>F7A5</td>
<td>CDA8F7</td>
<td>1599 CALL</td>
</tr>
<tr>
<td>F7A6</td>
<td>E3</td>
<td>1600_pause</td>
</tr>
<tr>
<td>F7A7</td>
<td>E3</td>
<td>1601_PAUSE</td>
</tr>
<tr>
<td>F7AA</td>
<td>C9</td>
<td>1602_EX</td>
</tr>
<tr>
<td>F7AB</td>
<td>CDB8F7</td>
<td>1603_CALL</td>
</tr>
<tr>
<td>F7AE</td>
<td>3ED0</td>
<td>1604_FORCE</td>
</tr>
<tr>
<td>F7B0</td>
<td>CDA3F7</td>
<td>1605_PAUSE</td>
</tr>
<tr>
<td>F7B1</td>
<td>DB10</td>
<td>1606_CALL</td>
</tr>
<tr>
<td>F7B5</td>
<td>CB7F</td>
<td>1607_READY</td>
</tr>
<tr>
<td>F7B7</td>
<td>C9</td>
<td>1608_RST</td>
</tr>
<tr>
<td>F7BB</td>
<td>3E1E</td>
<td>1609_REQ</td>
</tr>
<tr>
<td>F7BA</td>
<td>36CF0</td>
<td>1611_REQ</td>
</tr>
<tr>
<td>F7BD</td>
<td>CDA8F7</td>
<td>1612_PAUSE</td>
</tr>
<tr>
<td>F7C0</td>
<td>DB1C</td>
<td>1613_REQ</td>
</tr>
<tr>
<td>F7C2</td>
<td>CB57</td>
<td>1614_REQ</td>
</tr>
<tr>
<td>F7C4</td>
<td>CB</td>
<td>1615_REQ</td>
</tr>
<tr>
<td>F7C5</td>
<td>E6BB</td>
<td>1616_REQ</td>
</tr>
<tr>
<td>F7C7</td>
<td>D31C</td>
<td>1617_REQ</td>
</tr>
<tr>
<td>F7C9</td>
<td>C5</td>
<td>1618_REQ</td>
</tr>
<tr>
<td>F7CA</td>
<td>0600</td>
<td>1619_REQ</td>
</tr>
<tr>
<td>F7CC</td>
<td>CDDC7</td>
<td>1620_REQ</td>
</tr>
<tr>
<td>F7CF</td>
<td>2B02</td>
<td>1621_REQ</td>
</tr>
<tr>
<td>F7D1</td>
<td>10F9</td>
<td>1622_REQ</td>
</tr>
<tr>
<td>F7D3</td>
<td>0609</td>
<td>1623_DEPTH</td>
</tr>
<tr>
<td>F7D5</td>
<td>CDDC7</td>
<td>1624_DEPTH</td>
</tr>
<tr>
<td>F7DB</td>
<td>C9</td>
<td>1625_DEPTH</td>
</tr>
<tr>
<td>F7DC</td>
<td>DB1B</td>
<td>1626_DEPTH</td>
</tr>
<tr>
<td>F7DE</td>
<td>4F</td>
<td>1627_DEPTH</td>
</tr>
<tr>
<td>F7DF</td>
<td>DB1B</td>
<td>1628_DEPTH</td>
</tr>
<tr>
<td>F7E1</td>
<td>B9</td>
<td>1629_DEPTH</td>
</tr>
<tr>
<td>F7E2</td>
<td>2BFB</td>
<td>1630_DEPTH</td>
</tr>
<tr>
<td>F7E4</td>
<td>18CB</td>
<td>1631_DEPTH</td>
</tr>
<tr>
<td>F7E6</td>
<td>0000</td>
<td>1632_DEPTH</td>
</tr>
<tr>
<td>F7E8</td>
<td>0000</td>
<td>1633_DEPTH</td>
</tr>
<tr>
<td>F7E9</td>
<td>0000</td>
<td>1634_DEPTH</td>
</tr>
<tr>
<td>F7EA</td>
<td>0000</td>
<td>1635_DEPTH</td>
</tr>
<tr>
<td>F7EC</td>
<td>0000</td>
<td>1636_DEPTH</td>
</tr>
<tr>
<td>F7ED</td>
<td>0000</td>
<td>1637_DEPTH</td>
</tr>
<tr>
<td>F7EE</td>
<td>0000</td>
<td>1638_DEPTH</td>
</tr>
<tr>
<td>F7EF</td>
<td>0000</td>
<td>1639_DEPTH</td>
</tr>
<tr>
<td>F7F0</td>
<td>0000</td>
<td>1640_DEPTH</td>
</tr>
<tr>
<td>F7F2</td>
<td>0000</td>
<td>1641_DEPTH</td>
</tr>
<tr>
<td>F7FC</td>
<td>0000</td>
<td>1642_DEPTH</td>
</tr>
<tr>
<td>F7FD</td>
<td>0000</td>
<td>1643_DEPTH</td>
</tr>
<tr>
<td>F7FE</td>
<td>0000</td>
<td>1644_DEPTH</td>
</tr>
<tr>
<td>F7FF</td>
<td>0000</td>
<td>1645_DEPTH</td>
</tr>
<tr>
<td>FF00</td>
<td>0049</td>
<td>1646_DEPTH</td>
</tr>
<tr>
<td>FF0F</td>
<td>0058</td>
<td>1647_DEPTH</td>
</tr>
</tbody>
</table>
ERROR IF > 4 NUMBERS ENTERED
SAVE PARAMETER COUNT
READ A NUMBER FROM LINE BUFFER
ERROR IF RESULT OVER 16 BITS
POINT TO PARAM STORAGE AREA
ADD PARAMETER COUNT IN BC
STORE DATA RET FROM 'GETHEX'
GET ANOTHER ITEM IF SPACE
GET ANOTHER ITEM IF COMMA
ELSE CHECK FOR CARRIAGE RETURN
AND EXIT WITH CY=1 IF NOT
; GETHEX CONVERTS ASCII TO BINARY AND DOES
; HIGH LIMIT CHECKS TO LESS THAN 17 BITS.
; CARRY SET ON ILLEGAL CONVERSION RESULT
; TERMINATING CHARACTER RETURNS IN A.
; HL RETURNS WITH 16 BIT BINARY INTEGER

F39F 210000
F3A2 180B
F3A4 0604
F3A6 29
F3A7 DB
F3A8 10FC
F3AA 5F
F3AB 1600
F3AD 19
F3AE DB
F3AF FD7E00
F3B2 FD23
F3B4 4F
F3B5 CDBDF3
F3BB 30EA
F3BA 79
F3BB B7
F3BC C9
F3BD D630
F3BF DB
F3C0 FE0A
F3C2 3F
F3C3 D0
F3C4 D607
F3C6 FE0A
F3C8 DB
F3C9 FE10
F3CB 3F
F3CC C9
F3CD 7C
F3CE CDBDF3
F3D1 7D
F3D2 CDBDF3
F3D5 C302F4
F3D8 F5
F3D9 1F
F3DA 1F
F3DB 1F
F3DC 1F
F3DD CDE1F3
F3E0 F1
F3E1 E60F

0762 SRL A
0763 INC A
0764 RET
0765
0766 GETHEX CONVERTS ASCII TO BINARY AND DOES
0767 HIGH LIMIT CHECKS TO LESS THAN 17 BITS.
0768 CARRY SET ON ILLEGAL CONVERSION RESULT
0769 TERMINATING CHARACTER RETURNS IN A.
0770 HL RETURNS WITH 16 BIT BINARY INTEGER
0771
0772 GETHEX: LD HL,0
0773 JR GNUM3-$
0774
0775 GNUM1: LD B,4
0776 GNUM2: ADD HL,HL
0777 RET C
0778 DJNZ GNUM2-$
0779 E,A
0780 D,0
0781 ADD HL,DE
0782 RET C
0783 GNUM3: LD A, (IY+O)
0784 INC IY
0785 LD C,A
0786 CALL ASCHEX
0787 JR NC,GNUM1-$
0788 LD A,C
0789 OR A
0790 RET
0791
0792
0793 ASCHEX: SUB '0'
0794 RET C
0795 CP 10
0796 CCF
0797 RET NC
0798 SUB 7
0799 CP 10
0800 RET C
0801 CP 16
0802 CCF
0803 RET
0804
0805
0806
0807 PUT4HS: LD "A,H
0808 CALL PUT2HX
0809 LD A,L
0810 PUT2HS: CALL PUT2HX
0811 JP SPACE
0812
0813
0814 PUT2HX: PUSH AF
0815 RRA
0816 RRA
0817 RRA
0818 RRA
0819 CALL PUTNIB
0820 POP AF
0821 PUTNIB: AND $00001111
F3E3 C690 0822 ADD A, 90H
F3E5 27 0823 DAA
F3E6 CE40 0824 ADC A, 40H
F3EB 27 0825 DAA
F3E9 C315F4 0826 JP OUTPUT

0827 ;
0828 ;
0829 ; PMSG PRINTS THE STRING OF ASCII CHARACTERS
0830 ; POINTED TO BY THE RELATIVE ADDRESS IN DE
0831 ; UNTIL AN EOT IS ENCOUNTERED IN THE STRING.
0832 ;

>0004 0833 EDT EQU 04H
>000D 0834 CR EQU 0DH
>000A 0835 LF EQU 0AH
0836 ;
0837

F3EC E3 0838 PNEXT: EX (SP), HL
F3ED CDF2F3 0839 CALL PMSG
F3F0 E3 0840 EX (SP), HL
F3F1 C9 0841 RET
F3F2 7E 0842 ;
F3F3 23 0843 PMSG: LD A,(HL)
F3F4 FE04 0844 INC HL
F3F6 C8 0845 CP EOT
F3F7 CD15F4 0846 RET Z
F3FA 1BF6 0847 CALL OUTPUT
F3FB CDECFF3 0848 JR PMSG-$
F3FF 0D0A04 0849 ; CRLFS OUTPUTS A RETURN-LINEFEED-SPACE
F402 3E20 0850 TO THE CONSOLE DEVICE
F404 C315F4 0851 ;
F407 CD09F0 0852 CRLFS: CALL FNEXT
F40A F5 0853 ; ECHO INPUTS ONE CHARACTER FROM THE CONSOLE
F40B CD0CF0 0854 DEVICE, PRINTS IT ON THE CONSOLE OUTPUT AND
F40E F1 0855 THEN RETURNS IT IN REGISTER A WITH BIT 7 RESET
F40F FE5B 0856 SPACE: LD A,
F411 DB 0857 JP OUTPUT
F412 D620 0858 ;
F414 C9 0859 ; ECHO INPUTS ONE CHARACTER FROM THE CONSOLE
F415 CD0CF0 0860 ; DEVICE, PRINTS IT ON THE CONSOLE OUTPUT AND
F418 CD06F0 0861 THEN RETURNS IT IN REGISTER A WITH BIT 7 RESET
F41B 280F 0862 ; ECHO INPUTS ONE CHARACTER FROM THE CONSOLE
F41D CD09F0 0863 ; DEVICE, PRINTS IT ON THE CONSOLE OUTPUT AND
F420 FE0D 0864 THEN RETURNS IT IN REGISTER A WITH BIT 7 RESET
F422 2805 0865 ; ECHO INPUTS ONE CHARACTER FROM THE CONSOLE
F424 CD09F0 0866 ; DEVICE, PRINTS IT ON THE CONSOLE OUTPUT AND
F427 1BF3 0867 THEN RETURNS IT IN REGISTER A WITH BIT 7 RESET
F429 3284FF 0870 ECHO: CALL CONIN ; INPUT A CHARACTER AND ECHO IT
F42C 3A84FF 0871 PUSH AF
F42F B7 0872 CALL CONOUT
F430 C9 0873 POP AF
F435 FE5B 0874 CP ‘Z’+1
F438 DB 0875 RET C
F43A D620 0876 SUB 32 ; CONVERT UPPER CASE TO LOWER
F43C C9 0877 RET
F43D 0878 ;
F43E 0879 ;
F440 0880 ;
F442 0881 OUTPUT: CALL CONOUT ; SEE IF CONSOLE INPUT PENDING
F445 0882 CALL CONST ; SEE IF <CR> WAS TYPED
F448 0883 JR 2,OUTP2-
F44B 0884 CALL CONIN
F44E 0885 CP CR
F451 0886 JR 2,OUTP1-
F454 0887 CALL CONIN ; WAIT FOR ANOTHER INPUT CHAR
F457 1BF3 0888 JR OUTP2-
F45A 0889 ; THEN RET TO CALLING ROUTINE
F45D 0890 OUTP1: LD (ESCFGLG),A ; SET ESC FLAG TO NON-ZERO VALUE
F45F 0891 OUTP2: LD A,(ESCFGLG)
F462 B7 0892 OR A ; RETURN CURRENT STATUS OF ESC
F465 C9 0893 RET
F466 0894 ;
F467 0895 ;
F468 0896 ;
F469 0897 ; INCLUDE INTSRV.ASM
**INTERRUPT SERVICE ROUTINES FOR KEYBOARD**

**INPUT AND REAL-TIME CLOCK FUNCTIONS**

3-Aug-80

```
F431 3A30FF 0909 KBDST:  LD  A, (FIFCNT) ; GET INPUT FIFO BYTECOUNT
F434  B7  0910 OR  A ; TEST IF EQUAL ZERO
F435  C8  0911 RET  Z ; EXIT WITH A=0 IF QUEUE EMPTY
F436  3EFF  0912 LD  A,255 ; ELSE A=255 INDICATES DATA RDY
F438  C9  0913 RET
F439  CD31F4  0914
F43C  2BFB  0915 CALL  KBDST
F43E  E5  0916 JR  Z,KBDIN-; LOOP UNTIL KEYBOARD INPUT RDY
F43F  CD6DF4  0917 PUSH  HL
F442  E1  0918 CALL  REMOVE ; GET CHARACTER FROM INPUT QUEUE
F443  C9  0919 POP  HL
F444  2133FF  0920 RET
F447  BE  0921 CP  (HL) ; TEST IF A=SHIFT LOCK CHARACTER
F448  23  0922 INC  HL ; THEN POINT TO LOCK FLAG
F449  2002  0923 JR  NZ,STASH2-- ; JUMP IF NOT SHIFT CHARACTER
F44B  34  0924 INC  (HL) ; ELSE COMPLIMENT THE SHIFT LOCK
F44C  C9  0925 RET ; AND EXIT NOW
F44D  CB46  0926
F44F  280A  0927 STASH2:  BIT 0, (HL) ; TEST THE SHIFT LOCK FLAG
F451  FE40  0928 JR  Z,STASH3-- ; JUMP IF SHIFT LOCK NOT SET
F453  3806  0929 CP  40H ; ELSE CHECK FOR SHIFTABLE CHAR
F455  FE7F  0930 JR  C,STASH3-- ; AND JUMP IF NOT = OR GREATER
F457  3002  0931 CP  7FH ; THAN '3' AND LESS THAN RUBOUT
F459  EE20  0932 JR  NC,STASH3--
F45B  4F  0933 XOR  00100000B ; ELSE TOGGLE BIT 5 OF THE CHAR
F45C  2130FF  0934 LD  C,A
F45F  7E  0935 LD  A, (HL)
F460  3C  0936 INC  A
F461  FE10  0937 CP  16
F463  D0  0938 RET  NC ; EXIT NOW IF FIFO IS FULL
F464  77  0947 LD  (HL),A ; ELSE INCREMENT FIFO COUNT
F465  2131FF  0948 LD  HL,FIFIN ; POINT HL TO FIFO INPUT OFFSET
F468  CD74F4  0949 CALL  INDEX
F46B  71  0950 LD  (HL),C ; STORE CHARACTER IN FIFO @ HL
F46C  C9  0951 RET
```
F46D 2130FF 0957 REMOVE LD HL,FIFCNT
F470 35 0958 DEC (HL)
F471 2132FF 0959 LD HL,FIFOUT ;POINT HL TO FIFO OUTPUT OFFSET
F474 7E 0960 INDEX: LD A,(HL)
F475 3C 0961 INC A
F476 E60F 0962 AND 00001111B ;INCREMENT FIFO POINTER
F478 77 0963 LD (HL),A ;MODULO 16 AND REPLACE
F479 2120FF 0964 LD HL,FIFO
F47C B5 0965 ADD A,L ;INDEX INTO FIFO BY OFFSET IN A
F47D 6F 0966 LD L,A
F47E 7E 0967 LD A,(HL)
F47F C9 0968 RET

;SOFTWARE DISK MOTOR TURN-OFF TIMER ROUTINE
F480 216FF 0973 DSKTMR: LD HL,MOTOR ;DECREMENT DISK TURN-OFF TIMER
F483 35 0974 DEC (HL)
F484 C0 0975 RET NZ ;EXIT IF NOT TIMED OUT YET
F485 DB1C 0976 IN A,(BITDAT)
F487 F644 0977 OR 01000100B ;DISABLE ALL DRIVE SELECTS AND
F489 D31C 0978 OUT (BITDAT),A ;TURN OFF THE SPINDLE MOTORS
F48B C9 0979 RET

;-- INTERRUPT SERVICE ROUTINE FOR PARALLEL KEYBOARD --
F4BC ED735FF 0984 KEYSRV: LD (SSAVE),SP ;SAVE USR STACK POINT AND
F490 3157FF 0985 LD SP,TMPSTK+32;SWITCH TO LOCAL STACK
F493 E5 0986 PUSH HL
F494 D5 0987 PUSH DE
F495 C5 0988 PUSH BC
F496 F5 0989 PUSH AF ;SAVE MACHINE STATE
F497 DB1E 0990 IN A,(KBDDAT) ;READ KEYBOARD INPUT PORT
F499 2F 0991 CPL
F49A 2A59FF 0992 LD HL,(PINVEC);GET KBD INTERRUPT RTN VECTOR
F49D 1822 0993 JR DSPTCH-# ;AND JUMP TO DISPATCH POINT

;-- INTERRUPT SERVICE ROUTINE FOR ONE SECOND TIMER --
F49F ED7335FF 0999 TIMER: LD (SSAVE),SP ;SAVE USR STACK POINTER AND
F4A3 3157FF 1000 LD SP,TMPSTK+32 ;SWITCH TO LOCAL STACK
F4A6 E5 1001 PUSH HL
F4A7 D5 1002 PUSH DE
F4A8 C5 1003 PUSH BC
F4A9 F5 1004 PUSH AF ;SAVE MACHINE STATE
F4AA 2A57FF 1005 LD HL,(TIKVEC);GET CLOCK INTERRUPT RTN VECTOR
F4AD 1812 1006 JR DSPTCH-# ;AND JUMP TO DISPATCH POINT

;-- SERIAL INPUT INTERRUPT SERVICE ROUTINE FOR SIO --
F4AF ED7335FF 1012 SIOINT: LD (SSAVE),SP ;SAVE USER STACK POINTER AND
F4B3 3157FF 1013 LD SP,TMPSTK+32 ;SWITCH TO LOCAL STACK
F4B6 E5 1014 PUSH HL
F4B7 D5 1015 PUSH DE
F4B8 C5 1016 PUSH BC
F4B9 F5 1017 PUSH AF ;SAVE MACHINE STATE
F4BA DB05 1018 IN A,(SIODPB) ;READ SIO DATA INPUT PORT
F4BC E67F 1019 AND 01111111B
F4BE 2A5BFF 1020 LD HL,(SINVEC);GET SERIAL INPUT RTN VECTOR
DSPTCH: CALL CALLHL ; CALL SUBROUTINE ADDRESSED BY H
    POP AF
    POP BC
    POP DE
    POP HL
    LD SP,(SPSAVE) ; RE-ENABLE INTERRUPTS & RETURN
    EI
    RETI

-- RX ERROR INTERRUPT SERVICE ROUTINE FOR SIO --
ARIVE HERE IF RECEIVE INTERRUPT FROM FRAMING, OVERRUN
AND PARITY ERRORS. (PARITY CAN BE DISABLED)

SIOERR: LD (SPSAVE),SP ; SAVE USER STACK POINTER AND
    SP,TMPSTK+32 ; SWITCH TO LOCAL STACK
    PUSH AF
    CALL SIOIN2 ; CLEAR BAD CHARACTER FROM SIO
    LD A,'G'-64
    CALL SIOXMT ; OUTPUT A CTL-G AS A WARNING
    POP AF
    LD SP,(SPSAVE)
    EI
    RETI

CALLHL: JP (HL)

POLLED MODE I/O ROUTINES FOR SIO CHANEL B

SIOST: IN A,(SIOCPB) ; GET SIO STATUS REGISTER
    AND 00000001B
    RET Z ; ACC=0 IF NO DATA AVAILABLE
    LD A,255
    RET

SIDIN: CALL SIOST ; TEST CONSOLE STATUS
    Z,SIDIN-# ; LOOP UNTIL DATA IS RECEIVED
    LD A,00110000B ; RESET STATUS BITS IN SIO FO
    OUT (SIOCPB),A ; PARITY/OVER/RUN/FRAMING ERRORS.
    IN A,(SIODPB) ; THEN GET THE INPUT CHARACTER
    AND 01111111B
    RET

SIDOUT: CP '' ; TEST FOR CONTROL CHARACTERS
    JR NC,SIOXMT-# ; JUMP IF PRINTABLE CHARACTER
    CALL SIOXMT ; ELSE SEND CONTROL CHARACTER
    JR (NULLS) ; AND THEN SEND NULLS AS PADDLING
    INC A ; GET NULL PAD COUNT AND FIX SO
    JR PAD1-# ; THAT COUNT=0 SENDS NO NULLS
    JR PAD6-# ; THAT COUNT>0 SENDS NULLS

PAD: PUSH AF
    XOR A
    CALL SIOXMT ; OUTPUT A NULL TO THE SIO
    POP AF
    JR A,DEC
    CALL NR,PAD-# ; LOOP SENDING NULLS TO SIO

SIOXMT: PUSH AF
    IN A,(SIOCPB)
INCLUDE CRTOUT.ASM

**MEMORY-MAPPED CRT OUTPUT DRIVER**
* Russell Smith 18-August-1980

CRTBASE EQU CRTMEM.SHAR.B ;START PAGE# OF 3K CRT SPACE
CRTPTE EQU CRTMEM+3072.SHAR.B ;END PAGE# OF CRT SPACE

CRTOUT: PUSH HL
PUSH DE
PUSH BC
RES 7, A
LD C, A
DI ;KEEP WOLVES AWAY FOR A WHILE

LD (SPSAVE), SP
LD SP, TMPSTK+32 ;POINT SP TO TOP LOCAL STACK
IN A, (BITDAT)
SET 7, A ;SELECT ROM/CRT MEMORY BANK
OUT (BITDAT), A

FIRST REMOVE THE OLD CURSOR CHARACTER FROM THE SCREEN

LD HL, CHRSAV ;GET CHAR OVERLAYED BY CURSOR
LD B, (HL)
LD HL, (CURSOR); LOAD HL WITH CURSOR POINTER
LD A, H
AND 00001111B ;INSURANCE THAT HL CAN'T
OR CRTBASE ;EVER POINT OUTSIDE CRT MEMORY
LD H, A
LD (HL), B ;RMV CURSOR BY RESTORING CHAR

PROCESS CHARACTER PASSED IN C

CALL OUTCH

NOW STORE A NEW CURSOR CHARACTER AT THE CURSOR LOCATION

LD A, (HL) ;GET CHAR AT NEW CURSOR LOCAT.
LD (CHRSAV), A ;SAVE FOR NEXT TIME 'CRTOUT' IS CALLED

CP ' ' ;TEST IF CHARACTER IS A SPACE
SET 7, A ;THEN TURN ON BIT 7 TO ENABLE BLINK
JR NZ, CRT2-6 ;JUMP IF CHARACTER IS NON-BLANK
LD A, (CSRCHR) ;ELSE GET CHAR USED FOR CURSOR
LD (HL), A ;STORE CHAR IN A AS CURSOR MARK
LD (CURSOR), HL;SAVE HL AS CURSOR POINTER
LD SP, (SPSAVE)
IN A, (BITDAT)
F55C CBBF 1151 RES 7,A ;SWITCH BACK LOWER 16K OF RAM
F55E D31C 1152 OUT (BITDAT),A ;INTERRUPTS ARE SAFE AGAIN
F560 FB 1153 EI
F561 C1 1154 POP BC
F562 D1 1155 POP DE
F563 E1 1156 POP HL
F564 C9 1157 RET
F565 117BFF 1158 OUTCH: LD DE,LEADIN
F566 1A 1161 LD A,(DE) ;GET LEAD-IN SEQUENCE STATE
F569 B7 1162 DR A
F56A C270F6 1163 JP NZ,MULTI ;JUMP IF IN A LEAD-IN SEQUENCE
F56D 79 1164 LD A,C ;ELSE PROCESS CHARACTER IN C
F56E FE20 1165 CP ,'
F570 380F 1166 JR C,CTRL-$ ;JUMP IF A CONTROL CHARACTER
F572 71 1167 DISPLA: LD (HL),C ;ELSE STORE DISPLAYABLE CHAR
F573 23 1168 INC HL ;AND ADV POINTER TO NEXT COLUMN
F574 7D 1169 LD A,L
F575 E67F 1170 AND 01111111B ;EXTRACT COLUMN# FROM HL
F577 FE50 1171 CP B0
F579 D8 1172 RET C ;EXIT IF NOT PAST COLUMN 79
F57A CDE7F5 1173 CALL RETURN ;ELSE DON'T AUTOMATIC <CR>
F57D CD42F6 1174 CALL LFEED ;AND LINEFEED
F580 C9 1175 RET
F581 E5 1176 CRNL: PUSH HL
F582 21BFF5 1177 LD HL,CTLTAB ;SEARCH FOR CONTROL CHARACTER
F585 010D00 1178 LD BC,CTSIZ/3;HANDLING SUBROUTINE IN TABLE
F588 CD60F3 1179 CALL SEARCH
F58B E1 1180 POP HL
F58C C0 1181 RET NZ ;EXIT IF NOT IMPLEMENTED
F58D C5 1182 PUSH BC
F58E C9 1183 RET ;DO SNEAKY JUMP TO PRESERVE REGISTERS
F58F 1F 1184 CTLLTAB: DEFB ',-'-64
F590 1E 1185 DEFB ',-'-64
F591 1B 1186 DEFB ',-'-64
F592 1A 1187 DEFB ',-'-64
F593 18 1188 DEFB ',-'-64
F594 11 1189 DEFB ',-'-64
F595 0D 1190 DEFB ',-'-64
F596 0C 1191 DEFB ',-'-64
F597 0B 1192 DEFB ',-'-64
F598 0A 1193 DEFB ',-'-64
F599 09 1194 DEFB ',-'-64
F59A 08 1195 DEFB ',-'-64
F59B 07 1196 DEFB ',-'-64
F59C DCF5 1201 DEFB ',-'-64
F59D BCF5 1202 DEFB ',-'-64
F5A0 CCF5 1203 DEFB ',-'-64
F5A2 42F6 1204 DEFB ',-'-64
F5A4 2CF6 1205 DEFB ',-'-64
F5A6 C4F5 1206 DEFB ',-'-64
F5A9 DEFB ',-'-64
F5AB DEFB ',-'-64
F5AC DEFB ',-'-64
F5AD DEFB ',-'-64
F5AE DEFB ',-'-64
F5AF DEFB ',-'-64
F5B0 DEFB ',-'-64
F5B1 DEFB ',-'-64
F5B2 DEFB ',-'-64
F5B3 DEFB ',-'-64
F5B4 DEFB ',-'-64
F5B5 DEFB ',-'-64
F5B6 DEFB ',-'-64
F5B7 DEFB ',-'-64
F5B8 DEFB ',-'-64
F5B9 DEFB ',-'-64
F5BA DEFB ',-'-64
F5BB DEFB ',-'-64
F5BC DEFB ',-'-64
F5BD DEFB ',-'-64
F5BE DEFB ',-'-64
F5BF DEFB ',-'-64
F5C0 DEFB ',-'-64
F5C1 DEFB ',-'-64
F5C2 DEFB ',-'-64
F5C3 DEFB ',-'-64
F5C4 DEFB ',-'-64
F5C5 DEFB ',-'-64
F5C6 DEFB ',-'-64
F5C7 DEFB ',-'-64
F5C8 DEFB ',-'-64
F5C9 DEFB ',-'-64
F5CA DEFB ',-'-64
F5CB DEFB ',-'-64
F5CC DEFB ',-'-64
F5CD DEFB ',-'-64
F5CE DEFB ',-'-64
F5CF DEFB ',-'-64
F5D0 DEFB ',-'-64
F5D1 DEFB ',-'-64
F5D2 DEFB ',-'-64
F5D3 DEFB ',-'-64
F5D4 DEFB ',-'-64
F5D5 DEFB ',-'-64
F5D6 DEFB ',-'-64
F5D7 DEFB ',-'-64
F5D8 DEFB ',-'-64
F5D9 DEFB ',-'-64
F5DA DEFB ',-'-64
F5DB DEFB ',-'-64
F5DC DEFB ',-'-64
F5DD DEFB ',-'-64
F5DE DEFB ',-'-64
F5DF DEFB ',-'-64
F5E0 DEFB ',-'-64
F5E1 DEFB ',-'-64
F5E2 DEFB ',-'-64
F5E3 DEFB ',-'-64
F5E4 DEFB ',-'-64
F5E5 DEFB ',-'-64
F5E6 DEFB ',-'-64
F5E7 DEFB ',-'-64
F5E8 DEFB ',-'-64
F5E9 DEFB ',-'-64
F5EA DEFB ',-'-64
F5EB DEFB ',-'-64
F5EC DEFB ',-'-64
F5ED DEFB ',-'-64
F5EE DEFB ',-'-64
F5EF DEFB ',-'-64
F5F0 DEFB ',-'-64
F5F1 DEFB ',-'-64
F5F2 DEFB ',-'-64
F5F3 DEFB ',-'-64
F5F4 DEFB ',-'-64
F5F5 DEFB ',-'-64
F5F6 DEFB ',-'-64
F5F7 DEFB ',-'-64
F5F8 DEFB ',-'-64
F5F9 DEFB ',-'-64
F5FA DEFB ',-'-64
F5FB DEFB ',-'-64
F5FC DEFB ',-'-64
F5FD DEFB ',-'-64
F5FE DEFB ',-'-64
F5FF DEFB ',-'-64

;CTL-G IS THE BELL
;CTL-H IS CURSOR LEFT
;CTL-I IS TAB
;CTL-J IS CURSOR DOWN
;CTL-K IS CURSOR UP
;CTL-L IS CURSOR RIGHT
DEFW 1209 RETURN ; CTL-M IS <CR>
DEFW 1210 CLEOS ; CTL-Q CLEAR TO END-OF-SCREEN
DEFW 1211 CLEOL ; CTL-X CLEAR TO END-OF-LINE
DEFW 1212 CLRSCL ; CTL-Z IS CLEAR SCREEN
DEFW 1213 ESCAPE ; CTL-I IS ESCAPE
DEFW 1214 HOMEUP ; CTL-^ IS HOME UP
DEFW 1215 STUFF ; CTL-- IS DISPLAY CONTROL CHARS

>0027
DEFW 1217 CTLSIZ EQU $-CTLTAB

DEFW 1218 ;
DEFW 1219 ;

DEFW 1220 ESCAPE; LD A,1
DEFW 1221 (DE),A ; SET LEAD-IN SEQUENCE STATE
DEFW 1222 RET ; FOR XY CURSOR POSITIONING MODE

DEFW 1223 ;
DEFW 1224 ;

DEFW 1225 STUFF; LD A,4
DEFW 1226 (DE),A ; SET LEAD-IN SEQUENCE STATE
DEFW 1227 RET ; FOR CONTROL CHAR OUTPUT MODE

DEFW 1228 ;
DEFW 1229 ;

DEFW 1230 BAKSPC LD A,L ; CHECK FOR LEFT MARGIN
DEFW 1231 AND 0111111B
DEFW 1232 RET Z ; ABORT IF IN LEFTMOST COLUMN
DEFW 1233 DEC HL ; BACK UP CURSOR POINTER
DEFW 1234 RET

DEFW 1235 ;
DEFW 1236 ;

DEFW 1237 FORSPC LD A,L ; CHECK FOR RIGHTMOST COLUMN
DEFW 1238 AND 0111111B
DEFW 1239 CP 79
DEFW 1240 RET NC ; DO NOTHING IF ALREADY THERE
DEFW 1241 INC HL ; ELSE ADVANCE CURSOR POINTER
DEFW 1242 RET

DEFW 1243 ;
DEFW 1244 ;

DEFW 1245 TAB: LD DE,B ; TABBES ARE EVERY 8 COLUMNS
DEFW 1246 LD A,L ; GET COLUMN COMPONENT OF
DEFW 1247 AND 01111000B ; PREVIOUS TAB POSITION
DEFW 1248 ADD A,E
DEFW 1249 CP 80 ; EXIT IF NEXT TAB COLUMN WOULD
DEFW 1250 RET NC ; BE PAST THE RIGHT MARGIN
DEFW 1251 LD A,L ; ELSE INCREMENT THE CURSOR
DEFW 1252 AND 1111100B
DEFW 1253 LD L,A ; POINTER FOR REAL
DEFW 1254 ADD HL,DE
DEFW 1255 RET

DEFW 1256 ;
DEFW 1257 ;

DEFW 1258 BELL: IN A,(BITDAT) ; TOGGLE BIT 5 OF SYSTEM PIO TO
DEFW 1259 SET S,A ; TRIGGER BELL HARDWARE TO SOUND
DEFW 1260 OUT (BITDAT),A
DEFW 1261 RES S,A
DEFW 1262 (BITDAT),A
DEFW 1263 RET

DEFW 1264 ;
DEFW 1265 ;

DEFW 1266 RETURN: LD A,L
DEFW 1267 AND 10000000B
DEFW 1268 LD L,A ; MOVE CURSOR POINTER BACK
DEFW 1269 RET ; TO START OF LINE
DEFW 1270 ;
DEFW 1271 ;

DEFW 1272 CLRSCN: LD HL,CRTMEM
F5EF E5 1274 PUSH HL
F5F0 110130 1275 LD DE,CRTMEM+1
F5F3 01000C 1276 LD BC,24*128
F5F6 3620 1277 LD (HL),'
F5F8 EDB0 1278 LDIR ;FILL CRT MEMORY WITH SPACES
F5FA E1 1279 POP HL ;POINT TO HOME CURSOR POSITION
F5FB 3E17 1280 LD A,23
F5FD 3277FF 1281 LD (BASE),A ;MAKE BASE LINE# BE 23 AND
F600 D314 1282 OUT (SCROLL),A ;STORE IN SCROLL REGISTER
F602 C9 1283 ;
F603 E5 1284 CLRELD: PUSH HL ;SAVE CURSOR POINTER
F604 7D 1285 LD A,L
F605 E67F 1286 AND 01111111B ;GET COLUMN# COMPONENT OF
F607 4F 1287 LD C,A ;CURSOR POINTER INTO C
F608 3E50 1288 LD A,80 ;CALCULATE HOW MANY CHARS
F60A 91 1289 SUB C ;REMAIN ON CURRENT LINE
F60B 47 1290 LD B,A
F60C CD66F6 1291 CALL CLR ;CLEAR REST OF LINE @ HL
F60F E1 1292 POP HL
F610 C9 1293 RET

F611 CD03F6 1294 CLREDS: CALL CLREOL ;CLEAR REMAINDER OF CURRENT ROW
F614 E5 1295 PUSH HL
F615 3A77FF 1296 LD A,(BASE)
F61B 4F 1297 LD C,A ;COPY BASE SCREEN ROW# TO C
F619 7D 1298 LD A,L
F61A 17 1299 RLA
F61B 7C 1300 LD A,H ;ROW# COMPONENT OF HL INTO A
F61C 17 1301 RLA
F61D E61F 1302 AND 00011111B ;SEE IF HL IS AT BOTTOM ROW
F61F B9 1303 CP C
F622 2808 1304 JR Z,CLRS2- ;AND LEAVE CLEAR LOOP IF SO
F625 CD60F6 1305 CALL CLRLIN ;AND FILL THAT LINE WITH SPACES
F626 18EF 1306 JR CLRS1- ;
F62A E1 1307 CLRS2: POP HL ;RESTORE ORIGINAL CURSOR POINTER
F62B C9 1308 RET

F62C 1180FF 1309 UPCSR: LD DE,-128 ;SUBTRACT 1 FROM ROW# COMPONENT
F62F 19 1310 ADD HL,DE ;OF CURSOR POINTER IN HL
F630 7C 1311 LD A,H
F631 FE30 1312 ADD CP CRTC BAS ;CHECK FOR UNDERFLOW OF POINTER
F633 D0 1313 RET NC
F634 263B 1314 LD H,CRTTOP-1 ;WRAP CURSOR AROUND MODULO 3K
F636 C9 1315 RET

F637 118000 1316 DNCSR: LD DE,128 ;ADD 1 TO ROW# COMPONENT
F63A 19 1317 ADD HL,DE ;OF CURSOR POINTER IN HL
F63B 7C 1318 LD A,H
F63C FE3C 1319 CP CRTC TOP ;CHECK FOR OVERFLOW OF POINTER
F63E D8 1320 RET C
F63F 2630 1321 LD H,CRTBASE ;RESET POINTER MODULO 128*24
F641 C9 1322 RET

F642 7D 1323 LFEED: LD A,L
F643 17 1324 RET
F644 7C 1325 LD A,H
F645  17  1338  RLA  ;EXTRACT ROW# COMPONENT OF HL
F646  E61F  1339  AND  00011111B ;COPY ROW# TO C FOR SCROLL TEST
F648  4F  1340  LD  C,A  ;MOVE CURSOR TO NEXT ROW DOWN
F649  CD37F6  1341  CALL  DNC5R  ;TEST IF CURSOR ON BOTTOM ROW
F64C  3A77FF  1342  LD  A,(BASE)  ;OF SCREEN BEFORE MOVING DOWN
F64F  B9  1343  CP  C  ;EXIT IF NOT AT BOTTOM
F650  C0  1344  RET  NZ  
F651  E5  1345  PUSH  HL  ;ELSE PREP TO SCROLL SCREEN UP
F652  CD60F6  1346  CALL  CLRLIN  ;FILL NEW BOTTOM LINE WITH SPACES
F655  29  1348  ADD  HL,HL  ;GET ROW# PART OF HL INTO A
F656  7C  1349  LD  A,H  00011111B ;STORE NEW BASE LINE#
F657  E61F  1350  AND  (BASE),A  ;SCROLL UP NEW BLANK BOTTOM LINE
F659  3277FF  1351  LD  (BASE),A  ;STORE ASCII SPACES AT ADDR
F65C  D314  1352  OUT  (SCROLL),A  ;IN HL
F65E  E1  1353  POP  HL
F65F  C9  1354  RET  
F660  7D  1355  ;
F661  E680  1357  CLRLIN;  LD  A,L  ;POINT HL TO 1ST COLUMN OF ROW
F663  6F  1358  AND  10000000B  ;STATE TO ZERO BEFORE GOING ON
F664  0650  1359  LD  L,A  ;DE,HL
F666  3620  1360  CLR;  LD  (HL),0  ;STORE ASCII SPACES AT ADDR
F668  23  1361  INC  HL  ;IN HL
F669  10FB  1362  INC  HL  ;AND INCREMENT HL
F66B  C9  1363  DJNZ  CLR-  ;REPEAT NUMBER OF TIMES IN B
F66C  OE20  1364  RET  
F66E  1817  1365  HOMEUP;  LD  C,' '  ;FAKE-OUT CURSOR ADDR ROUTINE
F66F  1817  1367  JR  SETROW-  ;TO DO HOMEUP ALMOST FOR FREE
F670  EB  1368  ;
F671  3600  1369  MULTI;  EX  DE,HL  ;UNCONDITIONALLY RESET LEAD-IN
F673  EB  1370  1372  LD  (HL),0  ;STATE TO ZERO BEFORE GOING ON
F674  FE01  1373  EX  1  ;DE,HL
F676  2008  1374  JR  NZ,M2ST-  ;GET SECOND CHAR OF SEQUENCE
F678  79  1375  SETXY;  LD  A,C  ' '  ;ABORT SEQUENCE IF NOT '='
F679  FE0D  1376  CP  C  ;MAKE LEADIN=2 NEXT TIME
F67B  C0  1377  RET  
F67C  3E02  1378  LD  A,2  
F67E  12  1379  LD  (DE),A  
F67F  C9  1380  RET  
F680  FE02  1381  M2ST;  CP  2  
F682  2019  1382  NZ,M3ST-  ;MAKE LEADIN=3 NEXT TIME
F684  3E03  1383  LD  A,3  
F686  D2  1384  LD  (DE),A  ;ARRIVE HERE ON THIRD CHAR
F687  3A77FF  1385  SETROW;  LD  A,(BASE)  ;OF ESC, '= ', ROW, COL SEQUENCE
F68A  B1  1386  ADD  A,C  '-1'  
F68B  D41F  1387  SUB  24  
F68D  D61B  1388  SETR2;  SUB  NC,SETR2-  ;VERIFY ROW# BETWEEN 0 AND 23
F68F  30FC  1389  JR  A,24  
F691  C61B  1390  ADD  H,A  ;MERGE IN MSB'S OF CRT MEMORY
F693  F660  1391  OR  CRTMEM.SHR.7  ;VERIFY ROW# BETWEEN 0 AND 23
F695  67  1392  LD  L,0  
F696  2E00  1393  LD  H,0  
F69B  CB3C  1395  SRL  H  
F69A  CB1D  1396  RR  L  
F69C  C9  1397  RET  
F69D  FE03  1398  M3ST;  CP  3  
F69F  200C  1400  JR  NZ,M4ST-  ;
F6A1 79 1402 SETCOL: LD A,C ;ARRIVE HERE ON FOURTH CHAR
F6A2 D620 1403 SUB ' ' ;OF ESC,'=', ROW, COL SEQUENCE
F6A4 D650 1404 SETC2: SUB B0
F6A6 30FC 1405 JR NC, SETC2-# ;MAKE SURE COL# BETWEEN 0 & 79
F6A8 C650 1406 ADD A, B0
F6AA B5 1407 OR L ;MERGE IN COL# WITH L
F6AB 6F 1408 LD L, A
F6AC C9 1409 RET
F6AD CD72F5 1410 M4TST: CALL DISPLA ;DISPLAY THE CONTROL CHAR
F6B0 C9 1411 RET ;PASSED IN C
F6B1 79 1412 INCLUDE DISKID.ASM
F6B2 FE04 1413 ;*****************************************************************
F6B4 D0 1414 ;* DISK INPUT/OUTPUT DRIVER SUBROUTINE PACKAGE *
F6B5 CDBB8F7 1415 ;* FOR WESTERN DIGITAL 1771 DISK CONTROLLER *
F6B6 DB1C 1416 ;*
F6B7 47 1417 ;bullet-proof error recovery added 12-APR-80 *
F6B9 E6FB 1418 ;*
F6BA B1 1419 ;*****************************************************************
F6BB C9 1420 ;EQUIVATS FOR DISK CONTROLLER PORTS AND COMMAND CODES
F6BD 79 1421 ;
F6BE D31C 1422 >0010 1430 STSREG EQU WD1771+0 ;STATUS REGISTER
F6C0 CDAEF7 1423 >0010 1431 CMDREG EQU WD1771+0 ;COMMAND REGISTER
F6C2 79 1424 >0011 1432 TRKREG EQU WD1771+1 ;TRACK REGISTER
F6C4 FE04 1425 >0012 1433 SECREG EQU WD1771+2 ;SECTOR REGISTER
F6C6 D0 1426 >0013 1434 DATREG EQU WD1771+3 ;DATA REGISTER
F6C8 B1 1427 ;
F6CA 79 1428 >0066 1444 NMIVEC EQU 0066H ;THE NON-MASKABLE INTERRUPT IS
F6CC D31C 1429 ;USED FOR DATA SYNC BETWEEN
F6CD CDAEF7 1430 >009B 1441 HLOAD EQU 00000100B ;RD/WRT HEAD LOAD ENABLE
F6CE 79 1431 >00C9 1443 RET EQU 0C9H ;SUBROUTINE RETURN INSTR OPCODE
F6D0 CDAEF7 1432 >0008 1444 NMIVEC EQU 0066H ;THE Z-80 AND 1771
F6D2 79 1433 >0000 1445 ;THE NON-MASKABLE INTERRUPT IS
F6D4 FE04 1434 >0004 1446 ;USED FOR DATA SYNC BETWEEN
F6D6 D0 1435 >0012 1447 ;
F6D8 B1 1436 >0011 1448 ;
F6DA C9 1437 >0066 1449 ;SELECT: LD A,C ;GET UNIT# PASSED IN C AND
F6DB 79 1450 CP 4 ;CHECK FOR MAXIMUM VALID#
F6DC D0 1451 RET NC ;ERROR IF NUMBER > 3
F6DD CDBB8F7 1452 CALL TURNON ;MAKE SURE DISKS ARE TURNED ON
F6DE DB1C 1453 IN A, (BITDAT)
F6E0 47 1454 LD B,A ;SAVE CURRENT DRIVE SELECT DATA
F6E1 CDBB8F7 1455 AND 1111100B ;MERGE IN NEW DRIVE UNIT# IN C
F6E2 DB1C 1456 OR C ;IN PLACE OF THE CURRENT ONE
F6E3 B1 1457 OUT (BITDAT), A ;TO SELECT THE NEW DISK DRIVE
F6E5 D31C 1458 CALL FORCE ;TEST NEW DRIVE’S READY STATUS
CALLReady; CLEAR THE DISK CONTROLLER
RET NZ; EXIT IF DRIVE NOT READY
BIT 6,A
RET NZ; EXIT IF DISK WRITE-PROTECTED
LD B,WRTCMD
JR RDWRT-$
CALLReady; CLEAR DISK CONTROLLER
RET NZ; EXIT IF DRIVE NOT READY
LD B,RDCMD
LD (IOPTR),HL; STORE DISK I/O DATA POINTER
LD HL,SECTOR
LD (HL),C; STORE SECTOR# FOR READ/WRITE
INC HL
LD (HL),B; SAVE READ/WRITE COMMAND BYTE
INC HL
LD (HL),2; SET DISK RE-TRY COUNT
DI; NO INTERRUPTS DURING DISK I/O
LD HL,NMIVEC; SAVE BYTE AT NMI VECTOR LOCAT
LD D,(HL); IN D FOR DURATION OF READ/WRT
LD (HL),RET; LOOP AND REPLACE IT WITH A RET
LD HL,RELEN
LD B,(HL) B=NUMBER OF BYTES/SECTOR
LD C,DATREG; C=1771 DATA REGISTER PORT#
LD HL,(IOPTR); HL=DISK R/W DATA POINTER
LD A,(SECTOR); GET SECTOR NUMBER
LD (SECREG),A; OUTPUT SECTOR# TO 1771
OUT (SECMD,A); OUTPUT SECTOR# TO 1771
CALL FORCE; ISSUE FORCE INTERRUPT COMMAND
BIT 5,A; TO TEST HEAD LOAD STATUS
LD (CMDTP),A; GET READ OR WRITE COMMAND BYTE
JR NZ,RW2-$; JUMP IF HEAD IS ALREADY LOADED
OR HLLOAD; ELSE MERGE IN HL BIT
LD CMDOUT; START 1771 DOING IT'S THING
JR NZ,WLOOP-$; AND JUMP TO THE CORRECT LOOP
BIT 5,A; TEST IF COMMAND IS A R OR W
JR NZ,WLOOP-$; AND JUMP TO THE CORRECT LOOP
JR NZ,WLOOP-$; AND JUMP TO THE CORRECT LOOP
OUTI
CALL BUSY
AND 10011100B; MASK OFF TO READY, NOT FOUND.CRC
JR RW3-$; AND LOST DATA STATUS BITS
HALT
NCNZ,RLOOP; LOOP UNTIL 1771 COMES UN-BUSY
OUTI
CALL BUSY
AND 10111100B; MASK OFF AS ABOVE + WRT FAULT
LD HL,NMIVEC
LD (HL),D; RESTORE BYTE @ NMI VECTOR
RET Z; RETURN IF NO DISK I/O ERRORS
LD HL,RETRY
DEC (HL)
JR NZ,RW4-$; EXECUTE COMMAND AGAIN IF NOT=0
OR A
JR NW3-$; ELSE RETURN 1771 ERROR STATUS
LD HL,TRACK
JR NW3-$; GET TRACK# FOR THIS OPERATION
LD C,(HL)
JR NW3-$; TRY TO RE-CALIBRATE THE HEAD
JR NW3-$; BEFORE READ OR WRITE AGAIN
F793 3A6AFF 1588 STEP: D A, (SPEED) ;GET STEP SPEED VARIABLE
F796 E603 1589 AND 0000001B
F798 B0 1590 OR B ;MERGE WTH SEEK/HOME COMND IN B
F799 CDA3F7 1591 CALL CMDOUT ;OUTPUT COMMAND AND DELAY
F79C DB10 1592 BUSY: IN A, (STSREG)
F79E CB47 1593 BIT 0, A ;TEST BUSY BIT FROM
F7A0 20FA 1594 JR NZ, BUSY-$ ; 1771 AND LOOP TILL=0
F7A2 C9 1595 RET

F7A3 D310 1599 CMDOUT: OUT (CMDREG), A ;OUTPUT A COMMAND TO THE 1771
F7A5 CDA8F7 1600 CALL PAUSE ;WASTE 44 MICROSECONDS
F7A8 E3 1601 PAUSE: EX (SP), HL
F7A9 E3 1602 EX (SP), HL
F7AA C9 1603 RET

F7AB CD88F7 1604 ;CALCULATE READY: CALL TURNON ;KEEP THOSE DISKS SPINING FOLKS
F7AE 3ED0 1605 ;LOAD A, FINCMD ;ISSUE FORCE INTERRUPT COMMAND
F7E0 CDA3F7 1606 CALL CMDOUT
F7E3 DB10 1607 IN A, (STSREG) ;READ STATUS REGISTER CONTENTS
F7E5 CB7F 1608 BIT 7, A ;TEST DRIVE NOT READY BIT
F7E7 C9 1609 RET

F7BB 3E1E 1610 TURNN: :LD A,30
F7BA 326CFF 1611 LD (MOTOR), A ;RE-LOAD MOTOR TURN-OFF TIMER
F7BD CDA8F7 1612 CALL PAUSE
F7C0 DB1C 1613 IN A, (BITDAT)
F7C2 CB57 1614 BIT 2, A ;TEST IF MOTORS HAVE STOPPED
F7C4 CB 1615 RET 2 ;AND EXIT IF STILL TURNED ON
F7C5 E4BB 1616 AND 10111011B ;ELSE RE-ENABLE DRIVE SELECTS
F7C7 D31C 1617 OUT (BITDAT), A ;AND ACTIVATE THE MOTOR RELAY
F7C9 C5 1618 PUSH BC
F7CA 0600 1620 LD B, 0 ;SET READY LOOP MAX TIMEOUT
F7CC CD0CF7 1621 TURN2: CALL WAIT ;WAIT 1/93 SECOND & TEST READY
F7CF 2B02 1622 JR 2, TURN3-$ ;EXIT LOOP IF DRIVE READY
F7D1 10F9 1623 LD B, 9
F7D3 0609 1624 TURN3: LD A, (BITDAT)
F7D5 CD0CF7 1625 TURN4: CALL WAIT ;GIVE ABT 1/10 SEC MORE DELAY
F7DB C1 1626 DJNZ TURN4-$
F7DB C9 1627 POP BC
F7DC 1628 RET

F7DC DB1B 1632 WAIT: IN A, (CTC3) ;GET CURRENT CTC3 COUNT VALUE
F7DE 4F 1633 LD C, A
F7DF DB1B 1634 WAIT2: IN A, (CTC3)
F7E1 B9 1635 CP C ;SEE IF CTC3 CHANGED BY 1 COUNT
F7E2 2BFB 1636 JR 2, WAIT2-$ ;AND LOOP UNTIL IT CHANGES
F7E4 18C8 1637 JR FORCE-$ ;THEN TEST DRIVE READY STATUS

F7E6 0000 1647 ROMEND: DEFW 0 ;TAIL OF FREE MEM LINKED LIST
1648 ;
>FF00 1649 ORG RAM
1650 INCLUDE MEMORY.ASM
1651 ;**********************************************************************
1652 ;
1653 ; STORAGE ALLOCATION FOR 256 BYTE SCRATCH RAM
1654 ;
1655 ;**********************************************************************
1656 ;
1657 ;
1658 >FF00
1659 VECTAB EQU $ ; INTERRUPT VECTOR TABLE STARTS
1660 >FF00
1661 SIDVEC: DEFS 16 ; SPACE FOR 8 VECTORS FOR SID
1662 >FF10
1663 CTCVEC: DEFS 8 ; SPACE FOR 4 VECTORS FOR CTC
1664 ;
1665 >FF1B
1666 SYSVEC: DEFS 4 ; SPACE FOR 2 VECTORS FOR SYSTEM PIO
1667 ;
1668 >FF1C
1669 GENVEC: DEFS 4 ; SPACE FOR 2 VECTORS FOR GENERAL PIO
1670 ;
1671 ; KEYBOARD DATA INPUT FIFO VARIABLES
1672 ;
1673 >FF20
1674 FIFO: DEFS 16 ; CONSOLE INPUT FIFO
1675 >FF30
1676 FIFCNT: DEFS 1 ; FIFO DATA COUNTER
1677 >FF31
1678 FIFIN: DEFS 1 ; FIFI INPUT POINTER
1679 ;
1680 >FF32
1681 FIFOUT: DEFS 1 ; FIFI OUTPUT POINTER
1682 ;
1683 >FF33
1684 LOCK: DEFS 2 ; SHIFT LOCK CHAR+FLAG BYTE
1685 ;
1686 ; STACK POINTER SAVE AND LOCAL STACK FOR INTERRUPT ROUTINES
1687 ;
1688 >FF35
1689 SPSAVE: DEFS 2 ; USER STACK POINTER SAVE AREA
1690 ;
1691 >FF37
1692 TMPSTK: DEFS 32 ; LOCAL STACK FOR INTERRUPTS
1693 ;
1694 ; 'SOFTWARE' VECTORS FOR INTERRUPT SERVICE ROUTINES
1695 ;
1696 >FF57
1697 TIKVEC: DEFS 2 ; 1 SEC INTERRUPT ROUTINE VECTOR
1698 >FF59
1699 PINVEC: DEFS 2 ; PARALLEL CONSOLE INPUT VECTOR
1700 ;
1701 >FF5B
1702 SINVEC: DEFS 2 ; SERIAL CONSOLE INPUT VECTOR
1703 ;
1704 ; CLOCK-TIMER INTERRUPT VARIABLES
1705 ;
1706 >FF5D
1707 TIKCNT: DEFS 2 ; BINARY CLOCK TICK COUNTER
1708 >FF5F
1709 DAY: DEFS 1 ; CALENDAR DAY
1710 ;
1711 >FF60
1712 MONTH: DEFS 1 ; MONTH
1713 >FF61
1714 YEAR: DEFS 1 ; YEAR
1715 ;
1716 >FF62
1717 HRB: DEFS 1 ; CLOCK HOURS REGISTER
1718 ;
1719 >FF63
1720 MINS: DEFS 1 ; MINUTES REGISTER
1721 ;
1722 >FF64
1723 SECS: DEFS 1 ; SECONDS REGISTER
1724 ;
1725 ; DISK I/O DRIVER VARIABLES
1726 ;
1727 >FF65
1728 UNIT: DEFS 1 ; CURRENTLY SELECTED DISK#
1729 ;
1730 >FF66
1731 TRKTAB: DEFS 4 ; 4 DRIVE HEAD POSITION TABLE
1732 ;
1733 >FF6A
1734 SPEED: DEFS 1 ; SEEK SPEED FOR 1771 COMMANDS
1735 ;
1736 >FF6B
1737 RECLEN: DEFS 1 ; SECTOR RECORD LENGTH VARIABLE
1738 ;
1739 >FF6C
1740 MOTOR: DEFS 1 ; DRIVE MOTOR TURN-OFF TIMER
1741 ;
1742 >FF6D
1743 TRACK: DEFS 1
1744 ;
1745 >FF6E
1746 SECTOR: DEFS 1
CMDTY  DEFS  1  ;COMMAND BYTE FOR READS/WRITES
RETRY  DEFS  1  ;DISK OPERATION RE-TRY COUNT
IOPTR  DEFS  2  ;DISK I/O BUFFER POINTER
;CRT OUTPUT DRIVER VARIABLES
CURSOR  DEFS  2  ;CURSOR POINTER
CHRSAV  DEFS  1  ;CHAR OVERLAYERED BY CURSOR
CSRCHR  DEFS  1  ;CHAR USED FOR A CURSOR
BASE   DEFS  1  ;CURRENT CONTENTS OF SCROLL REGISTER
LEADIN  DEFS  1  ;STATE OF LEAD-IN SEQUENCE HANDLER
;NULL PAD COUNT FOR SERIAL OUTPUT DELAY
NULLS  DEFS  1  ;# OF NULLS SENT AFTER CONTROL CHAR.
;LISTHEAD POINTER FOR DYNAMIC MEMORY ALLOCATION SCHEME
FREPTR  DEFS  2
;CONSOLE MONITOR PROGRAM VARIABLES
PARAM1  DEFS  2  ;STORAGE FOR NUMBERS READ
PARAM2  DEFS  2  ;FROM LINE INPUT BUFFER
PARAM3  DEFS  2  ;BY 'PARAMS' SUBROUTINE
PARAM4  DEFS  2
ESCFLG  DEFS  1  ;CONSOLE ESCAPE FLAG
COFLAG  DEFS  1  ;CONSOLE OUTPUT TOGGLE
LAST    DEFS  2  ;LAST ADDRESS USED BY 'MEMDMP'
LINBUF  DEFS  64  ;CONSOLE LINE INPUT BUFFER
;END
ERRORS=0000
PFM USER'S MANUAL

The "VERIFY" command has been deleted in order to add the new S "SWITCH" command to the PFM monitor.
The S command is used to switch console output from the "signed on" device to the other output. If you have signed on to the Serial device as the console I/O (i.e. the first Return after a reset was typed on the Serial terminal keyboard) and you wish to redirect the output to the BigBoard video, simply type "S" followed by a return.
If, however, you signed on using the ASCII keyboard and video output combination, and you want to redirect the system output to a 300 Baud Serial device, simply type "S" with a Return then "S" with a Return a second time.
If the default value of 300 baud on the Serial device is not acceptable, then use the following procedure to define a new baud rate. Make a note of the desired baud rate, and refer to the Hex to baud conversion table on page 8 of the Theory of Operation. Select the corresponding Hex value of the desired baud rate. Refering to the PFM user's manual, output this hex value to Port OC Hex using the "O" command. Type O OC,<DATA>. The new selected baud rate is valid until a system Reset, a CP/M boot, or another baud rate is selected.
Selecting a new baud rate MUST be done BEFORE using the "S" command. It is mandatory that the baud rates of the serial device and the Big Board serial channel B must be set equal. Incompatible baud rates can cause unpredictable system operation!!!

ASCII KEYBOARD

Bit 8 (KB7) of the keyboard MUST be grounded.
The component legend on the PC board is incorrect at connector J2. The "14" should read "2", the "Keyboard Connector Pin Assignments" in the Theory of Operation is CORRECT as shown.

CTC OPTION

To utilize the CTC function of auto timeout of the disk drives, jumper pins 3 and 4, 7 and 8 on JB2.

SERIAL I/O OPTION

To connect a serial terminal to Serial Channel B, use the "M" jumpers on JB5 as shown in the "Serial I/O Strapping Options" in the Theory of Operation.
Under the construction section for this option we neglected to instruct that an 18 pin socket be soldered at U107 and a 5.0688 mhz crystal be soldered at Y3. Also the 8116 baud rate generator must be installed at U107.

VIDEO OUTPUT

Some brands of video monitors may require adjustment of the Vertical Height and Horizontal Width controls for proper size.

KEY POINTS TO REMEMBER...
The Big Board clears the screen and waits for a Return before PFM signs on.
You cannot Boot CP/M or read a diskette with the "R" command if you do not have a drive connected.

DO NOT test memory above EFFF Hex or you will KILL PFM.

If your system starts acting strange, immediately recheck the power supply voltages.

A cap marked 101J is 100 pf while a "103" is a .01 mfd. Also an NEC D780C is a Z-80 CPU chip.

Any factory installed components or modifications should be left alone!

The MC1488 and MC1489 line drivers NORMALLY run warm.

If you plug any cable in backwards, the chances are something will be blown!

Poor soldering will potentially cause MORE problems than anything else.

Software on cassette cannot be loaded into the Bis Board.

IMPORTANT!

When using the Bis Board with only a partial compliment of I/O options, one or more jumpers must be added to complete the priority chain. The priority chain goes from the Serial I/O to Keyboard PIO to the Optional Parallel I/O (PIO) to the Real Time Clock (CTC).

If you are running your Bis Board with Basic I/O (no options) then a jumper is required between pins 6 and 7 on U113.

If you add the Serial Option (SIO) later, then the jumper on U113 must be removed.

If the Serial Option is on board, and no optional Parallel Option has been added, then the addition of the CTC requires a jumper at U89 pin 22 to pin 24.

If the CTC only is added to the Basic I/O configuration, then BOTH of the above mentioned jumpers are required.

If the Parallel Option only is added to Basic I/O then the jumper at U113 pins 6,7 is needed.

A Bis Board with ALL I/O options added REQUIRES NO JUMPERS.

U52 IS A 14 PIN DEVICE!
THE SILK SCREEN LEGEND IS NOT CORRECT.
The Model 753 Keyboard was designed as a physical and electrical equivalent to the ASR-33-style Keyboard. Since most modern hardware supports both upper-and-lower-case alphanumeric characters, the 753 outputs both upper-and-lower-case. Most of the Model 33 TTY shifted and control parity codes are also available. Provisions are made for upper-case only operation, selectable parity, and either positive or negative logic data and strobe output.

1. It is possible to use the Model 753 Keyboard with nearly every microcomputer input board, video board, etc. on the market. Some minor hardwiring may be required in certain cases. Since it is impossible to give step-by-step instructions for wiring the keyboard interface with all possible combinations of hardware, we'll instead try to describe the basic procedure used.
   A. The 753 is capable of driving one standard TTL unit load. You must supply appropriate buffers if your interface applies a greater load to data and strobe lines. The keyboard will drive 5-6' of cable without buffering.
   B. Both +5 VDC and -12 VDC must be supplied to the keyboard. (Optional DC-DC convertor available). Both supply currents are 20 MA, max.
   C. Designers of the 753's encoder chip chose to allow the data to fluctuate between key depressions. Your interface must ignore all data unless a strobe signal is present.
   D. The MOS-LSI encoder is highly static-sensitive. Handle it with care! Replacement chips are $7.50 each, and our flat "fix-it-fee" is $15.00.

2. Interconnecting the Keyboard. The 753 is designed to work into an 8 bit parallel data input port. Study the interface schematic to determine the proper data polarity (pos. true or neg. true), method of sensing the strobe signal, and which data input is LSB, and MSB.

3. Your interface cable now must connect the appropriate data inputs of your interface to the 756 card-edge connector. Note that there are 2 bit 6 signals, one for UC/LC, and one for UC-only. Connect the input directly, or through an SPDT Switch as shown in fig. 1 for selectable "Alpha-Lock" operation. Use of parity is at your option, as many devices do not support parity checking.

4. Connect the power supplies, and ground as shown. You must jumper the data-strobe invert pin on the connector--either to +5 for negative logic output, or ground for positive logic. Parity select must be jumpered if you are using parity.
   a) The keyboard should now be operational. Double-check all wiring with a VOM before applying power! Insert the encoder only with power off, and with great care!
   b) Some interfaces require a pulsed strobe rather than the level provided by the 753, fig. 2 shows a simple way of solving this problem. Adjust the pulse width by varying the R-C values (1 MS shown).

7. Some software requires certain control codes not assigned to keys on the 753. The user-defined keys (Repeat, Break, & Here-Is) may be hardwired for these functions. Connect one side of each switch to the appropriate pin shown on the schematic (over).

8. Maintenance. The keys may be cleaned with ordinary soap and water. Protect the MOS encoder with the conductive foam when handling or transporting the keyboard.

**PIN ASSIGNMENTS**

Bit 6

| Upper-case | B6B |
| B6A |

**FIG. 2**

<table>
<thead>
<tr>
<th>14</th>
<th>11</th>
<th>10</th>
<th>9</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>74121</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STROBE PULSES**

- +5 VDC Logic
- GND

**NOTICE FOR KEYING**

- Jumper DATA and STROBE Parity Outputs for GND
- 2-95A PROVIDES BOTH UPPER AND LOWER CASE OPTION
- 3-95A PROVIDES UPPER CASE ONLY
- 4-95A PROVIDES COMPLETE INSTRUCTION ON CONNECTORS
- 5-95A PROVIDES PARTIAL INSTRUCTION ON CONTACT CLOSURES
- 6-95A PROVIDES COMPLETE INSTRUCTION ON CONNECTORS
POWER PINS:

+5  -12  GND

AY-5-2376  1  18  17

Note: Elexon DC-512 DC-DC Converter mounts on board if -12v not available.

11. Jumper DSI (data-Strobe invert) to +5 for neg. logic, to GND for pos. logic.
10. Strobe level goes when data is valid.
9. B6A is UC and LC.
8. Use B6B for UC only.
7. Repeat, Break, Here-is not encoded.
6. Numbers in boxes are ASCII keycodes.
5. Output drive capacity 1 std. TTL load.
4. All outputs TTL-CMOS compatible.
3. ( ) indicates optional Numeric Pad key.
2. Capacitors are disc ceramic.
1. All Resistors 1/4 watt, carbon comm.

1  2  3  4  5  6  7  8  9  (0) - =
Q  W  E  R  T  Y  U  I  O  P  @
A  S  D  F  G  H  J  K  L  ; : *
Z  X  C  V  B  N  M  , . / ? _
ETB  SP  CR  FF  LF  HT  SI  BS  DC1  VT  US
ACK  HT  SUB  ESC  EM  FS  ~↑↓\ (7) (8) (9)
SOH  DLE  DEL  GS  CAN  ETX  13C5 (4) (5) (6)
NUL  DC4  NAK  STX  SYN  GS  SUB  RS (1) (2) (3)

UNLESS OTHERWISE SPECIFIED
DIMENSIONS ARE IN INCHES TOLERANCES ON FRACTIONS DECIMALS ANGLES
± 1/64  ± .01  ± .05  ± 0°30'

MATERIAL  FINISH

GEORGE RISK INDUSTRIES, INC.
KIMBALL, NEBRASKA

ELECTRICAL SCHEMATIC - 753 KEYBOARD

STANDARD BLUE PRINT, OMAHA 111874

DRAWING NUMBER 2-753-003
This sheet supercedes all other instructions or specifications. PLEASE READ CAREFULLY!

Assembly instructions on your packaging card are NOT correct. Your Model 753 uses a new, single sided printed circuit board, instead of the double sided board shown on the card. Please substitute the instructions on this sheet for those printed on the reverse side of the packaging card. Retain this sheet for future reference.

THERE IS NO CHANGE TO INSTRUCTION STEPS 1, 2, 4, 5, or 7. Follow the directions on the card.

Step 3. PARTS LOCATION IS CHANGED.
Refer to the drawing below to install the resistors, capacitors, and other parts. Note proper orientation of the 40 pin IC socket.

Step 6. CONNECTOR PIN ASSIGNMENTS ARE CHANGED.
Refer to the new connector drawing below to wire up your interface cable. Notice that the drawing is shown looking at the solder side of the pc board. Be sure to double check all connections before applying power, and measure correct voltages at the 40 pin socket before inserting the 40 pin encoder IC. Observe all precautions for handling the MOS encoder, per the instructions.

Aside from the above changes, your 753 keyboard is exactly the same as the previous model, and as added benefits, it has greater static resistance, is more tolerant of spilled liquids, etc, because no exposed traces are on the top side of the board, and has no troublesome plated-thru holes to worry about. The "User defined" keys are terminated at solder pads, to allow the use of our standard 15P 15 position card edge connector. We hope you enjoy using your new and better Model 753 keyboard.

1  KR2376 I.C.
2  4.7K resistors
3  100K resistor
4  680K resistor
5  .05 uf capacitors
6  .0033 uf capacitor
7  56 pf capacitor

A optional DC512 converter for +5 volt only use
B USER DEFINED KEY termination area
C "Kluge Area"

CARD EDGE CONNECTOR PIN ASSIGNMENTS

Connector type: Cinch 251-15-30 (not incl)

Temporary data sheet 4/79