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 PFM 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 impliment 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 recomment 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

Z80PIO(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

BIG BOARD PARTS LIST (BASIC I/O)

SEMICONDUCTORS

	DEVICE		QUANTITY	COMMENTS
130	74LS00	65	2	QUAD 2 INPUT NAND
71	74LS02	71	1	QUAD 2 INPUT NOR
355	74LS04	71	5	HEX INVERTER
93	7406	93	1	HEX O.C. INVERTING BUFFER
210	74LS08	1.05	2	QUAD 2 INPUT AND
71	74LS10	71	1	TRIPLE 3 INPUT NAND
399	74LS14	1.33	3	SCHMITT HEX INVERTER
65	74LS20	65	1	DUAL 4 INPUT NAND
142	74LS32	.71	2	QUAD 2 INPUT OR
145	7445	1.45	1	BUFFERED O.C. BCD DECODER
360	74LS74	90	4	DUAL D FLIP FLOP
127	74136	1.27	1	QUAD 2 INPUT XOR
572	74LS123	1.74	3	DUAL MONO
508	74LS138	1.27	4	BINARY DECODER
117	74LS151	1.17	1	1 OF 8 MUX
448	74LS157	1.12	4	QUAD 2:1 MUX
224	74157	1.12	2	HI DRIVE VERSION OF ABOVE
171	74LS161	1.71	1	PRESETTABLE SYNCHRONUS BINARY COUNTER
174	74164	1.74	1	8 STAGE SHIFT REG.
254	74LS174	1.27	2	HEX D LATCH
167	74LS193	1.67 (N)	1	SYNC BINARY U/D COUNTER
534	74LS241	2.67	2	OCTAL BUFFER
502	74LS242	2.51	2	4 BIT INVERTING BUS BUFFER
1004	74LS243	2.51	4	4 BIT BUS BUFFER
342	74LS283	1.71	2	4 BIT FULL ADDER
328	74LS290	1.64	2	BCD COUNTER
180	74LS293	1.80	1	BINARY COUNTER
369	74LS373	3.69	1	OCTAL LATCH
558	74LS393	2.79	2	DUAL 4 BIT BINARY COUNTER
8620	8216		2	SIGNETICS/INTEL BUS BUFFER
	2114		4	1K X 4 NMOS STATIC RAM
	MK2716-6		1	350 NS CHARACTER GENERATOR
	2716		1	PFM MONITOR ROM
	MK3880		1	Z80-CPU
	MK3881		1	Z80-PIO
	FD 1771-B01		1	WESTERN DIGITAL FLOPPY CONTROLLER
111.36	MK4116-4	3.48	32	16K DYNAMIC RAM
	2N2907	.62	1	PNP TRANSISTOR
	2N2222	.62	1	NPN TRANSISTOR
	1N751	.16	1	5.1 VOLT 1/2 WATT ZENER

149.96

RESISTORS

QUANTITY	VALUE
9	33 OHM 1/4 WATT
2	820 OHM 1/4 WATT
5	4.7K OHM 1/4 WATT
8	10K OHM 1/4 WATT
1	1.2K OHM 1/4 WATT
4	220 OHM 1/4 WATT
1	330 OHM 1/4 WATT
2	1.0K 1/4 WATT
2	100K 1/4 WATT
1	75 OHM 1/4 WATT
1	1.5K 1/4 WATT
1	3.9K 1/4 WATT
1	4.3K 1/4 WATT
1	39K 1/4 WATT
1	68K 1/4 WATT
2	8 PIN 1K SIP, PIN 1 COMMON

CRYSTALS

ICM PART NO.	FREQUENCY	
433165	14.31818 MHZ	ALL CRYSTALS ARE SERIES RESONANT
435260	20.000000 MHZ	32 PF LOAD, HC-18 HOLDER
433165	5.0688 MHZ (OPTIONAL)	

CAPACITORS

QUANTITY	VALUE
105	.01 - .1 UF DISC OR MONOLITHIC 16V OR MORE
1	.01 UF DISC CERAMIC .25" LEAD SPACING
3	33 PF DISC
1	47 PF DISC
1	.0033 UF DISC
8	2.2 UF (OR GREATER) 20V AXIAL LEAD TANTILUM
2	68 UF 10V RADIAL LEAD ELECTROLYTIC
1	1.0 UF 10V RADIAL LEAD ELECTROLYTIC
1	100 PF DISC

CONNECTORS

PART NUMBER	QUANITY	DSECRIPTION
CA-D26SP100-230-430	1	C.A. 26 PIN CONNECTOR
CA-D50SP100-230-430	1	C.A. 50 PIN CONNECTOR
	44	14 PIN SOLDERTAIL SOCKETS
	55	16 PIN SOLDERTAIL SOCKETS
	5	18 PIN SOLDERTAIL SOCKETS

27898

	3	20 PIN SOLDERTAIL SOCKETS 89	2.67
	5	24 PIN SOLDERTAIL SOCKETS 1.07	5.35
	5	40 PIN SOLDERTAIL SOCKETS 1.76	8.80
	1	28 PIN SOLDERTAIL SOCKET 1.24	1.24
25-104-0853	1	ELECTROVERT 8 POS. TERMINAL BLOCK	

247.04

RESISTOR VALUES

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

ALL CAPACITORS NOT INCLUDED IN THE LIST BELOW ARE NON-CRITICAL AND CAN BE ANY DISK OR MONOLITHIC BETWEEN .01 UF TO .1 UF AT 16 VOLTS OR MORE

CAPACATOR 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
C-53 = 180 PF (H. SYNC TIMING)
C-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)
C-140 = 68 UF 10 V
C-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

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 U96, 97, 99, 100, 101, 103, 104, 105, 108, and 110.

[] 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 followins 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 indentaion on the resistor pack. Install the packs tilted at a slisht ansle away from connector J1 to allow for clearance with the matins floppy disk connector.

[] Install and solder TB1, the power connector for the Bis 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 insertins any IC's in their sockets, apply +5, +12, -12, and sround to TB1. With an accurate voltmeter, check the voltase on C21. It should be -5VDC (within 5%). When measurins voltases, be very careful not to short your probes to any of the srounds on the PCB, this would short out your power supplies.

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

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

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

[] The power supplies used to run the Bis Board MUST be of the hishest quality and should have Over Voltase Protection (OVP) alons with current limitins. A cheap power supply could become very expensive if you have to replace all of the IC's on a Bis 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 your self 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. Plug 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 Users 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 Asignments" 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 SIO/O 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 SIO 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 impedence 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 "RFSH" is high, generation of "CAS" and "MUXC" is enabled. "RFSH" 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

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 accomidate 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 "M1R" 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

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. Synchronus 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 * Rt * 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

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 accomidate the timing characteristics of diffrent 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 uncomitted 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 synchronus data communication with one of the involved protocols, you are elegible 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 0FF 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

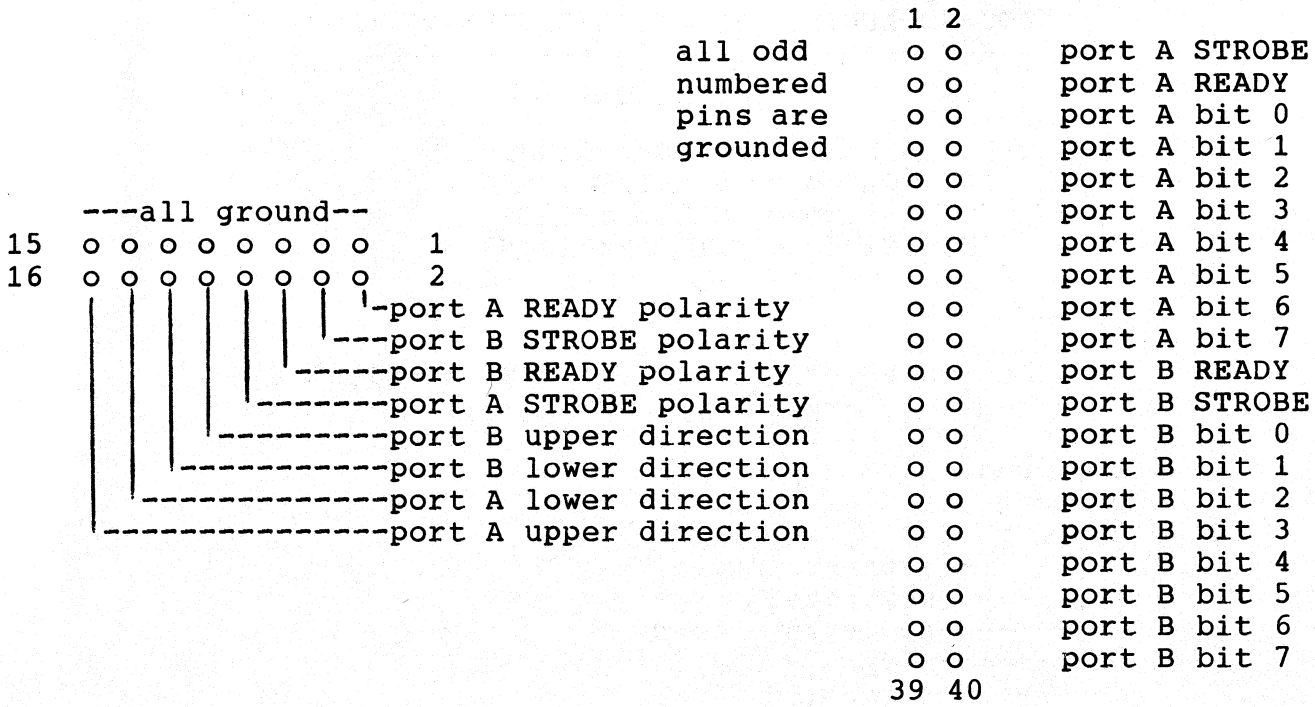
POWER SUPPLY REQUIREMENTS:

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)



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 "PBRDY" 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.

CTC STRAPPING AND I/O ASSIGNMENTS (JB2)

```
                  2 1
SYSTEM CLOCK      o o   CLOCK/TRIGGER 0
     ZC-TO0       o o   CLOCK/TRIGGER 1
     ZC-TO1       o o   CLOCK/TRIGGER 2
     ZC-TO2       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 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
```

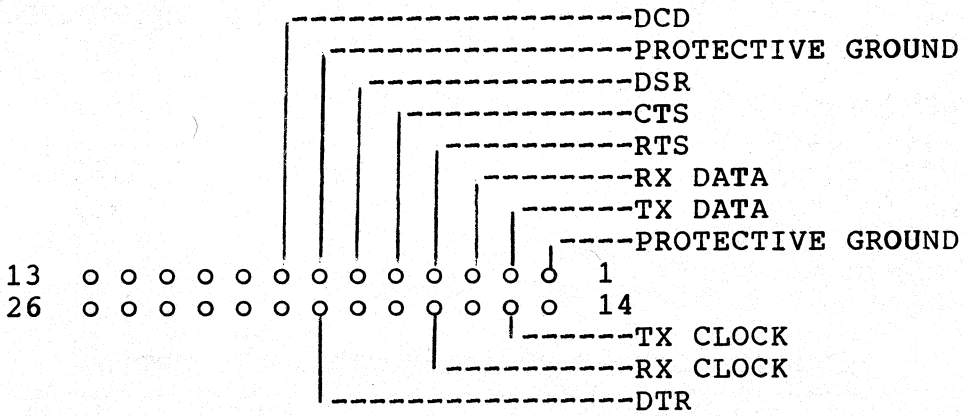
with no strapping options exercised:
- inverted video
- negative h.sync
- negative v.sync

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

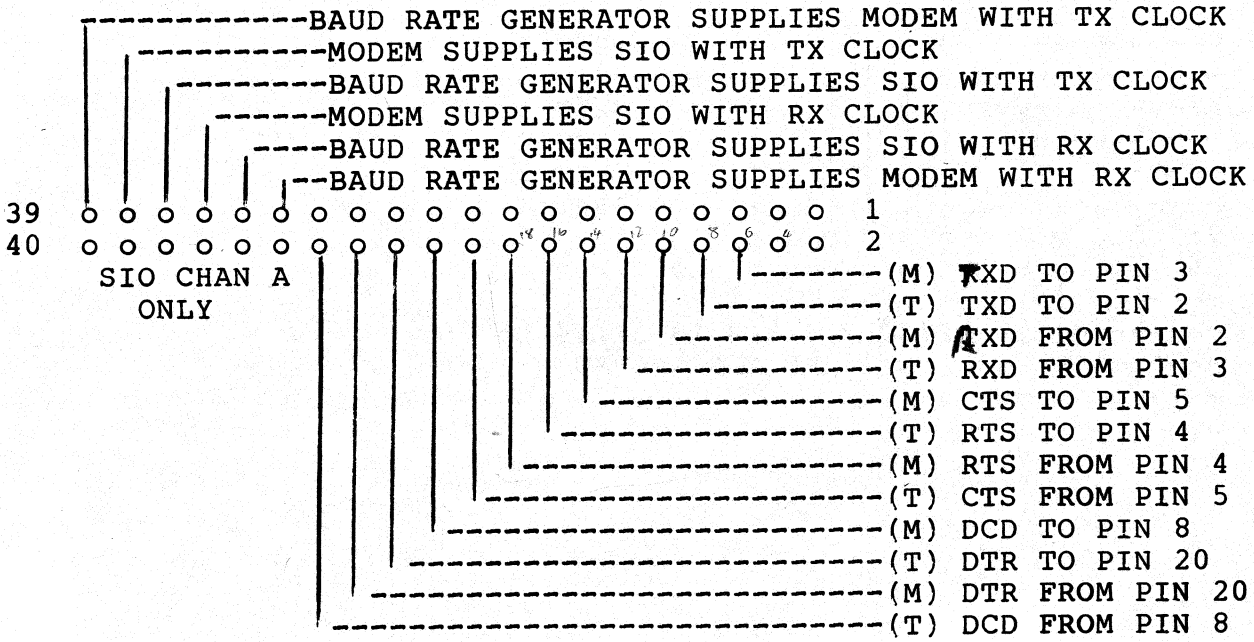
install R-21, R-25, R-26, Q-1, and R-20
install a 33 pf capacitor next to R-21

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



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

ONLY CHANNEL A IS CAPABLE OF ULITLIZING 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.



DATA SET READY IS ACTIVE ON BOTH CHANNELS

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.

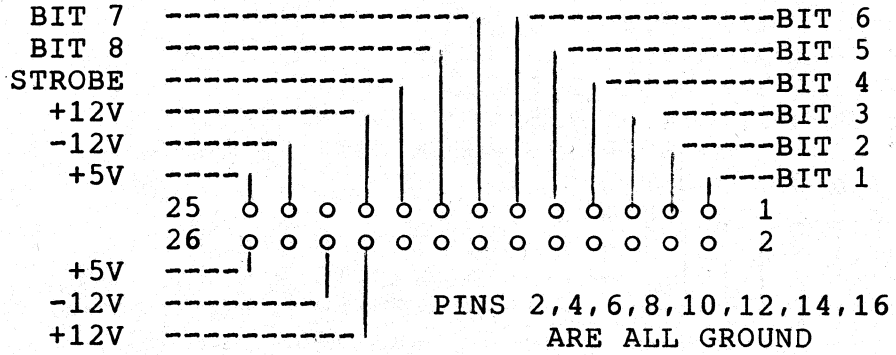
TXD=TRANSMITTED DATA
RXD=RECEIVED DATA
RTS=REQUEST TO SEND
CTS=CLEAR TO SEND
DTR=DATA TERMINAL READY
DCD=DATA CARRIER DETECT

KEYBOARD STROBE POLARITY STRAPPING OPTION

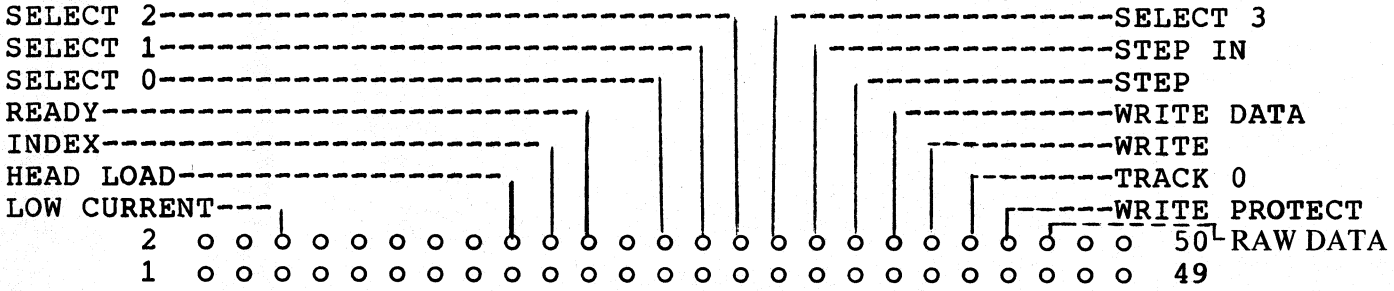
o o o
JB6

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)

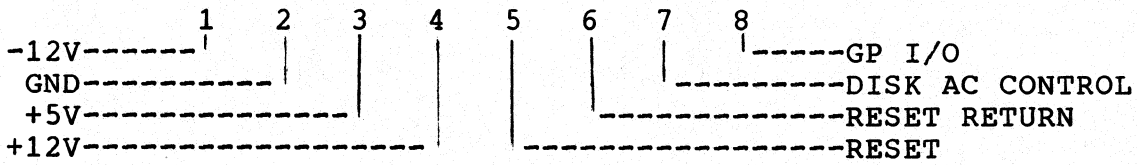


DISK CONNECTOR PIN ASSIGNMENTS (J1)



ALL ODD NUMBERED PINS ARE GROUNDED

POWER CONNECTOR PIN ASSIGNMENTS (TB1)



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 by 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 temporarialy 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

by the command. A detailed description of each command is provided in the the following pages.

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

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 CCCCCCCCCCCCCCCC

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 bewteen 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.

- 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 occurr, 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 addres 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.

6) VERIFY COMMAND <THIS COMMAND DELETED>

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.

bit	read/write	seek/restore/select
7	drive not ready	drive not ready
6	write protected	unused
5	write fault	unused
4	record not found	seek error
3	crc error	crc error
2	lost data	cannot restore
1	unused	unused
0	always=1	always=0

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.

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 necessarialy 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.

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.

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

17	^W	ETB - CLEAR TO END OF SCREEN
18	^X	CAN - CLEAR TO END OF LINE
19	^Y	EM - NONE
1A	^Z	SUB - CLEAR SCREEN
1B	^[	ESC - ESCAPE SEQUENCE
1C	^\	FS - NONE
1D	^]	GS - NONE
1E	^^	RS - HOME CURSOR
1F	^_	VS - DISPLAY CONTROL CHARACTER

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 recieved 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 recieved 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 successfuly, 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 driver routines will attempt to recover from any disk I/O errors that occurr, 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. Recieved 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

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 recieve 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. Chanels 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 sypply 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.

USER ACCESSABLE MONITOR ROUTINES AND VARIABLES

This section gives the locations and calling sequences of the user accessable 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.

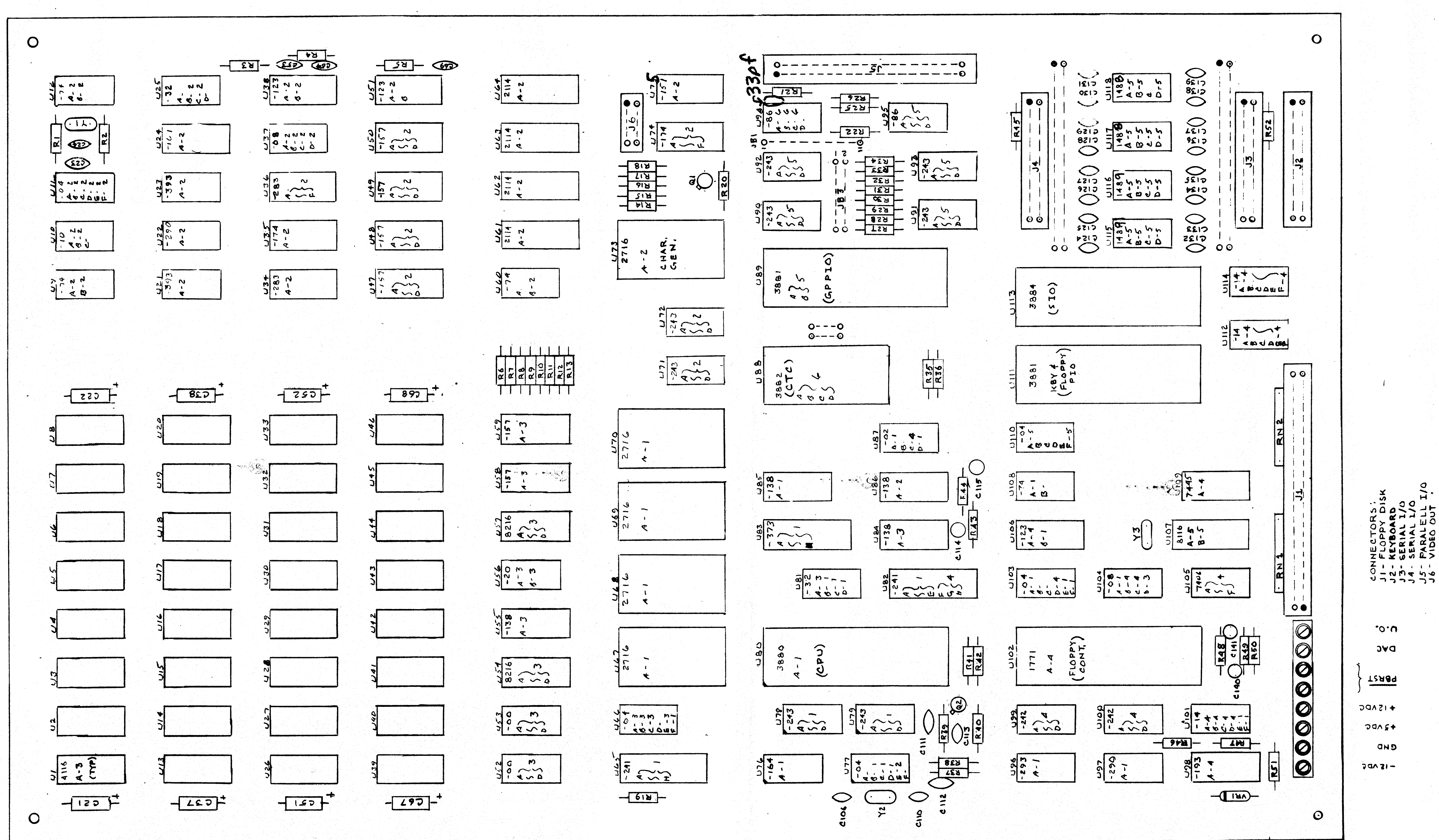
1) PFM-80 SUBROUTINE ENTRY POINTS

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

FUNCTION	PARAMETERS	DESCRIPTION
COLD	IN: none OUT: does not return	Perform cold start initialization of PFM-80 monitor and enter command mode.
WARM	IN: none OUT: does not return	Enter PFM-80 monitor command mode with no re-initialization.
CONST ...	IN: none OUT: status in a	Test for data ready in console input FIFO and return status in A. If data is available then A=FF, else A=0 hex.
CONIN ...	IN: none OUT: character in A	Return character from console input FIFO in A. If FIFO is empty then loop until character is input.
CONOUT ..	IN: character in A OUT: none	Output character passed in A to the console output device.
CRTOUT ..	IN: character in A OUT: none	Output character passed in A to the memory-mapped CRT display.
SIOST ...	IN: none OUT: status in A	Test for recieved data available from SIO channel B and return status in A. if data is available then A=0, else A=FF hex.
SIOIN ...	IN: none OUT: character in A	Return recieved data from SIO channel B in A. Loop until data is recieved if none is available on entry.
SIOOUT ..	IN: character in A OUT: none	Output character passed in A to SIO channel B transmit register.
SELECT ..	IN: unit number in C OUT: status in A	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.
HOME.....	IN: none OUT: status in A	Move read/write head to home position at track 0 And verify if it got there
SEEK	IN: track number in C OUT: status in A	Move read/write head to specified track And verify if it got there.
READ	IN: sector number in C buffer pointer in HL OUT: status in A	Read specified sector on current track into memory data buffer.
WRITE ...	IN: sector number in C buffer pointer in HL OUT: status in A	Write specified sector on current Track from memory data buffer.

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          RECLEN:  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
```



LAYOUT, CIRCUIT USAGE

R38 1K
R37 1K
C112 .01µF
U77 74LS04
Y2 20MHZ
C109 47PF
C110 100PF
U97 74LS290
U96 74LS293
C113 33PF
R39 220
R40 1.2K
Q2 2N2907
4MHZ (4)
2MHZ (4)
(2.5MHZ)
1.25MHZ (N6)
Φ (1,5)
U80 3880
Vcc
A15 A14 A13 A12 A11 A10 A9 A8
T4LS373 U83
A7 A6 A5 A4 A3 A2 A1 AØ
T4LS241 U65
(4,5) {A1 AØ
(3) RCEØ
(3) RCE1
(3) RCE2
(3) RCE3
U67 2716
U68 2716
U69 2716
U70 2716
ADDITIONAL ROM AREA
A15B A14B A13B A12B A11B (2, A10B A9B A8B (2 A7B (1 A6B (2 A5B (2, A4B (1,2 A3B A2B (1,2 A1B AØB (2
(1) MREQB
(1) RFSH
U66 74LS04
U87 74LS02
U76 74164
CAS (3)
MUXC (3)
D7 D6 D5 D4 D3 D2 D1 DØ
U78 74LS243
U79 74LS243
D7B D6B D5B D4B D3B D2B D1B DØB
U87 74LS02
BUSAK MREQ RD WR IORQ M1 RFSH HALT
U82 T4LS241
MREQB (1) RDB WRB IORQB (1) M1B (1) RFSH (1,3)
GND
U85 74LS138
(1) A2B A3B A4B A7B IORQB M1R
BAUDA (5) SIOCE GPPIO BAUDB (5) 1771CS (4) SCROLL (1) CTCCE (4) KBDPIO (4)
(PØ~3) (P4~7) (P8~B) (PC~F) (P10~13) (P14~17) (P18~1B) (P1C~1F)
"RESET"
TBI-6 TBI-5
R50 220
R49 10K
R48 10K
C140 68µF
C141 68µF
U101 74LS14
U103 74LS04
U81 74LS32
M1B
R43 10K
C114 .0033
U106 74LS123
U108 74
WAIT
BUSRQ
RST
R36 10K
(4,5) INT
INT
(4) DRQ
(4) INTRQ
NMI
U104 74LS08
M1R (1)
RST

PROCESSOR

C23 .01
U11 74LS04
C24 33P
Y1 14.318 MHZ
U11 74LS04
R1 820
R2 820
U11 74LS04
U24 74LS161AN
PE TE DA DB DC DD CK CLR TC LD QA QB QC QD
(2) DCA (2) DCB (2) DCC
U11 74LS04
÷7
U10 74LS10
(2) CC5
(2) CC6
U11 74LS04
HRST
U23 74LS393
CKA CKB A1 A2 A3 A4 B1 B2 B3 B4
128
HORIZ
(2) DCTC
(2) CC0 CC1 CC2 CC3 CC4 CC5 (2) CC6
U22 74LS290
CKA CKB A B C D RA RB
÷10
U9 74LS74
÷2
(2) SC0 SC1 SC2 (2) SC3
(2) LC0
U21 74LS393
CKA CKB A1 A2 A3 A4 B1 B2 B3 B4 RA RB
72 64 128 256
(2) LC1 LC2 LC3 LC4
U37 74LS08
U10 74LS10
(2) LC1
(2) SC3
U9 74LS74
(N/C)
U37 74LS08
U36 74LS283 C4 B4 A4 B3 A3 B2 A2 B1 A1 Σ4 Σ3 Σ2 Σ1 CI
LC4
U34 74LS283 C4 B4 A4 B3 A3 B2 A2 B1 A1 Σ4 Σ3 Σ2 Σ1 CI
LC3
LC2
LC1
LCØ
(1) A12B A11B A10B A9B A8B
U35 74LS174 DA DB DC DD DE DF QA QB QC QD QE QF CK
(1) SCROLL
U48 74LS157 BØ AØ B1 A1 B2 A2 B3 A3 QØ Q1 Q2 Q3 A/B E
(1) A11B (1) A10B (1) A9B (1) A8B
U49 74LS157
(1) A7B (2) CC6 (1) A6B (2) CC5 (1) A5B (2) CC4 (1) A4B
U50
13 14 10 11 12 9
U47 74LS157
(2) CC3 (1) A3B (2) CC2 (1) A2B (2) CC1 (1) A1B (2) CCØ (1) AØB
(A) DATA WR
U11 74LS04
U63 2114
U62 2114
U64 2114
U61 2114
A9 A8 A7 A6 A5 A4 A3 A2 A1 AØ D3 D2 D1 DØ E R/W
R4 100K
C54 4700P
U94 74LS86/74LS136
U38 74LS123 A1 B1 Q1 R
R22 10K
R25 3.9K
Q1 2N2222
R20 75
J6
-10 COMP.V.
0 -8
-6 V.SYNC
-4 H.SYNC
-2 VIDEO
C142 33P
R21 1.5K
R26 4.3K
U94 74LS86/74LS136
(2) CC4 (2) CC5 (2) CC6
U25 74LS32
U37 74LS08
R5 39K
C69 470P
R3 68K
C53 180P
U51 74LS123
U38 74LS123
(2) DCB
U38 74LS74
(2) DCC
(2) DCB
(2) DCA
U74 74LS174 DA DB DC DD DE DF CK QA QB QC QD QE QF
U75 74LS151 A B C S W Y Ø
R15 10K R14 10K R16 10K R17 10K R18 10K
(3) CRTCE
(2) DCTC
U12 74LS74 U12 74LS74
(1) RDB
(1) WRB
(1) MREQB
U86 74LS138 A B C G1A G1B G2 Ø 1 2 3 4 5 6 7
DATA WR
U71 74LS243
U72 74LS243
(1) D7B D6B D5B (1) D4B
(1) D3B D2B D1B (1) DØB
U77 74LS04
DATA RD
U73 2716 VPP A10 A9 A8 A7 A6 A5 A4 A3 A2 A1 AØ OE CS DØ D1 D2 D3 D4 D5 D6 D7
(2) SC2 (2) SC1 (2) SC0

CRT DISPLAY GEN.

(1) CAS — U66 — R19 33

74157 U59: (1) WRB; (1) A13B, A6B, A12B, A5B, A11B, A4B; A/B, E; (3) RAS3, RAS2, RAS1, RASØ

R12 33, R9 33, R7 33, R11 33, R8 33, R13 33, R6 33, R10 33

U1–U8, U13–U20, U26–U33, U39–U46: 4116 (RAS, CAS, WE, A6/13, A5/12, A4/11, A3/10, A2/9, A1/8, AØ/7, DI, DO)

74157 U58: (1) AIØB, A3B, A9B, A2B, A8B, A1B, A7B, (1) AØB; A/B, E; (1) MUXC

U56 74LS20: (3) SELECT, (1) MREQB, (1) RDB — U66 74LS04

U54 8216, U57 8216: DØB (1), D1B (1), D2B (1), D3B (1), D4B (1), D5B (1), D6B (1), D7B (1)

(1) A11B, A12B, (1) A13B — U84 74LS138 — (1) MREQB — U104 74LS08 — RCEØ (1), RCE1, RCE2, RCE3 (1), CRTCE (2)

(4) BANK — U81 74LS32

(1) A14B, (1) A15B — U55 74LS138 — U56 74LS20

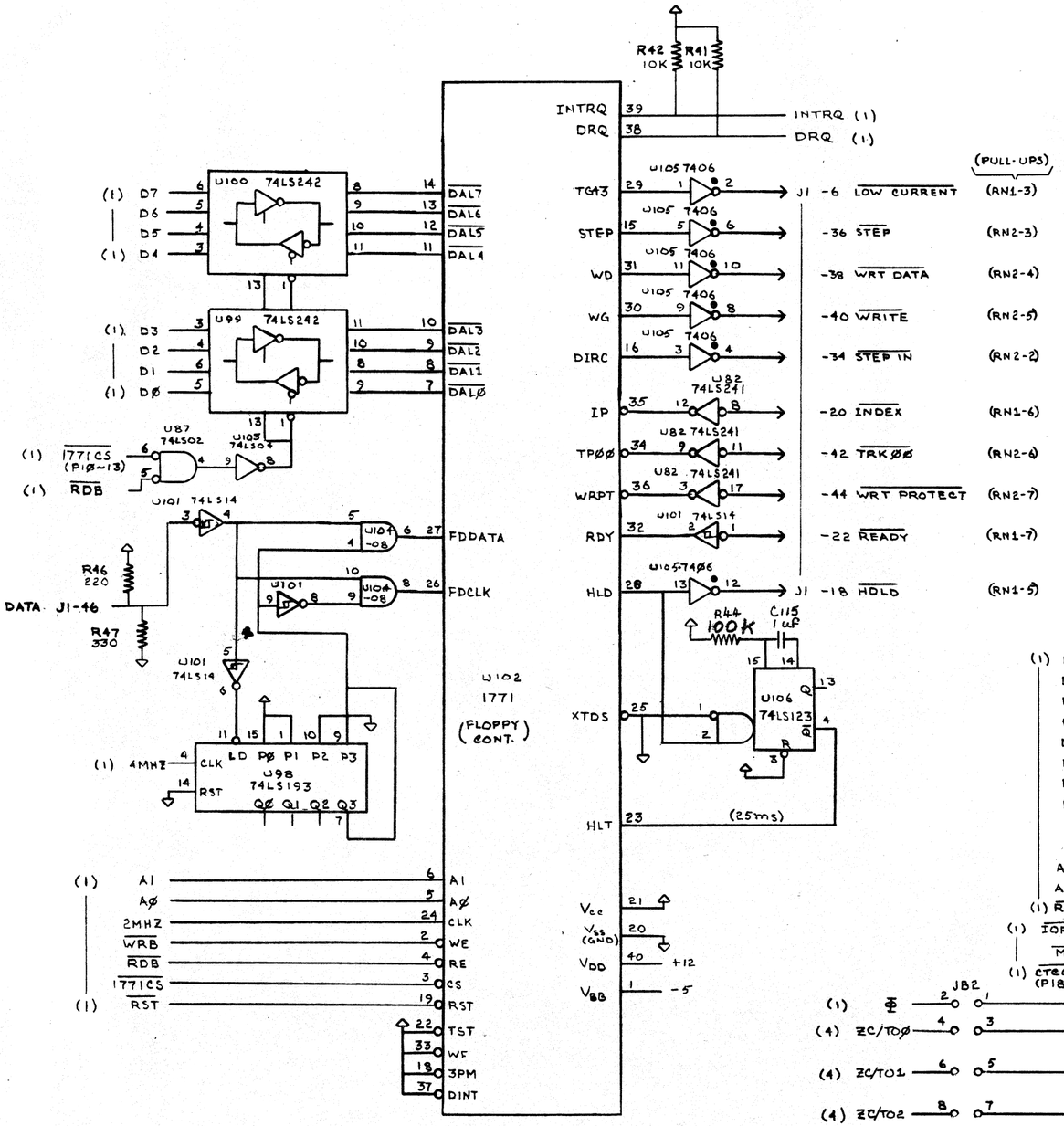
(1) RFSH — U53 74LS00 — U66 74LS04 — (1) MREQB — U52 74LS00

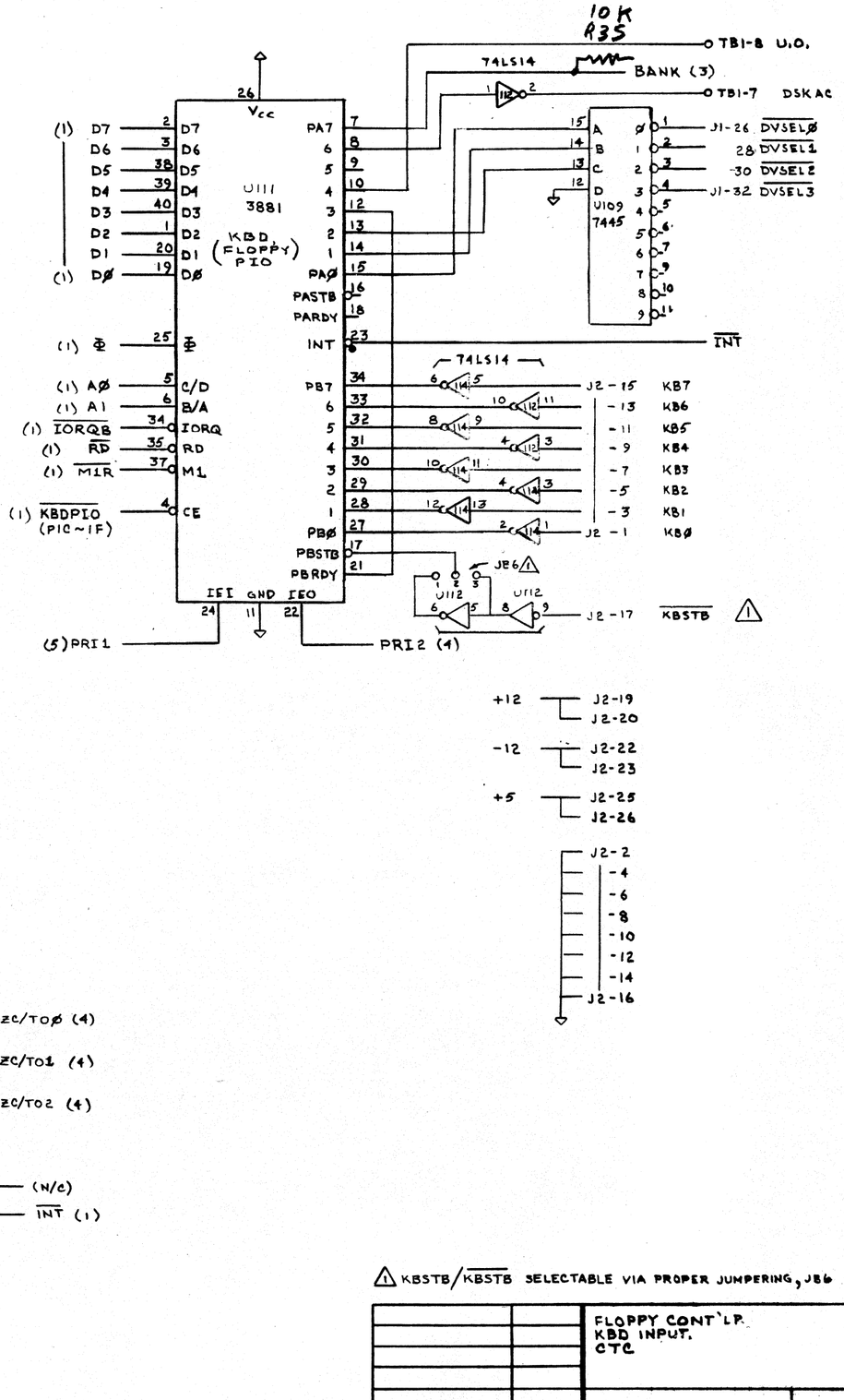
SELECT (3), RAS3, RAS2, RAS1, RASØ (3)

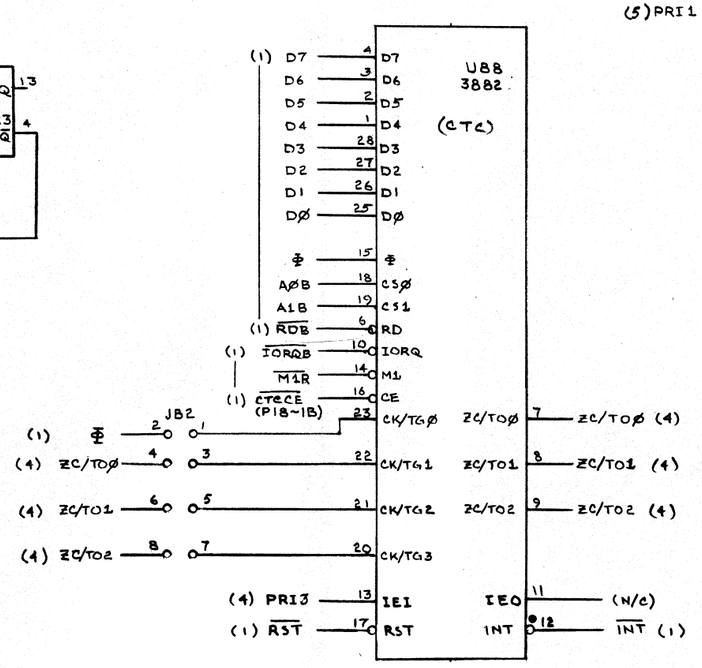
4116 POWER: PIN 1: -5VDC; PIN 8: +12VDC; PIN 9: +5VDC; PIN 16: GND

RAM

3

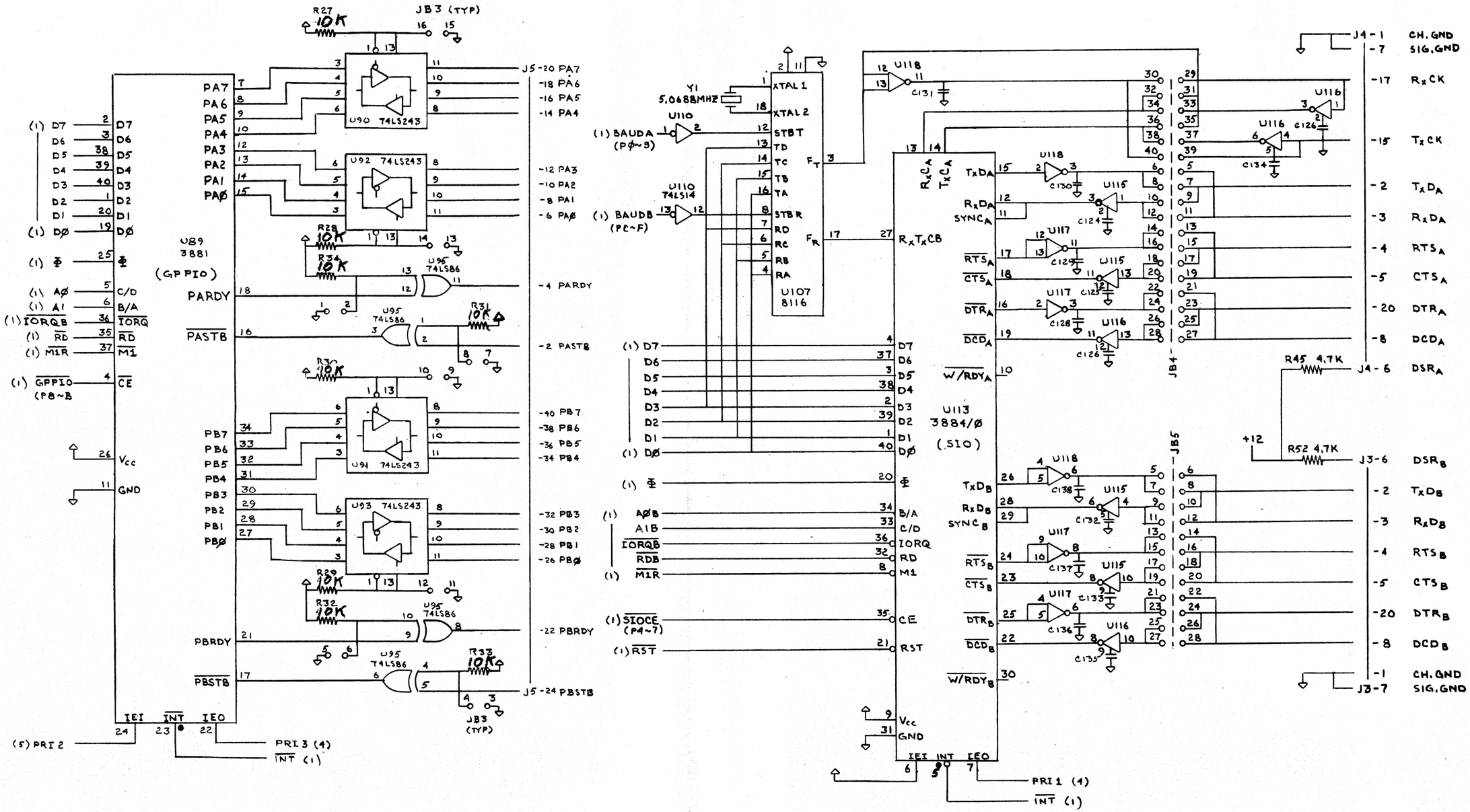






⚠1 KBSTB/$\overline{\text{KBSTB}}$ SELECTABLE VIA PROPER JUMPERING, JB6

		FLOPPY CONT'LR. KBD INPUT. CTC
		4



NOTE(S)
1. CAPS ON SIO IN/OUT DRIVERS: 390 PF

GP PIO/SIO

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
>F000               0010 ROM      EQU     0F000H      ;START OF 2K ROM
>FF00               0011 RAM      EQU     0FF00H      ;START OF 256 BYTE RAM
>3000               0012 CRTMEM   EQU     3000H       ;BASE OF 4K CRT MEMORY
                    0013 ;
                    0014 ;
>F000               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 ;
F000  C32AF0        0029 COLD:    JP      INIT        ;MONITOR COLD ENTRY POINT
F003  C32BF1        0030 WARM:    JP      PROMPT      ;MONITOR WARM ENTRY POINT
F006  C331F4        0031 CONST:   JP      KBDST       ;CONSOLE STATUS VECTOR
F009  C339F4        0032 CONIN:   JP      KBDIN       ;CONSOLE INPUT VECTOR
F00C  C320F5        0033 CONOUT:  JP      CRTOUT      ;CONSOLE OUTPUT VECTOR
F00F  C320F5        0034          JP      CRTOUT      ;CRT OUTPUT VECTOR
F012  C3E8F4        0035          JP      SIOST       ;SIO CHANEL B STATUS VECTOR
F015  C3F0F4        0036          JP      SIOIN       ;SIO CHANEL B INPUT VECTOR
F018  C3FEF4        0037          JP      SIOOUT      ;SIO CHANEL B OUTPUT VECTOR
F01B  C3B1F6        0038          JP      SELECT      ;DISK DRIVE SELECT
F01E  C3E9F6        0039          JP      HOME        ;HOME R/W HEAD
F021  C3FBF6        0040          JP      SEEK        ;SEEK TO TRACK
F024  C32AF7        0041          JP      READ        ;READ SECTOR
F027  C31FF7        0042          JP      WRITE       ;WRITE SECTOR
                    0043 ;
                    0044 ;
                    0045 ;
                    0046 ;DO A SHORT POST-RESET TIME DELAY. ALSO INITIALIZES THE
                    0047 ;STACK POINTER AND FILLS THE MONITOR SCRATCH RAM WITH ZEROS
                    0048 ;
F02A  F3            0049 INIT:    DI
F02B  2100FF        0050          LD      HL,RAM      ;POINT TO START OF MONITOR RAM
F02E  3600          0051 INIT1:   LD      (HL),0      ;FILL 256 BYTE SPACE WITH ZEROS
F030  F9            0052          LD      SP,HL       ;SOMETHING USEFUL TO ADD DELAY
F031  2C            0053          INC     L
F032  20FA          0054          JR      NZ,INIT1-$ ;LOOP TAKES 4 MILLISECONDS
                    0055 ;
                    0056 ;INITIALIZE THE Z-80 FOR INTERRUPT MODE #2
                    0057
F034  7C            0058          LD      A,H
```

```
F035  ED47      0059          LD      I,A         ;LOAD I REG W[redacted] MSB OF VECTOR
;                                                  TABLE
F037  ED5E      0060          IM      2           ;AND SELECT INTERRUPT MODE 2
                0061 ;
F039  CDECF5    0062          CALL    CLRSCN      ;FILL THE CRT MEMORY WTH BLANKS
                0063 ;
                0064 ;STORE ANY NON-ZERO VALUES FOR VARIABLES IN MEMORY
                0065 ;
F03C  21D3F0    0066          LD      HL,INTAB    ;POINT TO DEFAULT VAR TABLE
F03F  0600      0067 INIT2:   LD      B,0
F041  4E        0068          LD      C,(HL)      ;BC=DATA BLOCK BYTECOUNT
F042  23        0069          INC     HL
F043  5E        0070          LD      E,(HL)      ;DE=DESTINATION FOR DATA
F044  23        0071          INC     HL
F045  56        0072          LD      D,(HL)
F046  23        0073          INC     HL
F047  EDB0      0074          LDIR                ;COPY DATA @ HL TO VAR @ DE
F049  CB7E      0075          BIT     7,(HL)
F04B  28F2      0076          JR      Z,INIT2-$   ;LOOP AGAIN IF NOT END OF TABLE
                0077 ;
                0078 ;INITIALIZE THE PROGRAMMABLE I/O DEVICES
                0079 ;
F04D  23        0080          INC     HL          ;POINT TO I/O INIT DATA TABLE
F04E  46        0081 INIT3:   LD      B,(HL)      ;B=INIT LOOP BYTECOUNT
F04F  23        0082          INC     HL
F050  4E        0083          LD      C,(HL)      ;C=DEVICE CONTROL PORT#
F051  23        0084          INC     HL
F052  EDB3      0085          OTIR                ;SEND DATA @ HL TO PORT @ C
F054  CB7E      0086          BIT     7,(HL)      ;TEST FOR TABLE END MARKER
F056  28F6      0087          JR      Z,INIT3-$   ;LOOP AGAIN IF NOT AT END
                0088 ;
                0089 ;DETERMINE IF CONSOLE I/O CONFIGURATION WILL BE FOR THE
                0090 ;ON-BOARD CRT AND KEYBOARD OR AN EXTERNAL SERIAL TERMINAL.
                0091 ;
F058  ED78      0092          IN      A,(C)       ;TEST SIO READ REG 2 TO CHECK
F05A  FE06      0093          CP      00000110B   ;IF THE SIO IS INSTALLED
F05C  2012      0094          JR      NZ,PARALL-$;SKIP CONFIG TEST IF NO SIO
F05E  DB1E      0095          IN      A,(KBDDAT) ;MAKE SURE KBD PIO 'READY'RESET
F060  0610      0096          LD      B,00010000B;B=RESET SIO EXT STATUS COMMAND
F062  ED41      0097 DECIDE:  OUT     (C),B       ;TEST FOR ARRIVAL OF A SERIAL
F064  ED78      0098          IN      A,(C)       ;  INPUT CHAR START BIT
F066  CB67      0099          BIT     4,A
F068  200E      0100          JR      NZ,BAUD-$   ;EXIT LOOP IF START BIT DETECTED
F06A  DB1C      0101          IN      A,(BITDAT)
F06C  CB5F      0102          BIT     3,A         ;TEST FOR DATA RDY STROBE FROM
F06E  20F2      0103          JR      NZ,DECIDE-$;PARALLEL KBD, LOOP IF INACTIVE
F070  DB1E      0104 PARALL:  IN      A,(KBDDAT) ;DISCARD FIRST KEYBOARD CHAR
F072  3E83      0105          LD      A,10000011B
F074  D31F      0106          OUT     (KBDCTL),A ;ENABLE INTERRUPTS FROM KBD PIO
F076  182D      0107          JR      SIGNON-$    ;SIGNON WTH BUILT-IN CONSOLE I/O
                0108 ;
                0109 ;
                0110 ;AUTOMATIC BAUD RATE SETTING ROUTINE FOR SIO
                0111 ;
F078  AF        0112 BAUD:    XOR     A
F079  ED41      0113 BAUD1:   OUT     (C),B
F07B  ED50      0114          IN      D,(C)       ;READ SIO STATUS REGISTER
F07D  CB62      0115          BIT     4,D         ;TEST THE SYNC/HUNT BIT
F07F  28F8      0116          JR      Z,BAUD1-$   ;LOOP UNTIL IT CHANGES STATE
F081  3C        0117 BAUD2:   INC     A
F082  ED41      0118          OUT     (C),B       ;RESET REGISTER #0 FLAGS AGAIN
F084  ED50      0119          IN      D,(C)       ;&LOOP TIMING THE SYNC/HUNT BIT
F086  CB62      0120          BIT     4,D
F088  20F7      0121          JR      NZ,BAUD2-$  ;REPEAT UNTIL BIT CHANGES AGAIN
```

```
F08A  21CAF0          LD      HL,RATES-1
F08D  23        BAUD3:  INC     HL              ;INDEX INTO BAUD RATE TABLE
F08E  17                RLA                     ; USING COUNT DERRIVED IN A
F08F  30FC              JR      NC,BAUD3-$
F091  7E                LD      A,(HL)          ;GET BAUD RATE CONTROL BYTE FROM
F092  D30C              OUT     (BAUDB),A       ;TBL & OUTPUT TO COM-8116 TIMER
F094  CDF0F4            CALL    SIOIN           ;DISCARD 1ST SERIAL INPUT CHAR
F097  3E01              LD      A,1
F099  D307              OUT     (SIOCPB),A      ;RE-PROGRAM SIO B TO GENERATE
F09B  3E1C              LD      A,00011100B     ;INTERRUPTS ON RECEIVED DATA.
F09D  D307              OUT     (SIOCPB),A      ;PARITY DOES NOT AFFECT VECTOR
F09F  21FEF4            LD      HL,SIOOUT
F0A2  220DF0            LD      (CONOUT+1),HL   ;RE-DIRECT CONSOLE OUTPUT
;                                                TO SIO
                        ;
                        ;PRINT SIGNON MESSAGE
                        ;
F0A5  FB        SIGNON: EI
F0A6  CDECF3            CALL    PNEXT
F0A9  0D0A              DEFB    CR,LF
F0AB  2E2E2E20          DEFM    '... system monitor 3.3 ...'
      73797374
      656D206D
      6F6E6974
      6F722033
      2E33202E
      2E2E
F0C5  0D0A              DEFB    CR,LF
F0C7  04                DEFB    EOT
F0C8  C303F0            JP      WARM            ;GO ENTER MONITOR
                        ;
                        ;
                        ;
                        ;BAUD RATE CONSTANTS FOR COM 8116 BAUD RATE GENERATOR
                        ;
F0CB  05        RATES:  DEFB    0101B           ;    300 BAUD
F0CC  06                DEFB    0110B           ;    600 BAUD
F0CD  07                DEFB    0111B           ;   1200 BAUD
F0CE  0A                DEFB    1010B           ;   2400 BAUD
F0CF  0C                DEFB    1100B           ;   4800 BAUD
F0D0  0E                DEFB    1110B           ;   9600 BAUD
F0D1  0F                DEFB    1111B           ;  19200 BAUD
F0D2  0F                DEFB    1111B           ;  19200 BAUD
                        ;
                        ;
>F0D3           INTAB   EQU     $               ;INITIALIZATION DATA TABLES
                        ;
                        ;
                        ;INITIALIZE THE Z-80 'I' REGISTER INTERRUPT VECTOR TABLE
                        ;
F0D3  02                DEFB    2
F0D4  1AFF              DEFW    SYSVEC+2
F0D6  8CF4              DEFW    KEYSRV          ;PARALLEL KBD INTERRUPT VECTOR
                        ;
F0D8  02                DEFB    2
F0D9  16FF              DEFW    CTCVEC+6
F0DB  9FF4              DEFW    TIMER           ;1 SEC TIMER INTERRUPT VECTOR
                        ;
F0DD  04                DEFB    4
F0DE  04FF              DEFW    SIOVEC+4
F0E0  AFF4              DEFW    SIOINT          ;SIO RECEIVE INTERRUPT VECTOR
F0E2  CFF4              DEFW    SIOERR          ;SIO PARITY, OVERRUN & FRAMING
;                                                ERROR
                        ;
                        ;INITIALIZE DISK I/O DRIVER VARIABLES
                        ;
F0E4  08                DEFB    8
```

```
F0E5  65FF      0181          DEFW    UNIT
F0E7  FF        0182          DEFB    255             ;FLAG ALL DRIVES AS DE-SELECTED
F0E8  FFFFFFFF  0183          DEFB    255,255,255,255 ;CLEAR HEAD POSITION TABLE
F0EC  00        0184          DEFB    00000000B       ;SELECT FASTEST SEEK SPEED
F0ED  80        0185          DEFB    128             ;SELECT 128 BYTE SECTOR LENGTH
F0EE  1E        0186          DEFB    30              ;SET MOTOR TURN-OFF TIMER
                0187 ;
                0188 ;INITIALIZE THE CRT DISPLAY CURSOR
                0189 ;
F0EF  02        0190          DEFB    2
F0F0  75FF      0191          DEFW    CHRSAV
F0F2  20        0192          DEFB    ' '
F0F3  5F        0193          DEFB    '_'             ;USE NON-BLINKING UNDERSCORE
                0194 ;
                0195 ;SET DEFAULT 'SOFTWARE' INTERRUPT VECTORS
                0196 ;
F0F4  06        0197          DEFB    6
F0F5  57FF      0198          DEFW    TIKVEC
F0F7  80F4      0199          DEFW    DSKTMR          ;POINT'TIKVEC'TO DISK MTR TIMER
F0F9  44F4      0200          DEFW    STASH           ;POINT'PINVEC'TO FIFO INPUT
;                                                      ROUTINE
F0FB  44F4      0201          DEFW    STASH           ;POINT'SINVEC'TO FIFO INPUT
;                                                      ROUTINE
                0202 ;
                0203 ;SET FREE MEMORY POINTER
                0204 ;
F0FD  02        0205          DEFB    2
F0FE  7AFF      0206          DEFW    FREPTR
F100  E6F7      0207          DEFW    ROMEND          ;POINT TO 1ST LOC AFTER MONITOR
                0208 ;
                0209 ;
F102  FF        0210          DEFB    -1              ;END OF VARIABLE INIT TABLE
                0211 ;
                0212 ;
                0213 ;
>0000           0214 BAUDA    EQU     00H             ;CHANEL A BAUD RATE GENETATOR
>0004           0215 SIO      EQU     04H             ;DUAL SERIAL I/O
>0008           0216 GENPIO   EQU     08H             ;GENERAL PURPOSE PARALLEL I/O
>000C           0217 BAUDB    EQU     0CH             ;CHANEL B BAUD RATE GENERATOR
>0010           0218 WD1771   EQU     10H             ;WEST DIGITAL DISK CONTROLLER
>0014           0219 SCROLL   EQU     14H             ;CRT SCROLL MEM SCROLL REGISTER
>0018           0220 CTC      EQU     18H             ;QUAD COUNTER/TIMER CIRCUIT
>001C           0221 SYSPIO   EQU     1CH             ;SYSTEM PARALLEL I/O
                0222 ;
                0223 ;INITIALIZE SYSTEM PIO FOR USE AS BANK-SWITCH,
                0224 ;DISK DRIVE SELECT AND PARALLEL KEYBOARD INPUT
                0225 ;
>001C           0226 BITDAT   EQU     SYSPIO+0
>001D           0227 BITCTL   EQU     SYSPIO+1
>001E           0228 KBDDAT   EQU     SYSPIO+2
>001F           0229 KBDCTL   EQU     SYSPIO+3
                0230
F103  031D      0231          DEFB    3,BITCTL
F105  CF        0232          DEFB    11001111B       ;PUT SYSTEM PIO IN BIT MODE
F106  18        0233          DEFB    00011000B       ;MAKE BITS 4 AND 3 BE INPUTS
F107  40        0234          DEFB    01000000B       ;DISABLE INTERRUPTS
                0235 ;
F108  011C      0236          DEFB    1,BITDAT
F10A  00        0237          DEFB    00000000B       ;DE-SELECT ROMS, ENABLE DRIVE 0
                0238 ;
F10B  021F      0239          DEFB    2,KBDCTL
F10D  4F        0240          DEFB    01001111B       ;PUT KBD PORT IN INPUT MODE
F10E  1A        0241          DEFB    SYSVEC+2        ;LOAD KEYBOARD INTERRUPT VECTOR
                0242 ;
                0243 ;
```

```
                        0244 ;INITIALIZE CHANELS 2 AND 3 OF THE CTC
                        0245 ;TO GENERATE ONE SECOND INTERRUPTS FROM CTC3
                        0246 ;
>0018                   0247 CTC0    EQU     CTC+0       ;CTC CHANEL 0 PORT#
>0019                   0248 CTC1    EQU     CTC+1       ;CTC CHANEL 1
>001A                   0249 CTC2    EQU     CTC+2       ;CTC CHANEL 2
>001B                   0250 CTC3    EQU     CTC+3       ;CTC CHANEL 3
                        0251
F10F  011B              0252         DEFB    1,CTC0
F111  10                0253         DEFB    CTCVEC      ;BASE INTERRUPT VECTOR FOR CTC
                        0254 ;
F112  021A              0255         DEFB    2,CTC2
F114  27                0256         DEFB    00100111B   ;PUT CTC2 IN TIMER MODE
F115  69                0257         DEFB    105         ;CTC2 PERIOD=105*256*400 NS
                        0258 ;
F116  021B              0259         DEFB    2,CTC3
F118  C7                0260         DEFB    11000111B   ;PUT CTC3 IN COUNTER MODE
F119  5D                0261         DEFB    93          ;CTC3 PERIOD=999936 uS
                        0262 ;
                        0263 ;
                        0264 ;INITIALIZE SIO CHANEL B FOR ASYNCHRONOUS SERIAL
                        0265 ;INTERFACE TO PRINTER OR TERMINAL
                        0266 ;
>0004                   0267 SIODPA  EQU     SIO+0       ;SIO DATA PORT A
>0005                   0268 SIODPB  EQU     SIO+1       ;SIO DATA PORT B
>0006                   0269 SIOCPA  EQU     SIO+2       ;SIO CONTROL/STATUS PORT A
>0007                   0270 SIOCPB  EQU     SIO+3       ;SIO CONTROL/STATUS PORT B
                        0271
F11A  010C              0272         DEFB    1,BAUDB
F11C  05                0273         DEFB    0101B       ;SET COM 8116 TO 300 BD DEFAULT
                        0274
F11D  0B07              0275         DEFB    11,SIOCPB
F11F  04                0276         DEFB    4           ;SELECT REGISTER #4
F120  45                0277         DEFB    01000101B   ;16X CLK,1 STOP BIT,ODD PARITY
F121  01                0278         DEFB    1           ;SELECT REGISTER #1
F122  04                0279         DEFB    00000100B   ;STATUS AFFECTS VECTOR,
                                                          NO INTERRUPTS
F123  03                0280         DEFB    3           ;SELECT REGISTER #3
F124  41                0281         DEFB    01000001B   ;7 BITS/RX CHAR
F125  05                0282         DEFB    5           ;SELECT REGISTER #5
F126  AA                0283         DEFB    10101010B   ;7 BITS/TX CHAR, ASSERT DTR
F127  02                0284         DEFB    2           ;SELECT REGISTER #2
F128  00                0285         DEFB    SIOVEC      ;LOAD INTERRUPT VECTOR BASE
F129  02                0286         DEFB    2           ;SELECT READ REG#2 FOR SIO TEST
                        0287
F12A  FF                0288         DEFB    -1          ;END-OF-TABLE
                        0289 ;
                        0290 ;INIT DONE
                        0291 ;
                        0292 ;
                        0293         INCLUDE MONITOR.ASM
                        0294 ;*************************************************************
                        0295 ;*                                                           *
                        0296 ;*      BASIC HEX MONITOR FOR Z-80 PROCESSORS                *
                        0297 ;*                                          3-Aug-80         *
                        0298 ;*                                                           *
                        0299 ;*************************************************************
                        0300 ;
                        0301 ;
                        0302 ;
                        0303 ;
```

```
F12B  CDECF3    0304 PROMPT: CALL    PNEXT
F12E  0D0A      0305         DEFB    CR,LF
F130  2A20      0306         DEFM    '* '
F132  04        0307         DEFB    EOT
F133  2188FF    0308         LD      HL,LINBUF
F136  0E20      0309         LD      C,32
F138  CD3BF3    0310         CALL    GETLIN      ;INPUT A BUFERED CONSOLE LINE
F13B  3835      0311         JR      C,WHAT-$    ;PRINT 'WHAT ?' IF INPUT ERROR
              0312
F13D  AF        0313         XOR     A
F13E  3284FF    0314         LD      (ESCFLG),A
F141  CDFCF3    0315         CALL    CRLFS
F144  3A88FF    0316         LD      A,(LINBUF)  ;GET FIRST CHAR IN LINE
F147  FE0D      0317         CP      CR
F149  28E0      0318         JR      Z,PROMPT-$  ;JUMP IF A NULL LINE
F14B  2182F1    0319         LD      HL,CMDTAB   ;SEARCH FOR A MATCHING CHAR
F14E  010B00    0320         LD      BC,CMDSIZ/3     ; IN COMMAND SEARCH TABLE
F151  CD60F3    0321         CALL    SEARCH
F154  201C      0322         JR      NZ,WHAT-$   ;TRY AGAIN IF SEACRH 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  3810      0327         JR      C,WHAT-$
F162  2A7CFF    0328         LD      HL,(PARAM1)
F165  ED5B7EFF  0329         LD      DE,(PARAM2)
F169  ED4B80FF  0330         LD      BC,(PARAM3)
F16D  CD80F1    0331         CALL    CALLX       ;CALL SUBROUTINE @ IX
F170  30B9      0332         JR      NC,PROMPT-$;GO BACK TO PROMPT IF NO ERRORS
              0333
F172  CDECF3    0334 WHAT:   CALL    PNEXT
F175  20776861  0335         DEFM    ' what ?'
      74203F
F17C  07        0336         DEFB    'G'-64      ;SAY 'what ?' AND BEEP THE BELL
F17D  04        0337         DEFB    EOT
F17E  18AB      0338         JR      PROMPT-$
              0339 ;
              0340 ;
F180  DDE9      0341 CALLX:  JP      (IX)        ;CALL SUBROUTINE @ IX
              0342 ;
              0343 ;
              0344 ;
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  46        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'
              0356
F18D  29F3      0357         DEFW    SWITCH      ;SWITCH CONSOLE OUTPUT VECTOR
F18F  05F2      0358         DEFW    MEMDMP      ;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  D8F2      0362         DEFW    FILL        ;FILL MEMORY
F199  8CF2      0363         DEFW    TEST        ;RAM DIAGNOSTIC
F19B  81F2      0364         DEFW    GOTO        ;JUMP TO MEMORY LOCATION
F19D  FEF2      0365         DEFW    INCMD       ;READ FROM INPUT PORT
```

```
F19F  20F3      03[redacted]      DEFW    OUTCMD      ;WRITE TO OUTPUT PORT
F1A1  BDF1      03[redacted]      DEFW    DSKCMD      ;DISPLAY DISK SECTOR DATA
                0368
>0021           0369 CMDSIZ  EQU     $-CMDTAB
                0370 ;
                0371 ;
                0372 ;*****************************************************
                0373 ;*                                                   *
                0374 ;*      MONITOR COMMAND ACTION ROUTINES PACKAGE      *
                0375 ;*                                                   *
                0376 ;*****************************************************
                0377 ;
                0378 ;
                0379 ;
                0380 ;
                0381 ;
                0382 ;-- DISK BOOT LOADER COMMAND --
                0383 ;
F1A3  0E00      0384 BOOT:   LD      C,0         ;SELECT DRIVE 0 FOR BOOT LOAD
F1A5  CDB1F6    0385         CALL    SELECT
F1A8  203D      0386         JR      NZ,DSKERR-$
F1AA  CDE9F6    0387         CALL    HOME        ;HOME HEAD TO TRACK 0
F1AD  2038      0388         JR      NZ,DSKERR-$;ERROR IF NOT READY OR AT TRO
F1AF  218000    0389         LD      HL,0080H    ;POINT TO CP/M READ BUFFER
F1B2  0E01      0390         LD      C,1         ;SELECT SECTOR 1
F1B4  CD2AF7    0391         CALL    READ        ;READ TRACK 0/ SECTOR 1
F1B7  202E      0392         JR      NZ,DSKERR-$
F1B9  F1        0393         POP     AF          ;CLEAN UP STACK
F1BA  C38000    0394         JP      0080H       ;GO EXECUTE LOADER
                0395 ;
                0396 ;
                0397 ;-- DISK SECTOR READ COMMAND --
                0398 ;
F1BD  FE03      0399 DSKCMD: CP      3           ;CHECK PARAMETER COUNT
F1BF  37        0400         SCF
F1C0  C0        0401         RET     NZ
F1C1  4D        0402         LD      C,L         ;USE FIRST ARG AS UNIT#
F1C2  CDB1F6    0403         CALL    SELECT
F1C5  2020      0404         JR      NZ,DSKERR-$
F1C7  217EFF    0405         LD      HL,PARAM2
F1CA  4E        0406         LD      C,(HL)      ;USE SECOND ARG AS TRACK#
F1CB  CDFBF6    0407         CALL    SEEK
F1CE  2017      0408         JR      NZ,DSKERR-$
F1D0  2180FF    0409         LD      HL,PARAM3
F1D3  4E        0410         LD      C,(HL)      ;USE THIRD ARG AS SECTOR#
F1D4  218000    0411         LD      HL,0080H
F1D7  CD2AF7    0412         CALL    READ
F1DA  CBC7      0413         SET     0,A         ;MARK ERROR BYTE AS DUE TO READ
F1DC  2009      0414         JR      NZ,DSKERR-$
F1DE  218000    0415         LD      HL,0080H
F1E1  110800    0416         LD      DE,8
F1E4  C327F2    0417         JP      DUMP        ;DUMP DISK READ BUFFER & RETURN
                0418 ;
                0419 ;
F1E7  4F        0420 DSKERR: LD      C,A         ;SAVE 1771 STATUS
F1E8  CDECF3    0421         CALL    PNEXT
F1EB  6469736B  0422         DEFM    'disk error '
      20657272
      6F7220
F1F6  04        0423         DEFB    EOT
F1F7  0608      0424         LD      B,8         ;PRINT 1771 ERROR BYTE IN BIN
F1F9  AF        0425 DSKR2:  XOR     A
F1FA  CB11      0426         RL      C
F1FC  CE30      0427         ADC     A,'0'       ;TRANSFORM A INTO ASCII'1'OR'0'
F1FE  CD15F4    0428         CALL    OUTPUT
F201  10F6      0429         DJNZ    DSKR2-$     ;REPEAT FOR 8 BITS
F203  B7        0430         OR      A
F204  C9        0431         RET
```

```
                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  2808      0440         JR      Z,MDMP3-$
F20B  2A86FF    0441 MDMP1:  LD      HL,(LAST)
F20E  111000    0442 MDMP2:  LD      DE,16
F211  180D      0443         JR      MDMP3B-$
                0444
F213  EB        0445 MDMP3:  EX      DE,HL
F214  ED52      0446         SBC     HL,DE       ;DERRIVE BYTECNT FOR DUMP RANGE
F216  0604      0447         LD      B,4
F218  CB3C      0448 MDMP3A: SRL     H           ;DIVIDE BYTECOUNT BY 16
F21A  CB1D      0449         RR      L
F21C  10FA      0450         DJNZ    MDMP3A-$
F21E  23        0451         INC     HL
F21F  EB        0452         EX      DE,HL
F220  CD27F2    0453 MDMP3B: CALL    DUMP        ;DUMP DE*16 BYTES STRTING AT HL
F223  2286FF    0454         LD      (LAST),HL
F226  C9        0455         RET
                0456 ;
                0457 ;
F227  E5        0458 DUMP:   PUSH    HL          ;SAVE STARTING ADDRESS
F228  CDCDF3    0459         CALL    PUT4HS      ;PRINT STARTING ADDRESS IN HEX
F22B  CD02F4    0460         CALL    SPACE
F22E  0610      0461         LD      B,16
F230  7E        0462 DUMP2:  LD      A,(HL)      ;GET A DATA BYTE @ HL
F231  23        0463         INC     HL
F232  CDD2F3    0464         CALL    PUT2HS      ;PRINT THE DATA IN HEX
F235  10F9      0465         DJNZ    DUMP2-$     ;REPEAT 16 TIMES
F237  E1        0466         POP     HL          ;RESTORE STARTING ADDRESS
F238  0610      0467         LD      B,16
F23A  7E        0468 DUMP3:  LD      A,(HL)      ;GET BACK DATA BYTE @ HL
F23B  23        0469         INC     HL
F23C  CBBF      0470         RES     7,A
F23E  FE20      0471         CP      20H
F240  3804      0472         JR      C,DUMP4-$
F242  FE7F      0473         CP      7FH
F244  3802      0474         JR      C,DUMP5-$
F246  3E2E      0475 DUMP4:  LD      A,'.'       ;PRINT DOT IF DATA < 20 OR > 7F
F248  CD15F4    0476 DUMP5:  CALL    OUTPUT      ;PRINT ASCII CHARACTER IN A
F24B  10ED      0477         DJNZ    DUMP3-$
F24D  CDFCF3    0478         CALL    CRLFS
F250  C0        0479         RET     NZ          ;EXIT IF ESCAPE REQ INDICATED
F251  1B        0480         DEC     DE
F252  7A        0481         LD      A,D
F253  B3        0482         OR      E
F254  20D1      0483         JR      NZ,DUMP-$
F256  C9        0484         RET
                0485 ;
                0486 ;
                0487 ;
                0488 ;
                0489 ;-- MEMORY EXAMINE COMMAND --
                0490 ;
F257  CDCEF2    0491 VIEW:   CALL    MDATA
F25A  CD07F4    0492         CALL    ECHO
F25D  FE0D      0493         CP      CR
F25F  281B      0494         JR      Z,VIEW4-$
F261  FE2D      0495         CP      '-'
F263  2819      0496         JR      Z,VIEW5-$
F265  CDBDF3    0497 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       VIEW-$
                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  CD80F1    0526           CALL     CALLX          ;CALL ADDRESS PASSED IN HL
F28A  B7        0527           OR       A
F28B  C9        0528           RET                     ;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            ;GET ENDING PAGE ADDRESS INTO E
F292  54        0539           LD       D,H            ;GET STARTING PAGE ADDR INTO D
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
F299  AC        0544           XOR      H              ;GENERATE TEST BYTE
F29A  A8        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-$
                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
F2A5  AC        0555           XOR      H              ;RE-GENERATE TEST BYTE DATA
F2A6  A8        0556           XOR      B
F2A7  CDB9F2    0557           CALL     CHECK          ;VERIFY MEMORY DATA STILL GOOD
F2AA  C0        0558           RET      NZ             ;EXIT IF ESC REQ IS INDICATED
```

```
F2AB  23        0559           INC     HL          ;ELSE GO O[illegible]D 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-$   ;DO ANOTHER PASS IF NO ESCAPE
F2B8  C9        0567           RET
                0568 ;
                0569 ;
                0570 ;
F2B9  BE        0571 CHECK:    CP      (HL)
F2BA  C8        0572           RET     Z           ;RETURN IF (HL)=A
F2BB  F5        0573           PUSH    AF
F2BC  CDCEF2    0574           CALL    MDATA       ;PRINT WHAT WAS ACTUALLY READ
F2BF  CDECF3    0575           CALL    PNEXT
F2C2  73686F75  0576           DEFM    'should='
      6C643D
F2C9  04        0577           DEFB    EOT
F2CA  F1        0578           POP     AF
F2CB  C3D2F3    0579           JP      PUT2HS      ;PRINT WHAT SHOULD HAVE READ
                0580 ;
                0581 ;
F2CE  CDFCF3    0582 MDATA:    CALL    CRLFS
F2D1  CDCDF3    0583           CALL    PUT4HS
F2D4  7E        0584           LD      A,(HL)
F2D5  C3D2F3    0585           JP      PUT2HS
                0586 ;
                0587 ;
                0588 ;
                0589 ;-- FILL MEMORY WITH CONSTANT COMMAND --
                0590 ;
F2D8  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  38F7      0600           JR      C,FILL1-$
F2E5  C9        0601           RET
                0602 ;
                0603 ;
                0604 ;
                0605 ;
                0606 ;-- MEMORY BLOCK MOVE COMMAND --
                0607 ;
F2E6  FE03      0608 BLOCK:    CP      3           ;CHECK IF PARAMETER COUNT=3
F2E8  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  C8        0614           RET     Z           ;EXIT NOW IF BC=0
F2F0  EDB0      0615           LDIR
F2F2  C9        0616           RET
                0617 ;
                0618 ;
                0619 ;
F2F3  EB        0620 BLOCAD:   EX      DE,HL
F2F4  B7        0621           OR      A           ;CLEAR CARRY
F2F5  ED52      0622           SBC     HL,DE       ;GET DIFFRENCE BETWEEN
```

```
F2F7  EB        0625            EX      DE,HL       ;HL & DE FOR BYTECOUNT
F2F8  D5        0626            PUSH    DE
F2F9  C5        0627            PUSH    BC
F2FA  D1        0628            POP     DE          ;GET OLD BC INTO DE
F2FB  C1        0629            POP     BC
F2FC  03        0630            INC     BC          ;GET COUNT+1 INTO BC
F2FD  C9        0631            RET
                0632 ;
                0633 ;
                0634 ;
                0635 ;-- READ FROM INPUT PORT COMMAND --
                0636 ;
F2FE  3D        0637 INCMD:     DEC     A           ;CHECK IF PARAMETER COUNT=1
F2FF  37        0638            SCF
F300  C0        0639            RET     NZ
F301  4D        0640            LD      C,L         ;POINT C TO INPUT PORT
F302  CDFCF3    0641 IN1:       CALL    CRLFS
F305  79        0642            LD      A,C
F306  CDD2F3    0643            CALL    PUT2HS
F309  ED78      0644            IN      A,(C)
F30B  CDD2F3    0645            CALL    PUT2HS
F30E  CD07F4    0646            CALL    ECHO
F311  FE0D      0647            CP      CR
F313  2806      0648            JR      Z,IN2-$
F315  FE2D      0649            CP      '-'
F317  2804      0650            JR      Z,IN3-$
F319  B7        0651            OR      A
F31A  C9        0652            RET
                0653 ;
F31B  0C        0654 IN2:       INC     C
F31C  0C        0655            INC     C
F31D  0D        0656 IN3:       DEC     C
F31E  18E2      0657            JR      IN1-$
                0658 ;
                0659 ;
                0660 ;
                0661 ;-- WRITE TO OUTPUT PORT COMMAND --
                0662 ;
F320  FE02      0663 OUTCMD:    CP      2           ;CHECK IF PARAMETER COUNT=2
F322  37        0664            SCF
F323  C0        0665            RET     NZ
F324  4D        0666            LD      C,L         ;POINT C TO OUTPUT PORT
F325  ED59      0667            OUT     (C),E       ;OUTPUT DATA PASSED IN E
F327  B7        0668            OR      A
F328  C9        0669            RET
                0670 ;
                0671 ;
                0672 ;-- SWITCH CONSOLE OUTPUT DEVICE COMMAND --
                0673 ;
F329  2185FF    0674 SWITCH:    LD      HL,COFLAG
F32C  34        0675            INC     (HL)        ;TOGGLE CONSOLE OUT TYPE FLAG
F32D  CB46      0676            BIT     0,(HL)
F32F  21FEF4    0677            LD      HL,SIOOUT
F332  2803      0678            JR      Z,SWIT2-$   ;JUMP IF ZERO TO ONE TRANSITION
F334  2120F5    0679            LD      HL,CRTOUT
F337  220DF0    0680 SWIT2:     LD      (CONOUT+1),HL ;STORE NEW CNSL OUT ADDR
F33A  C9        0681            RET
                0682 ;
                0683 ;
                0684 ;**************************************************************
                0685 ;*                                                            *
                0686 ;*       CONSOLE I/O PACKAGE AND UTILITY ROUTINES             *
                0687 ;*                                                            *
                0688 ;**************************************************************
                0689 ;
                0690 ;
```

```
                    0690 ;
F33B  41            0691 GETLIN: LD      B,C          ;SAVE MAX LINE LNGTH PARAM IN B
F33C  CD07F4        0692 GLIN1:  CALL    ECHO         ;GET A CHAR FROM THE CONSOLE
F33F  FE0D          0693         CP      CR           ;CHECK FOR CARRIAGE RETURN
F341  280E          0694         JR      Z,GLIN2-$
F343  FE08          0695         CP      'H'-64       ;CHECK FOR CTL-H BACKSPACE
F345  280C          0696         JR      Z,GLIN4-$
F347  FE20          0697         CP      ' '
F349  D8            0698         RET     C            ;OTHER CONT CHARACTERS ILLEGAL
F34A  77            0699         LD      (HL),A
F34B  23            0700         INC     HL           ;STORE CHARACTER IN BUFFER
F34C  0D            0701         DEC     C
F34D  20ED          0702         JR      NZ,GLIN1-$   ;GET ANOTHER IF MORE ROOM
F34F  37            0703         SCF
F350  C9            0704         RET                  ;RETURN WITH CARRY=1 IF TOO
                    0705                              ;MANY CHARACTERS ARE ENTERED
F351  77            0706 GLIN2:  LD      (HL),A       ;PUT <CR> ON END OF LINE
F352  C9            0707         RET                  ;RETURN WITH CARRY BIT=0
                    0708
F353  2B            0709 GLIN4:  DEC     HL           ;DELETE LAST CHAR FROM BUFFER
F354  CDECF3        0710         CALL    PNEXT
F357  2008          0711         DEFB    ' ','H'-64   ;PRINT A SPACE TO OVERWRITE THE
F359  04            0712         DEFB    EOT          ;LAST CHAR,THEN DO A BACKSPACE
F35A  0C            0713         INC     C
F35B  78            0714         LD      A,B          ;MAKE SURE YOU'RE NOT TRYING TO
F35C  91            0715         SUB     C            ;<BS> PAST START OF THE LINE
F35D  30DD          0716         JR      NC,GLIN1-$
F35F  C9            0717         RET
                    0718 ;
                    0719 ;
                    0720 ;
F360  EDB1          0721 SEARCH: CPIR                 ;SRCH TBL @HL FOR MATCH WITH A
F362  C0            0722         RET     NZ           ;EXIT NOW IF SEARCH FAILS
F363  09            0723         ADD     HL,BC
F364  09            0724         ADD     HL,BC        ;+ RESIDUE FROM CPIR BYTECOUNT
F365  09            0725         ADD     HL,BC        ;TO HL 3 TIMES TO GET POINTER
F366  4E            0726         LD      C,(HL)       ;TO ADDRESS PART OF TABLE ENTRY
F367  23            0727         INC     HL
F368  46            0728         LD      B,(HL)
F369  C9            0729         RET                  ;EXIT WITH Z=1 TO SHOW MATCH
                    0730 ;
                    0731 ;
                    0732 ;
                    0733 ;
F36A  010000        0734 PARAMS: LD      BC,0
F36D  FD7E00        0735         LD      A,(IY+0)
F370  FE0D          0736         CP      CR           ;CHECK IF LINE TERMINATES
F372  2008          0737         JR      NZ,PARA2-$   ; IMMEDIATELY WITH A RETURN
F374  AF            0738         XOR     A
F375  C9            0739         RET                  ;RET WITH PARAM COUNT=0 IF SO
                    0740
F376  0C            0741 PARA1:  INC     C
F377  0C            0742         INC     C
F378  CB59          0743         BIT     3,C
F37A  37            0744         SCF
```

```
F37B  C0        0745          RET     NZ          ;ERROR IF > 4 NUMBERS ENTERED
F37C  C5        0746 PARA2:   PUSH    BC          ;SAVE PARAMETER COUNT
F37D  CD9FF3    0747          CALL    GETHEX      ;READ A NUMBER FROM LINE BUFFER
F380  C1        0748          POP     BC
F381  D8        0749 PARA4:   RET     C           ;ERROR IF RESULT OVER 16 BITS
F382  DD217CFF  0750          LD      IX,PARAM1   ;POINT TO PARAM STORAGE AREA
F386  DD09      0751          ADD     IX,BC       ;ADD PARAMETER COUNT IN BC
F388  DD7500    0752          LD      (IX+0),L
F38B  DD7401    0753          LD      (IX+1),H    ;STORE DATA RET FROM 'GETHEX'
F38E  FE20      0754          CP      ' '
F390  28E4      0755          JR      Z,PARA1-$   ;GET ANOTHER ITEM IF SPACE
F392  FE2C      0756          CP      ','
F394  28E0      0757          JR      Z,PARA1-$   ;GET ANOTHER ITEM IF COMMA
F396  FE0D      0758          CP      CR
F398  37        0759          SCF                 ;ELSE CHECK FOR CARRIAGE RETURN
F399  C0        0760          RET     NZ          ; AND EXIT WITH CY=1 IF NOT
F39A  79        0761 PAREND:  LD      A,C
```

```
F39B  CB3F      0762            SRL     A           ;A=COUNT OF NUMBERS ENTERED
F39D  3C        0763            INC     A
F39E  C9        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 ;
F39F  210000    0772 GETHEX: LD      HL,0
F3A2  180B      0773          JR      GNUM3-$
                0774
F3A4  0604      0775 GNUM1:  LD      B,4
F3A6  29        0776 GNUM2:  ADD     HL,HL       ;MULTIPLY RESULT BY 16
F3A7  D8        0777         RET     C           ;RETURN IF IT OVERFLOWS 16 BITS
F3A8  10FC      0778         DJNZ    GNUM2-$
F3AA  5F        0779         LD      E,A         ;APPEND NEW LOW ORDER DIGIT
F3AB  1600      0780         LD      D,0         ;AND GET RESULT BACK INTO DE
F3AD  19        0781         ADD     HL,DE
F3AE  D8        0782         RET     C           ;RETURN IF OVERFLOW
F3AF  FD7E00    0783 GNUM3:  LD      A,(IY+0)    ;GET A CHAR FROM LINE INPUT
F3B2  FD23      0784         INC     IY          ; BUFFER @ IY AND BUMP IY
F3B4  4F        0785         LD      C,A
F3B5  CDBDF3    0786         CALL    ASCHEX      ;CONVERT ASCII TO NUMERIC
F3B8  30EA      0787         JR      NC,GNUM1-$
F3BA  79        0788         LD      A,C
F3BB  B7        0789         OR      A
F3BC  C9        0790         RET
                0791 ;
                0792 ;
F3BD  D630      0793 ASCHEX: SUB     '0'
F3BF  D8        0794         RET     C
F3C0  FE0A      0795         CP      10
F3C2  3F        0796         CCF
F3C3  D0        0797         RET     NC
F3C4  D607      0798         SUB     7
F3C6  FE0A      0799         CP      10
F3C8  D8        0800         RET     C
F3C9  FE10      0801         CP      16
F3CB  3F        0802         CCF
F3CC  C9        0803         RET
                0804 ;
                0805 ;
                0806 ;
F3CD  7C        0807 PUT4HS: LD      A,H
F3CE  CDD8F3    0808         CALL    PUT2HX
F3D1  7D        0809         LD      A,L
F3D2  CDD8F3    0810 PUT2HS: CALL    PUT2HX
F3D5  C302F4    0811         JP      SPACE
                0812 ;
                0813 ;
F3D8  F5        0814 PUT2HX: PUSH    AF
F3D9  1F        0815         RRA
F3DA  1F        0816         RRA
F3DB  1F        0817         RRA
F3DC  1F        0818         RRA
F3DD  CDE1F3    0819         CALL    PUTNIB
F3E0  F1        0820         POP     AF
F3E1  E60F      0821 PUTNIB: AND     00001111B
```

```
F3E3  C690      0822            ADD     A,90H
F3E5  27        0823            DAA
F3E6  CE40      0824            ADC     A,40H
F3E8  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  18F6      0848            JR      PMSG-$
                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 ;
                0861 ;ECHO INPUTS ONE CHARACTER FROM THE CONSOLE
                0862 ;DEVICE, PRINTS IT ON THE CONSOLE OUTPUT AND
                0863 ;THEN RETURNS IT IN REGISTER A WITH BIT 7 RESET
                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
F418  CD06F0    0882            CALL    CONST       ;SEE IF CONSOLE INPUT PENDING
F41B  280F      0883            JR      Z,OUTP2-$
F41D  CD09F0    0884            CALL    CONIN
F420  FE0D      0885            CP      CR          ;SEE IF <CR> WAS TYPED
F422  2805      0886            JR      Z,OUTP1-$
F424  CD09F0    0887            CALL    CONIN       ;WAIT FOR ANOTHER INPUT CHAR
F427  1803      0888            JR      OUTP2-$     ; THEN RET TO CALLING ROUTINE
                0889
F429  3284FF    0890 OUTP1:     LD      (ESCFLG),A  ;SET ESC FLAG TO NON-ZERO VALUE
F42C  3A84FF    0891 OUTP2:     LD      A,(ESCFLG)
F42F  B7        0892            OR      A           ;RETURN CURRENT STATUS OF ESC
F430  C9        0893            RET                 ; FLAG TO CALLING ROUTINE
                0894 ;
                0895 ;
                0896 ;
                0897            INCLUDE INTSRV.ASM
```

```
                0898 ;*************************************************************
                0899 ;*                                                           *
                0900 ;*       INTERRUPT SERVICE ROUTINES FOR KEYBOARD             *
                0901 ;*       INPUT AND REAL-TIME CLOCK FUNCTIONS                 *
                0902 ;*                                  3-Aug-80                 *
                0903 ;*                                                           *
                0904 ;*************************************************************
                0905 ;
                0906 ;
                0907 ;
                0908 ;
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
                0914 ;
                0915 ;
                0916 ;
F439  CD31F4    0917 KBDIN:  CALL    KBDST
F43C  28FB      0918         JR      Z,KBDIN-$  ;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
                0923 ;
                0924 ;
                0925 ;
                0926 ;
                0927 ;
F444  2133FF    0928 STASH:  LD      HL,LOCK    ;POINT TO SHIFT LOCK VARIABLES
F447  BE        0929         CP      (HL)       ;TEST IF A=SHIFT LOCK CHARACTER
F448  23        0930         INC     HL         ;THEN POINT TO LOCK FLAG
F449  2002      0931         JR      NZ,STASH2-$;JUMP IF NOT SHIFT CHARACTER
F44B  34        0932         INC     (HL)       ;ELSE COMPLIMENT THE SHIFT LOCK
F44C  C9        0933         RET                ;AND EXIT NOW
                0934
F44D  CB46      0935 STASH2: BIT     0,(HL)     ;TEST THE SHIFT LOCK FLAG
F44F  280A      0936         JR      Z,STASH3-$ ;JUMP IF SHIFT LOCK NOT SET
F451  FE40      0937         CP      40H        ;ELSE CHECK FOR SHIFTABLE CHAR
F453  3806      0938         JR      C,STASH3-$ ;AND JUMP IF NOT = OR GREATER
F455  FE7F      0939         CP      7FH        ;THAN '@' AND LESS THAN RUBOUT
F457  3002      0940         JR      NC,STASH3-$
F459  EE20      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  FE10      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
                0953 ;
                0954 ;
                0955 ;
                0956 ;
```

```
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  85        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 ;
                0970 ;
                0971 ;SOFTWARE DISK MOTOR TURN-OFF TIMER ROUTINE
                0972 ;
F480  216CFF    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
                0980 ;
                0981 ;
                0982 ;-- INTERRUPT SERVICE ROUTINE FOR PARALLEL KEYBOARD --
                0983 ;
F48C  ED7335FF  0984 KEYSRV: LD      (SPSAVE),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
                0994 ;
                0995 ;
                0996 ;
                0997 ;-- INTERRUPT SERVICE ROUTINE FOR ONE SECOND TIMER --
                0998 ;
F49F  ED7335FF  0999 TIMER:  LD      (SPSAVE),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
F4AA  2A57FF    1005         LD      HL,(TIKVEC);GET CLOCK INTERRUPT RTN VECTOR
F4AD  1812      1006         JR      DSPTCH-$   ; AND JUMP TO DISPATCH POINT
                1007 ;
                1008 ;
                1009 ;
                1010 ;-- SERIAL INPUT INTERRUPT SERVICE ROUTINE FOR SIO --
                1011 ;
F4AF  ED7335FF  1012 SIOINT: LD      (SPSAVE),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
```

```
F4C1  CDE7F4     1021 DSPTCH: CALL    CALLHL      ;CALL SUBROUTINE ADDRESSED BY H
F4C4  F1         1022         POP     AF
F4C5  C1         1023         POP     BC
F4C6  D1         1024         POP     DE
F4C7  E1         1025         POP     HL
F4C8  ED7B35FF   1026         LD      SP,(SPSAVE)
F4CC  FB         1027         EI                  ;RE-ENABLE INTERRUPTS & RETURN
F4CD  ED4D       1028         RETI
                 1029 ;
                 1030 ;
                 1031 ;-- RX ERROR INTERRUPT SERVICE ROUTINE FOR SIO --
                 1032 ;
                 1033 ;ARRIVE HERE IF RECEIVE INTERRUPT FROM FRAMING, OVERRUN
                 1034 ;AND PARITY ERRORS. (PARITY CAN BE DISABLED)
                 1035 ;
F4CF  ED7335FF   1036 SIOERR: LD      (SPSAVE),SP  ;SAVE USER STACK POINTER AND
F4D3  3157FF     1037         LD      SP,TMPSTK+32 ; SWITCH TO LOCAL STACK
F4D6  F5         1038         PUSH    AF
F4D7  CDF5F4     1039         CALL    SIOIN2      ;CLEAR BAD CHARACTER FROM SIO
F4DA  3E07       1040         LD      A,'G'-64
F4DC  CD15F5     1041         CALL    SIOXMT      ;OUTPUT A CTL-G AS A WARNING
F4DF  F1         1042         POP     AF
F4E0  ED7B35FF   1043         LD      SP,(SPSAVE)
F4E4  FB         1044         EI
F4E5  ED4D       1045         RETI
                 1046 ;
                 1047 ;
F4E7  E9         1048 CALLHL: JP      (HL)
                 1049 ;
                 1050 ;
                 1051 ;
                 1052 ;POLLED MODE I/O ROUTINES FOR SIO CHANEL B
                 1053 ;
F4E8  DB07       1054 SIOST:  IN      A,(SIOCPB) ;GET SIO STATUS REGISTER
F4EA  E601       1055         AND     00000001B
F4EC  C8         1056         RET     Z          ;ACC=0 IF NO DATA AVAILABLE
F4ED  3EFF       1057         LD      A,255
F4EF  C9         1058         RET
                 1059 ;
                 1060 ;
F4F0  CDE8F4     1061 SIOIN:  CALL    SIOST      ;TEST CONSOLE STATUS
F4F3  28FB       1062         JR      Z,SIOIN-$  ;LOOP UNTIL DATA IS RECEIVED
R4F5  3E30       1063 SIOIN2: LD      A,00110000B     ;RESET STATUS BITS IN SIO FO
F4F7  D307       1064         OUT     (SIOCPB),A ;PARITY/OVERRUN/FRAMING ERRORS,
F4F9  DB05       1065         IN      A,(SIODPB) ;THEN GET THE INPUT CHARACTER
F4FB  E67F       1066         AND     01111111B
F4FD  C9         1067         RET
                 1068 ;
                 1069 ;
F4FE  FE20       1070 SIOOUT: CP      ' '        ;TEST FOR CONTROL CHARACTERS
F500  3013       1071         JR      NC,SIOXMT-$;JUMP IF PRINTABLE CHARACTER
F502  CD15F5     1072         CALL    SIOXMT     ;ELSE SEND CONTROL CHARACTER
F505  3A79FF     1073         LD      A,(NULLS)  ;AND THEN SEND NULLS AS PADDING
F508  3C         1074         INC     A          ;GET NULL PAD COUNT AND FIX SO
F509  1806       1075         JR      PAD1-$     ;THAT COUNT=0 SENDS NO NULLS
                 1076
F50B  F5         1077 PAD:    PUSH    AF
F50C  AF         1078         XOR     A
F50D  CD15F5     1079         CALL    SIOXMT     ;OUTPUT A NULL TO THE SIO
F510  F1         1080         POP     AF
F511  3D         1081 PAD1:   DEC     A
F512  20F7       1082         JR      NZ,PAD-$   ;LOOP SENDING NULLS TO SIO
F514  C9         1083         RET
                 1084 ;
                 1085 ;
F515  F5         1086 SIOXMT: PUSH    AF
F516  DB07       1087 SIOX1:  IN      A,(SIOCPB)
```

```
F518  E604      1088          AND     00000100B  ;TEST TBE STATUS BIT
F51A  28FA      1089          JR      Z,SIOX1-$
F51C  F1        1090          POP     AF
F51D  D305      1091          OUT     (SIODPB),A ;OUTPUT DATA TO SIO
F51F  C9        1092          RET
                1093 ;
                1094 ;
                1095 ;
                1096 ;
                1097          INCLUDE CRTOUT.ASM
                1098 ;*******************************************************
                1099 ;*                                                     *
                1100 ;*       MEMORY-MAPPED CRT OUTPUT DRIVER               *
                1101 ;*                                                     *
                1102 ;*       Russell Smith     18-August-1980              *
                1103 ;*                                                     *
                1104 ;*******************************************************
                1105 ;
                1106 ;
>0030           1107 CRTBAS   EQU     CRTMEM.SHR.8 ;START PAGE# OF 3K CRT SPACE
>003C           1108 CRTTOP   EQU     CRTMEM+3072.SHR.8 ;END PAGE# OF CRT SPACE
                1109 ;
                1110 ;
F520  E5        1111 CRTOUT:  PUSH    HL
F521  D5        1112          PUSH    DE
F522  C5        1113          PUSH    BC
F523  CBBF      1114          RES     7,A
F525  4F        1115          LD      C,A
F526  F3        1116          DI                 ;KEEP WOLVES AWAY FOR A WHILE
F527  ED7335FF  1117          LD      (SPSAVE),SP
F52B  3157FF    1118          LD      SP,TMPSTK+32 ;POINT SP TO TOP LOCAL STACK
F52E  DB1C      1119          IN      A,(BITDAT)
F530  CBFF      1120          SET     7,A        ;SELECT ROM/CRT MEMORY BANK
F532  D31C      1121          OUT     (BITDAT),A
                1122 ;
                1123 ;FIRST REMOVE THE OLD CURSOR CHARACTER FROM THE SCREEN
                1124 ;
F534  2175FF    1125          LD      HL,CHRSAV  ;GET CHAR OVERLAYED BY CURSOR
F537  46        1126          LD      B,(HL)
F538  2A73FF    1127          LD      HL,(CURSOR);LOAD HL WITH CURSOR POINTER
F53B  7C        1128          LD      A,H
F53C  E60F      1129          AND     00001111B  ;INSURANCE THAT HL CAN'T
F53E  F630      1130          OR      CRTBAS     ;EVER POINT OUTSIDE CRT MEMORY
F540  67        1131          LD      H,A
F541  70        1132          LD      (HL),B     ;RMV CURSOR BY RESTORING CHAR
                1133 ;
                1134 ;PROCESS CHARACTER PASSED IN C
                1135 ;
F542  CD65F5    1136          CALL    OUTCH
                1137 ;
                1138 ;NOW STORE A NEW CURSOR CHARACTER AT THE CURSOR LOCATION
                1139 ;
F545  7E        1140          LD      A,(HL)     ;GET CHAR AT NEW CURSOR LOCAT.
F546  3275FF    1141          LD      (CHRSAV),A ;SAVE FOR NEXT TIME 'CRTOUT' IS
;                                                  CALLED
F549  FE20      1142          CP      ' '        ;TEST IF CHARACTER IS A SPACE
F54B  CBFF      1143          SET     7,A        ;THEN TURN ON BIT 7 TO ENABLE
;                                                  BLINK
F54D  2003      1144          JR      NZ,CRT2-$  ;JUMP IF CHARACTER IS NON-BLANK
F54F  3A76FF    1145          LD      A,(CSRCHR) ;ELSE GET CHAR USED FOR CURSOR
F552  77        1146 CRT2:    LD      (HL),A     ;STORE CHAR IN A AS CURSOR MARK
F553  2273FF    1147          LD      (CURSOR),HL;SAVE HL AS CURSOR POINTER
                1148
F556  ED7B35FF  1149          LD      SP,(SPSAVE)
F55A  DB1C      1150          IN      A,(BITDAT)
```

```
F55C  CBBF      1151            RES     7,A             ;SWITCH BACK LOWER 16K OF RAM
F55E  D31C      1152            OUT     (BITDAT),A
F560  FB        1153            EI                      ;INTERRUPTS ARE SAFE AGAIN
F561  C1        1154            POP     BC
F562  D1        1155            POP     DE
F563  E1        1156            POP     HL
F564  C9        1157            RET
                1158 ;
                1159 ;
                1160 ;
F565  1178FF    1161 OUTCH:     LD      DE,LEADIN
F568  1A        1162            LD      A,(DE)          ;GET LEAD-IN SEQUENCE STATE
F569  B7        1163            OR      A
F56A  C270F6    1164            JP      NZ,MULTI        ;JUMP IF IN A LEAD-IN SEQUENCE
F56D  79        1165            LD      A,C             ; ELSE PROCESS CHARACTER IN C
F56E  FE20      1166            CP      ' '
F570  380F      1167            JR      C,CONTRL-$      ;JUMP IF A CONTROL CHARACTER
F572  71        1168 DISPLA:    LD      (HL),C          ;ELSE STORE DISPLAYABLE CHAR
F573  23        1169            INC     HL              ;AND ADV POINTER TO NEXT COLUMN
F574  7D        1170            LD      A,L
F575  E67F      1171            AND     01111111B       ;EXTRACT COLUMN# FROM HL
F577  FE50      1172            CP      80
F579  D8        1173            RET     C               ;EXIT IF NOT PAST COLUMN 79
F57A  CDE7F5    1174            CALL    RETURN          ;ELSE DO AUTOMATIC <CR>
F57D  CD42F6    1175            CALL    LFEED           ;AND LINEFEED
F580  C9        1176            RET
                1177 ;
                1178 ;
                1179 ;
F581  E5        1180 CONTRL:    PUSH    HL
F582  218FF5    1181            LD      HL,CTLTAB       ;SEARCH FOR CONTROL CHARACTER
F585  010D00    1182            LD      BC,CTLSIZ/3     ;HANDLING SUBROUTINE IN TABLE
F588  CD60F3    1183            CALL    SEARCH
F58B  E1        1184            POP     HL
F58C  C0        1185            RET     NZ              ;EXIT IF NOT IMPLEMENTED
F58D  C5        1186            PUSH    BC
F58E  C9        1187            RET                     ;DO SNEAKY JUMP TO PRESERVE
;                                                        REGISTERS
                1188
F58F  1F        1189 CTLTAB:    DEFB    '_'-64
F590  1E        1190            DEFB    '^'-64
F591  1B        1191            DEFB    '['-64
F592  1A        1192            DEFB    'Z'-64
F593  18        1193            DEFB    'X'-64
F594  11        1194            DEFB    'Q'-64
F595  0D        1195            DEFB    'M'-64
F596  0C        1196            DEFB    'L'-64
F597  0B        1197            DEFB    'K'-64
F598  0A        1198            DEFB    'J'-64
F599  09        1199            DEFB    'I'-64
F59A  08        1200            DEFB    'H'-64
F59B  07        1201            DEFB    'G'-64
                1202
F59C  DCF5      1203            DEFW    BELL            ;CTL-G IS THE BELL
F59E  BEF5      1204            DEFW    BAKSPC          ;CTL-H IS CURSOR LEFT
F5A0  CCF5      1205            DEFW    TAB             ;CTL-I IS TAB
F5A2  42F6      1206            DEFW    LFEED           ;CTL-J IS CURSOR DOWN
F5A4  2CF6      1207            DEFW    UPCSR           ;CTL-K IS CURSOR UP
F5A6  C4F5      1208            DEFW    FORSPC          ;CTL-L IS CURSOR RIGHT
```

```
F5A8  E7F5      1209           DEFW    RETURN      ;CTL-M IS <CR>
F5AA  11F6      1210           DEFW    CLREOS      ;CTL-Q CLEAR TO END-OF-SCREEN
F5AC  03F6      1211           DEFW    CLREOL      ;CTL-X IS CLEAR TO END-OF-LINE
F5AE  ECF5      1212           DEFW    CLRSCN      ;CTL-Z IS CLEAR SCREEN
F5B0  B6F5      1213           DEFW    ESCAPE      ;CTL-[ IS ESCAPE
F5B2  6CF6      1214           DEFW    HOMEUP      ;CTL-^ IS HOME UP
F5B4  BAF5      1215           DEFW    STUFF       ;CTL-_ IS DISPLAY CONTROL CHARS
                1216
>0027           1217 CTLSIZ    EQU     $-CTLTAB
                1218 ;
                1219 ;
F5B6  3E01      1220 ESCAPE:   LD      A,1
F5B8  12        1221           LD      (DE),A      ;SET LEAD-IN SEQUENCE STATE
F5B9  C9        1222           RET                 ;FOR XY CURSOR POSITIONING MODE
                1223 ;
                1224 ;
F5BA  3E04      1225 STUFF:    LD      A,4
F5BC  12        1226           LD      (DE),A      ;SET LEAD-IN SEQUENCE STATE
F5BD  C9        1227           RET                 ;FOR CONTROL CHAR OUTPUT MODE
                1228 ;
                1229 ;
F5BE  7D        1230 BAKSPC    LD      A,L         ;CHECK FOR LEFT MARGIN
F5BF  E67F      1231           AND     01111111B
F5C1  C8        1232           RET     Z           ;ABORT IF IN LEFTMOST COLUMN
F5C2  2B        1233           DEC     HL          ;BACK UP CURSOR POINTER
F5C3  C9        1234           RET
                1235 ;
                1236 ;
F5C4  7D        1237 FORSPC:   LD      A,L         ;CHECK FOR RIGHTMOST COLUNM
F5C5  E67F      1238           AND     01111111B
F5C7  FE4F      1239           CP      79
F5C9  D0        1240           RET     NC          ;DO NOTHING IF ALREADY THERE
F5CA  23        1241           INC     HL
F5CB  C9        1242           RET                 ;ELSE ADVANCE CURSOR POINTER
                1243 ;
                1244 ;
F5CC  110800    1245 TAB:      LD      DE,8        ;TABS ARE EVERY 8 COLUMNS
F5CF  7D        1246           LD      A,L         ;GET COLUMN COMPONENT OF
F5D0  E678      1247           AND     01111000B   ; PREVIOUS TAB POSITION
F5D2  83        1248           ADD     A,E
F5D3  FE50      1249           CP      80          ;EXIT IF NEXT TAB COLUMN WOULD
F5D5  D0        1250           RET     NC          ; BE PAST THE RIGHT MARGIN
F5D6  7D        1251           LD      A,L
F5D7  E6F8      1252           AND     11111000B   ;ELSE INCREMENT THE CURSOR
F5D9  6F        1253           LD      L,A         ; POINTER FOR REAL
F5DA  19        1254           ADD     HL,DE
F5DB  C9        1255           RET
                1256 ;
                1257 ;
F5DC  DB1C      1258 BELL:     IN      A,(BITDAT)
F5DE  CBEF      1259           SET     5,A         ;TOGGLE BIT 5 OF SYSTEM PIO TO
F5E0  D31C      1260           OUT     (BITDAT),A  ;TRIGGER BELL HARDWARE TO SOUND
F5E2  CBAF      1261           RES     5,A
F5E4  D31C      1262           OUT     (BITDAT),A
F5E6  C9        1263           RET
                1264 ;
                1265 ;
F5E7  7D        1266 RETURN:   LD      A,L
F5E8  E680      1267           AND     10000000B
F5EA  6F        1268           LD      L,A         ;MOVE CURSOR POINTER BACK
F5EB  C9        1269           RET                 ; TO START OF LINE
                1270 ;
                1271 ;
F5EC  210030    1272 CLRSCN:   LD      HL,CRTMEM
```

```
F5EF  E5        127[redacted]       PUSH    HL
F5F0  110130    1274        ,       LD      DE,CRTMEM+1
F5F3  01000C    1275                LD      BC,24*128
F5F6  3620      1276                LD      (HL),' '
F5F8  EDB0      1277                LDIR                    ;FILL CRT MEMORY WITH SPACES
F5FA  E1        1278                POP     HL              ;POINT TO HOME CURSOR POSITION
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
                1283 ;
                1284 ;
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 CHARS
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
                1295 ;                                      ''
                1296 ;
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
F61A  17        1302                RLA
F61B  7C        1303                LD      A,H
F61C  17        1304                RLA                     ;ROW# COMPONENT OF HL INTO A
F61D  E61F      1305                AND     00011111B
F61F  B9        1306                CP      C               ;SEE IF HL IS AT BOTTOM ROW
                                                             OF SCREEN
;
F620  2808      1307                JR      Z,CLRS2-$       ;AND LEAVE CLEAR LOOP IF SO
F622  CD37F6    1308                CALL    DNCSR           ;ELSE POINT HL TO NEXT ROW DOWN
F625  CD60F6    1309                CALL    CLRLIN          ;AND FILL THAT LINE WITH SPACES
F628  18EF      1310                JR      CLRS1-$
                1311
F62A  E1        1312 CLRS2:         POP     HL              ;RESTR ORIGINAL CURSOR POINTER
F62B  C9        1313                RET
                1314 ;
                1315 ;
F62C  1180FF    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      CRTBAS          ;CHECK FOR UNDERFLOW OF POINTER
F633  D0        1320                RET     NC
F634  263B      1321                LD      H,CRTTOP-1      ;WRAP CURSOR AROUND MODULO 3K
F636  C9        1322                RET
                1323 ;
                1324 ;
F637  118000    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      CRTTOP          ;CHECK FOR OVERFLOW OF POINTER
F63E  D8        1329                RET     C
F63F  2630      1330                LD      H,CRTBAS        ;RESET POINTER MODULO 128*24
F641  C9        1331                RET
                1332 ;
                1333 ;
                1334 ;
F642  7D        1335 LFEED:         LD      A,L
F643  17        1336                RLA
F644  7C        1337                LD      A,H
```

```
F645  17        1338            RLA                     ;EXTRACT ROW# COMPONENT OF HL
F646  E61F      1339            AND     00011111B
F648  4F        1340            LD      C,A             ;COPY ROW# TO C FOR SCROLL TEST
F649  CD37F6    1341            CALL    DNCSR           ;MOVE CURSOR TO NEXT ROW DOWN
F64C  3A77FF    1342            LD      A,(BASE)        ;TEST IF CURSOR ON BOTTOM ROW
F64F  B9        1343            CP      C               ;OF SCREEN BEFORE MOVING DOWN
F650  C0        1344            RET     NZ              ;EXIT IF NOT AT BOTTOM
                1345
F651  E5        1346            PUSH    HL              ;ELSE PREP TO SCROLL SCREEN UP
F652  CD60F6    1347            CALL    CLRLIN          ;FILL NEW BOTTOM LINE WTH SPACES
F655  29        1348            ADD     HL,HL
F656  7C        1349            LD      A,H             ;GET ROW# PART OF HL INTO A
F657  E61F      1350            AND     00011111B
F659  3277FF    1351            LD      (BASE),A        ;STORE NEW BASE LINE#
F65C  D314      1352            OUT     (SCROLL),A      ;SCROLL UP NEW BLANK BOTTM LINE
F65E  E1        1353            POP     HL
F65F  C9        1354            RET
                1355 ;
                1356 ;
F660  7D        1357 CLRLIN:    LD      A,L
F661  E680      1358            AND     10000000B       ;POINT HL TO 1ST COLUMN OF ROW
F663  6F        1359            LD      L,A
F664  0650      1360            LD      B,80
F666  3620      1361 CLR:       LD      (HL),' '        ;STORE ASCII SPACES AT ADDR
                                                         IN HL
;
F668  23        1362            INC     HL              ;AND INCREMENT HL
F669  10FB      1363            DJNZ    CLR-$           ;REPEAT NUMBER OF TIMES IN B
F66B  C9        1364            RET
                1365 ;
                1366 ;
F66C  0E20      1367 HOMEUP:    LD      C,' '           ;FAKE-OUT CURSOR ADDR ROUTINE
F66E  1817      1368            JR      SETROW-$        ;TO DO HOMEUP ALMOST FOR FREE
                1369 ;
                1370 ;
F670  EB        1371 MULTI:     EX      DE,HL           ;UNCONDITIONALLY RESET LEAD-IN
F671  3600      1372            LD      (HL),0          ;STATE TO ZERO BEFORE GOING ON
F673  EB        1373            EX      DE,HL
F674  FE01      1374            CP      1
F676  2008      1375            JR      NZ,M2TST-$
F678  79        1376 SETXY:     LD      A,C             ;GET SECOND CHAR OF SEQUENCE
F679  FE3D      1377            CP      '='
F67B  C0        1378            RET     NZ              ;ABORT SEQUENCE IF NOT '='
F67C  3E02      1379            LD      A,2
F67E  12        1380            LD      (DE),A          ;MAKE LEADIN=2 NEXT TIME
F67F  C9        1381            RET
                1382
F680  FE02      1383 M2TST:     CP      2
F682  2019      1384            JR      NZ,M3TST-$
F684  3E03      1385            LD      A,3
F686  12        1386            LD      (DE),A          ;MAKE LEADIN=3 NEXT TIME
F687  3A77FF    1387 SETROW:    LD      A,(BASE)        ;ARRIVE HERE ON THIRD CHAR
F68A  81        1388            ADD     A,C             ; OF ESC,'=',ROW,COL SEQUENCE
F68B  D61F      1389            SUB     ' '-1
F68D  D618      1390 SETR2:     SUB     24
F68F  30FC      1391            JR      NC,SETR2-$      ;VERIFY ROW# BETWEEN 0 AND 23
F691  C618      1392            ADD     A,24
F693  F660      1393            OR      CRTMEM.SHR.7    ;MERGE IN MSB'S OF CRT MEMORY
F695  67        1394            LD      H,A
F696  2E00      1395            LD      L,0
F698  CB3C      1396            SRL     H
F69A  CB1D      1397            RR      L
F69C  C9        1398            RET
                1399
F69D  FE03      1400 M3TST:     CP      3
F69F  200C      1401            JR      NZ,M4TST-$
```

```
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     80
F6A6  30FC      1405         JR      NC,SETC2-$   ;MAKE SURE COL# BETWEEN 0 & 79
F6A8  C650      1406         ADD     A,80
F6AA  B5        1407         OR      L            ;MERGE IN COL# WITH L
F6AB  6F        1408         LD      L,A
F6AC  C9        1409         RET
                1410
F6AD  CD72F5    1411 M4TST:  CALL    DISPLA       ;DISPLAY THE CONTROL CHAR
F6B0  C9        1412         RET                  ;PASSED IN C
                1413 ;
                1414 ;
                1415 ;
                1416 ;
                1417         INCLUDE DISKIO.ASM
                1418 ;*****************************************************
                1419 ;*                                                   *
                1420 ;*      DISK INPUT/OUTPUT DRIVER SUBROUTINE PACKAGE  *
                1421 ;*      FOR WESTERN DIGITAL 1771 DISK CONTROLLER     *
                1422 ;*                                                   *
                1423 ;*      bullet-proof error recovery added 12-APR-80  *
                1424 ;*                                                   *
                1425 ;*****************************************************
                1426 ;
                1427 ;
                1428 ;EQUATES FOR DISK CONTROLLER PORTS AND COMMAND CODES
                1429 ;
>0010           1430 STSREG  EQU     WD1771+0     ;STATUS REGISTER
>0010           1431 CMDREG  EQU     WD1771+0     ;COMMAND REGISTER
>0011           1432 TRKREG  EQU     WD1771+1     ;TRACK REGISTER
>0012           1433 SECREG  EQU     WD1771+2     ;SECTOR REGISTER
>0013           1434 DATREG  EQU     WD1771+3     ;DATA REGISTER
                1435 ;
>0088           1436 RDCMD   EQU     10001000B    ;READ COMMAND
>00A8           1437 WRTCMD  EQU     10101000B    ;WRITE COMMAND
>001C           1438 SKCMD   EQU     00011100B    ;SEEK COMMAND
>00D0           1439 FINCMD  EQU     11010000B    ;FORCE INTR COMMAND
>000C           1440 RSTCMD  EQU     00001100B    ;RESTORE COMMAND
>0004           1441 HLOAD   EQU     00000100B    ;RD/WRT HEAD LOAD ENABLE
                1442 ;
>00C9           1443 RET     EQU     0C9H         ;SUBROUTINE RETURN INSTR OPCODE
>0066           1444 NMIVEC  EQU     0066H        ;THE NON-MASKABLE INTERRUPT IS
                1445                              ;USED FOR DATA SYNC BETWEEN
                1446                              ;THE Z-80 AND 1771
                1447 ;
                1448 ;
                1449 ;
F6B1  79        1450 SELECT: LD      A,C          ;GET UNIT# PASSED IN C AND
F6B2  FE04      1451         CP      4            ; CHECK FOR MAXIMUM VALID#
F6B4  D0        1452         RET     NC           ;ERROR IF NUMBER > 3
F6B5  CD88F7    1453         CALL    TURNON       ;MAKE SURE DISKS ARE TURNED ON
F6B8  DB1C      1454         IN      A,(BITDAT)
F6BA  47        1455         LD      B,A          ;SAVE CURRENT DRIVE SELECT DATA
F6BB  E6F8      1456         AND     11111000B    ;MERGE IN NEW DRIVE UNIT# IN C
F6BD  B1        1457         OR      C            ;IN PLACE OF THE CURRENT ONE
F6BE  D31C      1458         OUT     (BITDAT),A   ;TO SELECT THE NEW DISK DRIVE
F6C0  CDAEF7    1459         CALL    FORCE        ;TEST NEW DRIVE'S READY STATUS
```

```
F6C3  2806      1460           JR      Z,SEL2-$    ;AND CONTINUE IF ITS READY
F6C5  78        1461           LD      A,B
F6C6  D31C      1462           OUT     (BITDAT),A ;ELSE PUT BACK OLD DRIVE SELECT
F6C8  3E80      1463           LD      A,10000000B;AND RETURN DRIVE-NOT-READY
F6CA  C9        1464           RET
              1465
F6CB  2165FF    1466 SEL2:     LD      HL,UNIT     ;POINT HL TO DRIVE SELECT DATA
F6CE  7E        1467           LD      A,(HL)      ;LOAD A WITH CURRENT UNIT#
F6CF  71        1468           LD      (HL),C      ;AND STORE NEW UNIT# FROM C
F6D0  FEFF      1469           CP      255         ;TEST IF NO DRIVE SELECTED
F6D2  2806      1470           JR      Z,SEL3-$    ;YET & SKIP NEXT SEGMENT IF SO
F6D4  23        1471           INC     HL          ;POINT TO HEAD POSITION TABLE
F6D5  85        1472           ADD     A,L         ;AND ADD IN NEW UNIT# AS INDEX
F6D6  6F        1473           LD      L,A
F6D7  DB11      1474           IN      A,(TRKREG)  ;GET CURRENT HEAD POSITION
F6D9  77        1475           LD      (HL),A      ;AND STORE IN TABLE @ HL
F6DA  2166FF    1476 SEL3:     LD      HL,TRKTAB
F6DD  7D        1477           LD      A,L
F6DE  81        1478           ADD     A,C         ;INDEX INTO TABLE TO GET
F6DF  6F        1479           LD      L,A         ;HEAD POSITION OF NEW DRIVE
F6E0  7E        1480           LD      A,(HL)
F6E1  FEFF      1481           CP      255         ;TEST IF NEW DRIVE WAS EVER
F6E3  2804      1482           JR      Z,HOME-$    ;SELECTED AND DO A HOME IF NOT
F6E5  D311      1483           OUT     (TRKREG),A ;OUTPUT DRIVE'S CURRENT HEAD
F6E7  AF        1484           XOR     A           ;POSITION TO THE TRACK REGISTER
F6E8  C9        1485           RET
              1486 ;
              1487 ;
              1488 ;
F6E9  CDABF7    1489 HOME:     CALL    READY       ;CLEAR DISK CONTROLLER
F6EC  C0        1490           RET     NZ          ;EXIT IF DRIVE NOT READY
F6ED  AF        1491           XOR     A
F6EE  326DFF    1492           LD      (TRACK),A   ;SET TRACK# IN MEM TO ZERO
F6F1  060C      1493 RESTOR:   LD      B,RSTCMD    ;LOAD B WITH A RESTORE COMMAND
F6F3  CD93F7    1494           CALL    STEP        ;EXECUTE HEAD MOVING OPERATION
F6F6  EE04      1495           XOR     00000100B   ;GET TRUE TRACK 0 STATUS
F6F8  E69C      1496           AND     10011100B   ;MASK TO ERROR BITS
F6FA  C9        1497           RET                 ;RETURN 1771 STATUS IN A
              1498 ;
              1499 ;
              1500 ;
F6FB  CDABF7    1501 SEEK:     CALL    READY       ;CLEAR DISK CONTROLLER
F6FE  C0        1502           RET     NZ          ;EXIT IF DRIVE NOT READY
F6FF  79        1503           LD      A,C         ;GET TRACK# DATA FROM C AND
F700  FE4D      1504           CP      77          ;CHECK FOR MAXIMUM VALID#
F702  D0        1505           RET     NC          ;FORGET IT IF TRACK# > 76
F703  326DFF    1506           LD      (TRACK),A   ;ELSE STORE TRACK# FOR SEEK
F706  D313      1507           OUT     (DATREG),A ;OUTPUT TRACK # TO 1771
F708  061C      1508           LD      B,SKCMD     ;LOAD B WITH A SEEK COMMAND AND
F70A  CD93F7    1509           CALL    STEP        ;GO SEEK WITH PROPER STEP RATE
F70D  E698      1510           AND     10011000B   ;MASK TO READY,SEEK & CRC ERROR
F70F  C8        1511           RET     Z           ;BITS AND RETURN IF ALL GOOD
              1512
F710  CDF1F6    1513           CALL    RESTOR      ;ELSE TRY TO RE-CAILBRATE HEAD
F713  C0        1514           RET     NZ          ;ERROR IF WE CAN'T FIND TRACK 0
F714  79        1515           LD      A,C
F715  D313      1516           OUT     (DATREG),A ;OUTPUT TRACK# TO 1771
F717  061C      1517           LD      B,SKCMD
F719  CD93F7    1518           CALL    STEP        ;TRY TO SEEK THE TRACK AGAIN
F71C  E698      1519           AND     10011000B
F71E  C9        1520           RET                 ;RETURN FINAL SEEK STATUS IN A
              1521 ;
              1522 ;
              1523 ;
```

```
F71F  CDABF7   1524 WRITE:  CALL   READY       ;CLEAR THE DISK CONTROLLER
F722  C0       1525         RET    NZ          ;EXIT IF DRIVE NOT READY
F723  CB77     1526         BIT    6,A
F725  C0       1527         RET    NZ          ;EXIT IF DISK WRITE-PROTECTED
F726  06A8     1528         LD     B,WRTCMD
F728  1806     1529         JR     RDWRT-$
              1530
F72A  CDABF7   1531 READ:   CALL   READY       ;CLEAR DISK CONTROLLER
F72D  C0       1532         RET    NZ          ;EXIT IF DRIVE NOT READY
F72E  0688     1533         LD     B,RDCMD
F730  2271FF   1534 RDWRT:  LD     (IOPTR),HL  ;STORE DISK I/O DATA POINTER
F733  216EFF   1535         LD     HL,SECTOR
F736  71       1536         LD     (HL),C      ;STORE SECTOR# FOR READ/WRITE
F737  23       1537         INC    HL
F738  70       1538         LD     (HL),B      ;SAVE READ/WRITE COMMAND BYTE
F739  23       1539         INC    HL
F73A  3602     1540         LD     (HL),2      ;SET DISK RE-TRY COUNT
F73C  F3       1541 RW1:    DI                 ;NO INTERRUPTS DURING DISK I/O
F73D  216600   1542         LD     HL,NMIVEC   ;SAVE BYTE AT NMI VECTOR LOCAT
F740  56       1543         LD     D,(HL)      ;IN D FOR DURATION OF READ/WRIT
F741  36C9     1544         LD     (HL),RET    ;LOOP AND REPLACE IT WITH A RET
F743  216BFF   1545         LD     HL,RECLEN
F746  46       1546         LD     B,(HL)      ;B=NUMBER OF BYTES/SECTOR
F747  0E13     1547         LD     C,DATREG    ;C=1771 DATA REGISTER PORT#
F749  2A71FF   1548         LD     HL,(IOPTR)  ;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         ;TO TEST HEAD LOAD STATUS
F756  3A6FFF   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         OR     HLOAD       ;ELSE MERGE IN HLD BIT
F75D  CDA3F7   1556 RW2:    CALL   CMDOUT      ;START 1771 DOING IT'S THING
F760  CB6F     1557         BIT    5,A         ;TEST IF COMMAND IS A R OR W
F762  200D     1558         JR     NZ,WLOOP-$  ;AND JUMP TO THE CORRECT LOOP
F764  76       1559 RLOOP:  HALT
F765  EDA2     1560         INI
F767  C264F7   1561         JP     NZ,RLOOP
F76A  CD9CF7   1562         CALL   BUSY        ;LOOP UNTIL 1771 COMES UN-BUSY
F76D  E69C     1563         AND    10011100B   ;MASK OFF TO READY,NOT FOUND.CRC
F76F  180B     1564         JR     RW3-$       ;AND LOST DATA STATUS BITS
              1565
F771  76       1566 WLOOP:  HALT
F772  EDA3     1567         OUTI
F774  C271F7   1568         JP     NZ,WLOOP
F777  CD9CF7   1569         CALL   BUSY
F77A  E6BC     1570         AND    10111100B   ;MASK OFF AS ABOVE + WRT FAULT
F77C  216600   1571 RW3:    LD     HL,NMIVEC
F77F  72       1572         LD     (HL),D      ;RESTORE BYTE @ NMI VECTOR
F780  FB       1573         EI
F781  C8       1574         RET    Z           ;RETURN IF NO DISK I/O ERRORS
F782  2170FF   1575         LD     HL,RETRY
F785  35       1576         DEC    (HL)        ;DECREMENT RE-TRY COUNT AND
F786  2002     1577         JR     NZ,RW4-$    ; EXECUTE COMAND AGAIN IF NOT=0
F788  B7       1578         OR     A
F789  C9       1579         RET                ;ELSE RETURN 1771 ERROR STATUS
              1580
F78A  216DFF   1581 RW4:    LD     HL,TRACK
F78D  4E       1582         LD     C,(HL)      ;GET TRACK# FOR THIS OPERATION
F78E  CDFBF6   1583         CALL   SEEK        ;TRY TO RE-CAILBRATE THE HEAD
F791  18A9     1584         JR     RW1-$       ;BEFORE READ OR WRITE AGAIN
              1585 ;
              1586 ;
              1587 ;
```

```
F793  3A6AFF  1588 STEP:   LD      A,(SPEED)   ;GET STEP SPEED VARIABLE
F796  E603    1589        AND     00000011B
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
              1596 ;
              1597 ;
              1598 ;
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
              1604 ;
              1605 ;
              1606 ;
F7AB  CDB8F7  1607 READY:  CALL    TURNON      ;KEEP THOSE DISKS SPINING FOLKS
F7AE  3ED0    1608 FORCE:  LD      A,FINCMD    ;ISSUE FORCE INTERRUPT COMMAND
F7B0  CDA3F7  1609        CALL    CMDOUT
F7B3  DB10    1610        IN      A,(STSREG)  ;READ STATUS REGISTER CONTENTS
F7B5  CB7F    1611        BIT     7,A         ;TEST DRIVE NOT READY BIT
F7B7  C9      1612        RET
              1613 ;
              1614 ;
              1615 ;
F7B8  3E1E    1616 TURNON: LD      A,30
F7BA  326CFF  1617        LD      (MOTOR),A   ;RE-LOAD MOTOR TURN-OFF TIMER
F7BD  CDA8F7  1618        CALL    PAUSE
F7C0  DB1C    1619        IN      A,(BITDAT)
F7C2  CB57    1620        BIT     2,A         ;TEST IF MOTORS HAVE STOPPED
F7C4  C8      1621        RET     Z           ;AND EXIT IF STILL TURNED ON
F7C5  E6BB    1622        AND     10111011B   ;ELSE RE-ENABLE DRIVE SELECTS
F7C7  D31C    1623        OUT     (BITDAT),A  ;AND ACTIVATE THE MOTOR RELAY
F7C9  C5      1624        PUSH    BC
F7CA  0600    1625        LD      B,0         ;SET READY LOOP MAX TIMEOUT
F7CC  CDDCF7  1626 TURN2:  CALL    WAIT        ;WAIT 1/93 SECOND & TEST READY
F7CF  2802    1627        JR      Z,TURN3-$   ;EXIT LOOP IF DRIVE READY
F7D1  10F9    1628        DJNZ    TURN2-$     ;ELSE TRY AGAIN UP TO 256 TIMES
F7D3  0609    1629 TURN3:  LD      B,9
F7D5  CDDCF7  1630 TURN4:  CALL    WAIT        ;GIVE ABT 1/10 SEC MORE DELAY
F7D8  10FB    1631        DJNZ    TURN4-$
F7DA  C1      1632        POP     BC
F7DB  C9      1633        RET
              1634 ;
              1635 ;
F7DC  DB1B    1636 WAIT:   IN      A,(CTC3)    ;GET CURRENT CTC3 COUNT VALUE
F7DE  4F      1637        LD      C,A
F7DF  DB1B    1638 WAIT2:  IN      A,(CTC3)
F7E1  B9      1639        CP      C           ;SEE IF CTC3 CHANGED BY 1 COUNT
F7E2  28FB    1640        JR      Z,WAIT2-$   ;AND LOOP UNTIL IT CHANGES
F7E4  18C8    1641        JR      FORCE-$     ;THEN TEST DRIVE READY STATUS
              1642 ;
              1643 ;
              1644 ;
              1645 ;
              1646 ;
F7E6  0000    1647 ROMEND: DEFW    0           ;TAIL OF FREE MEM LINKED LIST
              1648 ;
>FF00         1649        ORG     RAM
              1650        INCLUDE MEMORY.ASM
```

```
F37B  C0        0745          RET     NZ          ;ERROR IF > 4 NUMBERS ENTERED
F37C  C5        0746 PARA2:   PUSH    BC          ;SAVE PARAMETER COUNT
F37D  CD9FF3    0747          CALL    GETHEX      ;READ A NUMBER FROM LINE BUFFER
F380  C1        0748          POP     BC
F381  D8        0749 PARA4:   RET     C           ;ERROR IF RESULT OVER 16 BITS
F382  DD217CFF  0750          LD      IX,PARAM1   ;POINT TO PARAM STORAGE AREA
F386  DD09      0751          ADD     IX,BC       ;ADD PARAMETER COUNT IN BC
F388  DD7500    0752          LD      (IX+0),L
F38B  DD7401    0753          LD      (IX+1),H    ;STORE DATA RET FROM 'GETHEX'
F38E  FE20      0754          CP      ' '
F390  28E4      0755          JR      Z,PARA1-$   ;GET ANOTHER ITEM IF SPACE
F392  FE2C      0756          CP      ','
F394  28E0      0757          JR      Z,PARA1-$   ;GET ANOTHER ITEM IF COMMA
F396  FE0D      0758          CP      CR
F398  37        0759          SCF                 ;ELSE CHECK FOR CARRIAGE RETURN
F399  C0        0760          RET     NZ          ; AND EXIT WITH CY=1 IF NOT
F39A  79        0761 PAREND:  LD      A,C
```

```
F39B  CB3F      0762          SRL     A           ;A=COUNT OF NUMBERS ENTERED
F39D  3C        0763          INC     A
F39E  C9        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 ;
F39F  210000    0772 GETHEX: LD      HL,0
F3A2  180B      0773          JR      GNUM3-$
                0774
F3A4  0604      0775 GNUM1:  LD      B,4
F3A6  29        0776 GNUM2:  ADD     HL,HL       ;MULTIPLY RESULT BY 16
F3A7  D8        0777          RET     C           ;RETURN IF IT OVERFLOWS 16 BITS
F3A8  10FC      0778          DJNZ    GNUM2-$
F3AA  5F        0779          LD      E,A         ;APPEND NEW LOW ORDER DIGIT
F3AB  1600      0780          LD      D,0         ;AND GET RESULT BACK INTO DE
F3AD  19        0781          ADD     HL,DE
F3AE  D8        0782          RET     C           ;RETURN IF OVERFLOW
F3AF  FD7E00    0783 GNUM3:  LD      A,(IY+0)    ;GET A CHAR FROM LINE INPUT
F3B2  FD23      0784          INC     IY          ; BUFFER @ IY AND BUMP IY
F3B4  4F        0785          LD      C,A
F3B5  CDBDF3    0786          CALL    ASCHEX      ;CONVERT ASCII TO NUMERIC
F3B8  30EA      0787          JR      NC,GNUM1-$
F3BA  79        0788          LD      A,C
F3BB  B7        0789          OR      A
F3BC  C9        0790          RET
                0791 ;
                0792 ;
F3BD  D630      0793 ASCHEX: SUB     '0'
F3BF  D8        0794          RET     C
F3C0  FE0A      0795          CP      10
F3C2  3F        0796          CCF
F3C3  D0        0797          RET     NC
F3C4  D607      0798          SUB     7
F3C6  FE0A      0799          CP      10
F3C8  D8        0800          RET     C
F3C9  FE10      0801          CP      16
F3CB  3F        0802          CCF
F3CC  C9        0803          RET
                0804 ;
                0805 ;
                0806 ;
F3CD  7C        0807 PUT4HS: LD      A,H
F3CE  CDD8F3    0808          CALL    PUT2HX
F3D1  7D        0809          LD      A,L
F3D2  CDD8F3    0810 PUT2HS: CALL    PUT2HX
F3D5  C302F4    0811          JP      SPACE
                0812 ;
                0813 ;
F3D8  F5        0814 PUT2HX: PUSH    AF
F3D9  1F        0815          RRA
F3DA  1F        0816          RRA
F3DB  1F        0817          RRA
F3DC  1F        0818          RRA
F3DD  CDE1F3    0819          CALL    PUTNIB
F3E0  F1        0820          POP     AF
F3E1  E60F      0821 PUTNIB: AND     00001111B
```

```
F3E3  C690      0822            ADD     A,90H
F3E5  27        0823            DAA
F3E6  CE40      0824            ADC     A,40H
F3E8  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  18F6      0848          JR        PMSG-$
                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 ;
                0861 ;ECHO INPUTS ONE CHARACTER FROM THE CONSOLE
                0862 ;DEVICE, PRINTS IT ON THE CONSOLE OUTPUT AND
                0863 ;THEN RETURNS IT IN REGISTER A WITH BIT 7 RESET
                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
F418  CD06F0    0882          CALL      CONST       ;SEE IF CONSOLE INPUT PENDING
F41B  280F      0883          JR        Z,OUTP2-$
F41D  CD09F0    0884          CALL      CONIN
F420  FE0D      0885          CP        CR          ;SEE IF <CR> WAS TYPED
F422  2805      0886          JR        Z,OUTP1-$
F424  CD09F0    0887          CALL      CONIN       ;WAIT FOR ANOTHER INPUT CHAR
F427  1803      0888          JR        OUTP2-$     ; THEN RET TO CALLING ROUTINE
                0889
F429  3284FF    0890 OUTP1:   LD        (ESCFLG),A  ;SET ESC FLAG TO NON-ZERO VALUE
F42C  3A84FF    0891 OUTP2:   LD        A,(ESCFLG)
F42F  B7        0892          OR        A           ;RETURN CURRENT STATUS OF ESC
F430  C9        0893          RET                   ; FLAG TO CALLING ROUTINE
                0894 ;
                0895 ;
                0896 ;
                0897          INCLUDE INTSRV.ASM
```

```
                0898 ;*************************************************************
                0899 ;*                                                           *
                0900 ;*       INTERRUPT SERVICE ROUTINES FOR KEYBOARD             *
                0901 ;*       INPUT AND REAL-TIME CLOCK FUNCTIONS                 *
                0902 ;*                                3-Aug-80                   *
                0903 ;*                                                           *
                0904 ;*************************************************************
                0905 ;
                0906 ;
                0907 ;
                0908 ;
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
                0914 ;
                0915 ;
                0916 ;
F439  CD31F4    0917 KBDIN:  CALL    KBDST
F43C  28FB      0918         JR      Z,KBDIN-$  ;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
                0923 ;
                0924 ;
                0925 ;
                0926 ;
                0927 ;
F444  2133FF    0928 STASH:  LD      HL,LOCK    ;POINT TO SHIFT LOCK VARIABLES
F447  BE        0929         CP      (HL)       ;TEST IF A=SHIFT LOCK CHARACTER
F448  23        0930         INC     HL         ;THEN POINT TO LOCK FLAG
F449  2002      0931         JR      NZ,STASH2-$;JUMP IF NOT SHIFT CHARACTER
F44B  34        0932         INC     (HL)       ;ELSE COMPLIMENT THE SHIFT LOCK
F44C  C9        0933         RET                ;AND EXIT NOW
                0934
F44D  CB46      0935 STASH2: BIT     0,(HL)     ;TEST THE SHIFT LOCK FLAG
F44F  280A      0936         JR      Z,STASH3-$ ;JUMP IF SHIFT LOCK NOT SET
F451  FE40      0937         CP      40H        ;ELSE CHECK FOR SHIFTABLE CHAR
F453  3806      0938         JR      C,STASH3-$ ;AND JUMP IF NOT = OR GREATER
F455  FE7F      0939         CP      7FH        ;THAN '@' AND LESS THAN RUBOUT
F457  3002      0940         JR      NC,STASH3-$
F459  EE20      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  FE10      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
                0953 ;
                0954 ;
                0955 ;
                0956 ;
```

```
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  85        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 ;
                0970 ;
                0971 ;SOFTWARE DISK MOTOR TURN-OFF TIMER ROUTINE
                0972 ;
F480  216CFF    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
                0980 ;
                0981 ;
                0982 ;-- INTERRUPT SERVICE ROUTINE FOR PARALLEL KEYBOARD --
                0983 ;
F48C  ED7335FF  0984 KEYSRV: LD      (SPSAVE),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
                0994 ;
                0995 ;
                0996 ;
                0997 ;-- INTERRUPT SERVICE ROUTINE FOR ONE SECOND TIMER --
                0998 ;
F49F  ED7335FF  0999 TIMER:  LD      (SPSAVE),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
F4AA  2A57FF    1005         LD      HL,(TIKVEC);GET CLOCK INTERRUPT RTN VECTOR
F4AD  1812      1006         JR      DSPTCH-$   ; AND JUMP TO DISPATCH POINT
                1007 ;
                1008 ;
                1009 ;
                1010 ;-- SERIAL INPUT INTERRUPT SERVICE ROUTINE FOR SIO --
                1011 ;
F4AF  ED7335FF  1012 SIOINT: LD      (SPSAVE),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
```

```
F4C1  CDE7F4      1021 DSPTCH: CALL    CALLHL      ;CALL SUBROUTINE ADDRESSED BY H
F4C4  F1          1022         POP     AF
F4C5  C1          1023         POP     BC
F4C6  D1          1024         POP     DE
F4C7  E1          1025         POP     HL
F4C8  ED7B35FF    1026         LD      SP,(SPSAVE)
F4CC  FB          1027         EI                  ;RE-ENABLE INTERRUPTS & RETURN
F4CD  ED4D        1028         RETI
                  1029 ;
                  1030 ;
                  1031 ;-- RX ERROR INTERRUPT SERVICE ROUTINE FOR SIO --
                  1032 ;
                  1033 ;ARRIVE HERE IF RECEIVE INTERRUPT FROM FRAMING, OVERRUN
                  1034 ;AND PARITY ERRORS. (PARITY CAN BE DISABLED)
                  1035 ;
F4CF  ED7335FF    1036 SIOERR: LD      (SPSAVE),SP  ;SAVE USER STACK POINTER AND
F4D3  3157FF      1037         LD      SP,TMPSTK+32 ; SWITCH TO LOCAL STACK
F4D6  F5          1038         PUSH    AF
F4D7  CDF5F4      1039         CALL    SIOIN2      ;CLEAR BAD CHARACTER FROM SIO
F4DA  3E07        1040         LD      A,'G'-64
F4DC  CD15F5      1041         CALL    SIOXMT      ;OUTPUT A CTL-G AS A WARNING
F4DF  F1          1042         POP     AF
F4E0  ED7B35FF    1043         LD      SP,(SPSAVE)
F4E4  FB          1044         EI
F4E5  ED4D        1045         RETI
                  1046 ;
                  1047 ;
F4E7  E9          1048 CALLHL: JP      (HL)
                  1049 ;
                  1050 ;
                  1051 ;
                  1052 ;POLLED MODE I/O ROUTINES FOR SIO CHANEL B
                  1053 ;
F4E8  DB07        1054 SIOST:  IN      A,(SIOCPB) ;GET SIO STATUS REGISTER
F4EA  E601        1055         AND     00000001B
F4EC  C8          1056         RET     Z          ;ACC=0 IF NO DATA AVAILABLE
F4ED  3EFF        1057         LD      A,255
F4EF  C9          1058         RET
                  1059 ;
                  1060 ;
F4F0  CDE8F4      1061 SIOIN:  CALL    SIOST      ;TEST CONSOLE STATUS
F4F3  28FB        1062         JR      Z,SIOIN-$  ;LOOP UNTIL DATA IS RECEIVED
R4F5  3E30        1063 SIOIN2: LD      A,00110000B     ;RESET STATUS BITS IN SIO FO
F4F7  D307        1064         OUT     (SIOCPB),A ;PARITY/OVERRUN/FRAMING ERRORS,
F4F9  DB05        1065         IN      A,(SIODPB) ;THEN GET THE INPUT CHARACTER
F4FB  E67F        1066         AND     01111111B
F4FD  C9          1067         RET
                  1068 ;
                  1069 ;
F4FE  FE20        1070 SIOOUT: CP      ' '        ;TEST FOR CONTROL CHARACTERS
F500  3013        1071         JR      NC,SIOXMT-$;JUMP IF PRINTABLE CHARACTER
F502  CD15F5      1072         CALL    SIOXMT     ;ELSE SEND CONTROL CHARACTER
F505  3A79FF      1073         LD      A,(NULLS)  ;AND THEN SEND NULLS AS PADDING
F508  3C          1074         INC     A          ;GET NULL PAD COUNT AND FIX SO
F509  1806        1075         JR      PAD1-$     ;THAT COUNT=0 SENDS NO NULLS
                  1076
F50B  F5          1077 PAD:    PUSH    AF
F50C  AF          1078         XOR     A
F50D  CD15F5      1079         CALL    SIOXMT     ;OUTPUT A NULL TO THE SIO
F510  F1          1080         POP     AF
F511  3D          1081 PAD1:   DEC     A
F512  20F7        1082         JR      NZ,PAD-$   ;LOOP SENDING NULLS TO SIO
F514  C9          1083         RET
                  1084 ;
                  1085 ;
F515  F5          1086 SIOXMT: PUSH    AF
F516  DB07        1087 SIOX1:  IN      A,(SIOCPB)
```

```
F518  E604      1088          AND     00000100B  ;TEST TBE STATUS BIT
F51A  28FA      1089          JR      Z,SIOX1-$
F51C  F1        1090          POP     AF
F51D  D305      1091          OUT     (SIODPB),A ;OUTPUT DATA TO SIO
F51F  C9        1092          RET
                1093 ;
                1094 ;
                1095 ;
                1096 ;
                1097          INCLUDE CRTOUT.ASM
                1098 ;*****************************************************
                1099 ;*                                                   *
                1100 ;*       MEMORY-MAPPED CRT OUTPUT DRIVER             *
                1101 ;*                                                   *
                1102 ;*       Russell Smith     18-August-1980            *
                1103 ;*                                                   *
                1104 ;*****************************************************
                1105 ;
                1106 ;
>0030           1107 CRTBAS   EQU     CRTMEM.SHR.8 ;START PAGE# OF 3K CRT SPACE
>003C           1108 CRTTOP   EQU     CRTMEM+3072.SHR.8 ;END PAGE# OF CRT SPACE
                1109 ;
                1110 ;
F520  E5        1111 CRTOUT:  PUSH    HL
F521  D5        1112          PUSH    DE
F522  C5        1113          PUSH    BC
F523  CBBF      1114          RES     7,A
F525  4F        1115          LD      C,A
F526  F3        1116          DI                 ;KEEP WOLVES AWAY FOR A WHILE
F527  ED7335FF  1117          LD      (SPSAVE),SP
F52B  3157FF    1118          LD      SP,TMPSTK+32 ;POINT SP TO TOP LOCAL STACK
F52E  DB1C      1119          IN      A,(BITDAT)
F530  CBFF      1120          SET     7,A        ;SELECT ROM/CRT MEMORY BANK
F532  D31C      1121          OUT     (BITDAT),A
                1122 ;
                1123 ;FIRST REMOVE THE OLD CURSOR CHARACTER FROM THE SCREEN
                1124 ;
F534  2175FF    1125          LD      HL,CHRSAV  ;GET CHAR OVERLAYED BY CURSOR
F537  46        1126          LD      B,(HL)
F538  2A73FF    1127          LD      HL,(CURSOR);LOAD HL WITH CURSOR POINTER
F53B  7C        1128          LD      A,H
F53C  E60F      1129          AND     00001111B  ;INSURANCE THAT HL CAN'T
F53E  F630      1130          OR      CRTBAS     ;EVER POINT OUTSIDE CRT MEMORY
F540  67        1131          LD      H,A
F541  70        1132          LD      (HL),B     ;RMV CURSOR BY RESTORING CHAR
                1133 ;
                1134 ;PROCESS CHARACTER PASSED IN C
                1135 ;
F542  CD65F5    1136          CALL    OUTCH
                1137 ;
                1138 ;NOW STORE A NEW CURSOR CHARACTER AT THE CURSOR LOCATION
                1139 ;
F545  7E        1140          LD      A,(HL)     ;GET CHAR AT NEW CURSOR LOCAT.
F546  3275FF    1141          LD      (CHRSAV),A ;SAVE FOR NEXT TIME 'CRTOUT' IS
;                                                  CALLED
F549  FE20      1142          CP      ' '        ;TEST IF CHARACTER IS A SPACE
F54B  CBFF      1143          SET     7,A        ;THEN TURN ON BIT 7 TO ENABLE
;                                                  BLINK
F54D  2003      1144          JR      NZ,CRT2-$  ;JUMP IF CHARACTER IS NON-BLANK
F54F  3A76FF    1145          LD      A,(CSRCHR) ;ELSE GET CHAR USED FOR CURSOR
F552  77        1146 CRT2:    LD      (HL),A     ;STORE CHAR IN A AS CURSOR MARK
F553  2273FF    1147          LD      (CURSOR),HL;SAVE HL AS CURSOR POINTER
                1148
F556  ED7B35FF  1149          LD      SP,(SPSAVE)
F55A  DB1C      1150          IN      A,(BITDAT)
```

```
F55C  CBBF      1151            RES     7,A         ;SWITCH BACK LOWER 16K OF RAM
F55E  D31C      1152            OUT     (BITDAT),A
F560  FB        1153            EI                  ;INTERRUPTS ARE SAFE AGAIN
F561  C1        1154            POP     BC
F562  D1        1155            POP     DE
F563  E1        1156            POP     HL
F564  C9        1157            RET
                1158 ;
                1159 ;
                1160 ;
F565  1178FF    1161 OUTCH:     LD      DE,LEADIN
F568  1A        1162            LD      A,(DE)      ;GET LEAD-IN SEQUENCE STATE
F569  B7        1163            OR      A
F56A  C270F6    1164            JP      NZ,MULTI    ;JUMP IF IN A LEAD-IN SEQUENCE
F56D  79        1165            LD      A,C         ; ELSE PROCESS CHARACTER IN C
F56E  FE20      1166            CP      ' '
F570  380F      1167            JR      C,CONTRL-$  ;JUMP IF A CONTROL CHARACTER
F572  71        1168 DISPLA:    LD      (HL),C      ;ELSE STORE DISPLAYABLE CHAR
F573  23        1169            INC     HL          ;AND ADV POINTER TO NEXT COLUMN
F574  7D        1170            LD      A,L
F575  E67F      1171            AND     01111111B   ;EXTRACT COLUMN# FROM HL
F577  FE50      1172            CP      80
F579  D8        1173            RET     C           ;EXIT IF NOT PAST COLUMN 79
F57A  CDE7F5    1174            CALL    RETURN      ;ELSE DO AUTOMATIC <CR>
F57D  CD42F6    1175            CALL    LFEED       ;AND LINEFEED
F580  C9        1176            RET
                1177 ;
                1178 ;
                1179 ;
F581  E5        1180 CONTRL:    PUSH    HL
F582  218FF5    1181            LD      HL,CTLTAB   ;SEARCH FOR CONTROL CHARACTER
F585  010D00    1182            LD      BC,CTLSIZ/3;HANDLING SUBROUTINE IN TABLE
F588  CD60F3    1183            CALL    SEARCH
F58B  E1        1184            POP     HL
F58C  C0        1185            RET     NZ          ;EXIT IF NOT IMPLEMENTED
F58D  C5        1186            PUSH    BC
F58E  C9        1187            RET                 ;DO SNEAKY JUMP TO PRESERVE
;                                                    REGISTERS
                1188
F58F  1F        1189 CTLTAB:    DEFB    '_'-64
F590  1E        1190            DEFB    '^'-64
F591  1B        1191            DEFB    '['-64
F592  1A        1192            DEFB    'Z'-64
F593  18        1193            DEFB    'X'-64
F594  11        1194            DEFB    'Q'-64
F595  0D        1195            DEFB    'M'-64
F596  0C        1196            DEFB    'L'-64
F597  0B        1197            DEFB    'K'-64
F598  0A        1198            DEFB    'J'-64
F599  09        1199            DEFB    'I'-64
F59A  08        1200            DEFB    'H'-64
F59B  07        1201            DEFB    'G'-64
                1202
F59C  DCF5      1203            DEFW    BELL        ;CTL-G IS THE BELL
F59E  BEF5      1204            DEFW    BAKSPC      ;CTL-H IS CURSOR LEFT
F5A0  CCF5      1205            DEFW    TAB         ;CTL-I IS TAB
F5A2  42F6      1206            DEFW    LFEED       ;CTL-J IS CURSOR DOWN
F5A4  2CF6      1207            DEFW    UPCSR       ;CTL-K IS CURSOR UP
F5A6  C4F5      1208            DEFW    FORSPC      ;CTL-L IS CURSOR RIGHT
```

```
F5A8  E7F5      1209            DEFW    RETURN      ;CTL-M IS <CR>
F5AA  11F6      1210            DEFW    CLREOS      ;CTL-Q CLEAR TO END-OF-SCREEN
F5AC  03F6      1211            DEFW    CLREOL      ;CTL-X IS CLEAR TO END-OF-LINE
F5AE  ECF5      1212            DEFW    CLRSCN      ;CTL-Z IS CLEAR SCREEN
F5B0  B6F5      1213            DEFW    ESCAPE      ;CTL-[ IS ESCAPE
F5B2  6CF6      1214            DEFW    HOMEUP      ;CTL-^ IS HOME UP
F5B4  BAF5      1215            DEFW    STUFF       ;CTL-_ IS DISPLAY CONTROL CHARS
                1216
>0027           1217 CTLSIZ     EQU     $-CTLTAB
                1218 ;
                1219 ;
F5B6  3E01      1220 ESCAPE:    LD      A,1
F5B8  12        1221            LD      (DE),A      ;SET LEAD-IN SEQUENCE STATE
F5B9  C9        1222            RET                 ;FOR XY CURSOR POSITIONING MODE
                1223 ;
                1224 ;
F5BA  3E04      1225 STUFF:     LD      A,4
F5BC  12        1226            LD      (DE),A      ;SET LEAD-IN SEQUENCE STATE
F5BD  C9        1227            RET                 ;FOR CONTROL CHAR OUTPUT MODE
                1228 ;
                1229 ;
F5BE  7D        1230 BAKSPC     LD      A,L         ;CHECK FOR LEFT MARGIN
F5BF  E67F      1231            AND     01111111B
F5C1  C8        1232            RET     Z           ;ABORT IF IN LEFTMOST COLUMN
F5C2  2B        1233            DEC     HL          ;BACK UP CURSOR POINTER
F5C3  C9        1234            RET
                1235 ;
                1236 ;
F5C4  7D        1237 FORSPC:    LD      A,L         ;CHECK FOR RIGHTMOST COLUNM
F5C5  E67F      1238            AND     01111111B
F5C7  FE4F      1239            CP      79
F5C9  D0        1240            RET     NC          ;DO NOTHING IF ALREADY THERE
F5CA  23        1241            INC     HL
F5CB  C9        1242            RET                 ;ELSE ADVANCE CURSOR POINTER
                1243 ;
                1244 ;
F5CC  110800    1245 TAB:       LD      DE,8        ;TABS ARE EVERY 8 COLUMNS
F5CF  7D        1246            LD      A,L         ;GET COLUMN COMPONENT OF
F5D0  E678      1247            AND     01111000B   ; PREVIOUS TAB POSITION
F5D2  83        1248            ADD     A,E
F5D3  FE50      1249            CP      80          ;EXIT IF NEXT TAB COLUMN WOULD
F5D5  D0        1250            RET     NC          ; BE PAST THE RIGHT MARGIN
F5D6  7D        1251            LD      A,L
F5D7  E6F8      1252            AND     11111000B   ;ELSE INCREMENT THE CURSOR
F5D9  6F        1253            LD      L,A         ; POINTER FOR REAL
F5DA  19        1254            ADD     HL,DE
F5DB  C9        1255            RET
                1256 ;
                1257 ;
F5DC  DB1C      1258 BELL:      IN      A,(BITDAT)
F5DE  CBEF      1259            SET     5,A         ;TOGGLE BIT 5 OF SYSTEM PIO TO
F5E0  D31C      1260            OUT     (BITDAT),A  ;TRIGGER BELL HARDWARE TO SOUND
F5E2  CBAF      1261            RES     5,A
F5E4  D31C      1262            OUT     (BITDAT),A
F5E6  C9        1263            RET
                1264 ;
                1265 ;
F5E7  7D        1266 RETURN:    LD      A,L
F5E8  E680      1267            AND     10000000B
F5EA  6F        1268            LD      L,A         ;MOVE CURSOR POINTER BACK
F5EB  C9        1269            RET                 ; TO START OF LINE
                1270 ;
                1271 ;
F5EC  210030    1272 CLRSCN:    LD      HL,CRTMEM
```

```
F5EF  E5        1273            PUSH    HL
F5F0  110130    1274            LD      DE,CRTMEM+1
F5F3  01000C    1275            LD      BC,24*128
F5F6  3620      1276            LD      (HL),' '
F5F8  EDB0      1277            LDIR                    ;FILL CRT MEMORY WITH SPACES
F5FA  E1        1278            POP     HL              ;POINT TO HOME CURSOR POSITION
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
                1283 ;
                1284 ;
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 CHARS
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
                1295 ;
                1296 ;
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
F61A  17        1302            RLA
F61B  7C        1303            LD      A,H
F61C  17        1304            RLA                     ;ROW# COMPONENT OF HL INTO A
F61D  E61F      1305            AND     00011111B
F61F  B9        1306            CP      C               ;SEE IF HL IS AT BOTTOM ROW
                                                         OF SCREEN
F620  2808      1307            JR      Z,CLRS2-$       ;AND LEAVE CLEAR LOOP IF SO
F622  CD37F6    1308            CALL    DNCSR           ;ELSE POINT HL TO NEXT ROW DOWN
F625  CD60F6    1309            CALL    CLRLIN          ;AND FILL THAT LINE WITH SPACES
F628  18EF      1310            JR      CLRS1-$
                1311
F62A  E1        1312 CLRS2:     POP     HL              ;RESTR ORIGINAL CURSOR POINTER
F62B  C9        1313            RET
                1314 ;
                1315 ;
F62C  1180FF    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      CRTBAS          ;CHECK FOR UNDERFLOW OF POINTER
F633  D0        1320            RET     NC
F634  263B      1321            LD      H,CRTTOP-1      ;WRAP CURSOR AROUND MODULO 3K
F636  C9        1322            RET
                1323 ;
                1324 ;
F637  118000    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      CRTTOP          ;CHECK FOR OVERFLOW OF POINTER
F63E  D8        1329            RET     C
F63F  2630      1330            LD      H,CRTBAS        ;RESET POINTER MODULO 128*24
F641  C9        1331            RET
                1332 ;
                1333 ;
                1334 ;
F642  7D        1335 LFEED:     LD      A,L
F643  17        1336            RLA
F644  7C        1337            LD      A,H
```

```
F645  17        1338           RLA                 ;EXTRACT ROW# COMPONENT OF HL
F646  E61F      1339           AND     00011111B
F648  4F        1340           LD      C,A         ;COPY ROW# TO C FOR SCROLL TEST
F649  CD37F6    1341           CALL    DNCSR       ;MOVE CURSOR TO NEXT ROW DOWN
F64C  3A77FF    1342           LD      A,(BASE)    ;TEST IF CURSOR ON BOTTOM ROW
F64F  B9        1343           CP      C           ;OF SCREEN BEFORE MOVING DOWN
F650  C0        1344           RET     NZ          ;EXIT IF NOT AT BOTTOM
                1345
F651  E5        1346           PUSH    HL          ;ELSE PREP TO SCROLL SCREEN UP
F652  CD60F6    1347           CALL    CLRLIN      ;FILL NEW BOTTOM LINE WTH SPACES
F655  29        1348           ADD     HL,HL
F656  7C        1349           LD      A,H         ;GET ROW# PART OF HL INTO A
F657  E61F      1350           AND     00011111B
F659  3277FF    1351           LD      (BASE),A    ;STORE NEW BASE LINE#
F65C  D314      1352           OUT     (SCROLL),A  ;SCROLL UP NEW BLANK BOTTM LINE
F65E  E1        1353           POP     HL
F65F  C9        1354           RET
                1355 ;
                1356 ;
F660  7D        1357 CLRLIN:   LD      A,L
F661  E680      1358           AND     10000000B   ;POINT HL TO 1ST COLUMN OF ROW
F663  6F        1359           LD      L,A
F664  0650      1360           LD      B,80
F666  3620      1361 CLR:      LD      (HL),' '    ;STORE ASCII SPACES AT ADDR
;                                                   IN HL
F668  23        1362           INC     HL          ;AND INCREMENT HL
F669  10FB      1363           DJNZ    CLR-$       ;REPEAT NUMBER OF TIMES IN B
F66B  C9        1364           RET
                1365 ;
                1366 ;
F66C  0E20      1367 HOMEUP:   LD      C,' '       ;FAKE-OUT CURSOR ADDR ROUTINE
F66E  1817      1368           JR      SETROW-$    ;TO DO HOMEUP ALMOST FOR FREE
                1369 ;
                1370 ;
F670  EB        1371 MULTI:    EX      DE,HL       ;UNCONDITIONALLY RESET LEAD-IN
F671  3600      1372           LD      (HL),0      ;STATE TO ZERO BEFORE GOING ON
F673  EB        1373           EX      DE,HL
F674  FE01      1374           CP      1
F676  2008      1375           JR      NZ,M2TST-$
F678  79        1376 SETXY:    LD      A,C         ;GET SECOND CHAR OF SEQUENCE
F679  FE3D      1377           CP      '='
F67B  C0        1378           RET     NZ          ;ABORT SEQUENCE IF NOT '='
F67C  3E02      1379           LD      A,2
F67E  12        1380           LD      (DE),A      ;MAKE LEADIN=2 NEXT TIME
F67F  C9        1381           RET
                1382
F680  FE02      1383 M2TST:    CP      2
F682  2019      1384           JR      NZ,M3TST-$
F684  3E03      1385           LD      A,3
F686  12        1386           LD      (DE),A      ;MAKE LEADIN=3 NEXT TIME
F687  3A77FF    1387 SETROW:   LD      A,(BASE)    ;ARRIVE HERE ON THIRD CHAR
F68A  81        1388           ADD     A,C         ; OF ESC,'=',ROW,COL SEQUENCE
F68B  D61F      1389           SUB     ' '-1
F68D  D618      1390 SETR2:    SUB     24
F68F  30FC      1391           JR      NC,SETR2-$  ;VERIFY ROW# BETWEEN 0 AND 23
F691  C618      1392           ADD     A,24
F693  F660      1393           OR      CRTMEM.SHR.7 ;MERGE IN MSB'S OF CRT MEMORY
F695  67        1394           LD      H,A
F696  2E00      1395           LD      L,0
F698  CB3C      1396           SRL     H
F69A  CB1D      1397           RR      L
F69C  C9        1398           RET
                1399
F69D  FE03      1400 M3TST:    CP      3
F69F  200C      1401           JR      NZ,M4TST-$
```

```
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     80
F6A6  30FC      1405         JR      NC,SETC2-$   ;MAKE SURE COL# BETWEEN 0 & 79
F6A8  C650      1406         ADD     A,80
F6AA  B5        1407         OR      L            ;MERGE IN COL# WITH L
F6AB  6F        1408         LD      L,A
F6AC  C9        1409         RET
                1410
F6AD  CD72F5    1411 M4TST:  CALL    DISPLA       ;DISPLAY THE CONTROL CHAR
F6B0  C9        1412         RET                  ;PASSED IN C
                1413 ;
                1414 ;
                1415 ;
                1416 ;
                1417         INCLUDE DISKIO.ASM
                1418 ;*************************************************************
                1419 ;*                                                           *
                1420 ;*       DISK INPUT/OUTPUT DRIVER SUBROUTINE PACKAGE         *
                1421 ;*       FOR WESTERN DIGITAL 1771 DISK CONTROLLER            *
                1422 ;*                                                           *
                1423 ;*       bullet-proof error recovery added 12-APR-80         *
                1424 ;*                                                           *
                1425 ;*************************************************************
                1426 ;
                1427 ;
                1428 ;EQUATES FOR DISK CONTROLLER PORTS AND COMMAND CODES
                1429 ;
>0010           1430 STSREG  EQU     WD1771+0     ;STATUS REGISTER
>0010           1431 CMDREG  EQU     WD1771+0     ;COMMAND REGISTER
>0011           1432 TRKREG  EQU     WD1771+1     ;TRACK REGISTER
>0012           1433 SECREG  EQU     WD1771+2     ;SECTOR REGISTER
>0013           1434 DATREG  EQU     WD1771+3     ;DATA REGISTER
                1435 ;
>0088           1436 RDCMD   EQU     10001000B    ;READ COMMAND
>00A8           1437 WRTCMD  EQU     10101000B    ;WRITE COMMAND
>001C           1438 SKCMD   EQU     00011100B    ;SEEK COMMAND
>00D0           1439 FINCMD  EQU     11010000B    ;FORCE INTR COMMAND
>000C           1440 RSTCMD  EQU     00001100B    ;RESTORE COMMAND
>0004           1441 HLOAD   EQU     00000100B    ;RD/WRT HEAD LOAD ENABLE
                1442 ;
>00C9           1443 RET     EQU     0C9H         ;SUBROUTINE RETURN INSTR OPCODE
>0066           1444 NMIVEC  EQU     0066H        ;THE NON-MASKABLE INTERRUPT IS
                1445                              ;USED FOR DATA SYNC BETWEEN
                1446                              ;THE Z-80 AND 1771
                1447 ;
                1448 ;
                1449 ;
F6B1  79        1450 SELECT: LD      A,C          ;GET UNIT# PASSED IN C AND
F6B2  FE04      1451         CP      4            ; CHECK FOR MAXIMUM VALID#
F6B4  D0        1452         RET     NC           ;ERROR IF NUMBER > 3
F6B5  CD88F7    1453         CALL    TURNON       ;MAKE SURE DISKS ARE TURNED ON
F6B8  DB1C      1454         IN      A,(BITDAT)
F6BA  47        1455         LD      B,A          ;SAVE CURRENT DRIVE SELECT DATA
F6BB  E6F8      1456         AND     11111000B    ;MERGE IN NEW DRIVE UNIT# IN C
F6BD  B1        1457         OR      C            ;IN PLACE OF THE CURRENT ONE
F6BE  D31C      1458         OUT     (BITDAT),A   ;TO SELECT THE NEW DISK DRIVE
F6C0  CDAEF7    1459         CALL    FORCE        ;TEST NEW DRIVE'S READY STATUS
```

```
F6C3  2806      1460          JR      Z,SEL2-$    ;AND CONTINUE IF ITS READY
F6C5  78        1461          LD      A,B
F6C6  D31C      1462          OUT     (BITDAT),A  ;ELSE PUT BACK OLD DRIVE SELECT
F6C8  3E80      1463          LD      A,10000000B;AND RETURN DRIVE-NOT-READY
F6CA  C9        1464          RET
                1465
F6CB  2165FF    1466 SEL2:    LD      HL,UNIT     ;POINT HL TO DRIVE SELECT DATA
F6CE  7E        1467          LD      A,(HL)      ;LOAD A WITH CURRENT UNIT#
F6CF  71        1468          LD      (HL),C      ;AND STORE NEW UNIT# FROM C
F6D0  FEFF      1469          CP      255         ;TEST IF NO DRIVE SELECTED
F6D2  2806      1470          JR      Z,SEL3-$    ;YET & SKIP NEXT SEGMENT IF SO
F6D4  23        1471          INC     HL          ;POINT TO HEAD POSITION TABLE
F6D5  85        1472          ADD     A,L         ;AND ADD IN NEW UNIT# AS INDEX
F6D6  6F        1473          LD      L,A
F6D7  DB11      1474          IN      A,(TRKREG)  ;GET CURRENT HEAD POSITION
F6D9  77        1475          LD      (HL),A      ;AND STORE IN TABLE @ HL
F6DA  2166FF    1476 SEL3:    LD      HL,TRKTAB
F6DD  7D        1477          LD      A,L
F6DE  81        1478          ADD     A,C         ;INDEX INTO TABLE TO GET
F6DF  6F        1479          LD      L,A         ;HEAD POSITION OF NEW DRIVE
F6E0  7E        1480          LD      A,(HL)
F6E1  FEFF      1481          CP      255         ;TEST IF NEW DRIVE WAS EVER
F6E3  2804      1482          JR      Z,HOME-$    ;SELECTED AND DO A HOME IF NOT
F6E5  D311      1483          OUT     (TRKREG),A  ;OUTPUT DRIVE'S CURRENT HEAD
F6E7  AF        1484          XOR     A           ;POSITION TO THE TRACK REGISTER
F6E8  C9        1485          RET
                1486 ;
                1487 ;
                1488 ;
F6E9  CDABF7    1489 HOME:    CALL    READY       ;CLEAR DISK CONTROLLER
F6EC  C0        1490          RET     NZ          ;EXIT IF DRIVE NOT READY
F6ED  AF        1491          XOR     A
F6EE  326DFF    1492          LD      (TRACK),A   ;SET TRACK# IN MEM TO ZERO
F6F1  060C      1493 RESTOR:  LD      B,RSTCMD    ;LOAD B WITH A RESTORE COMMAND
F6F3  CD93F7    1494          CALL    STEP        ;EXECUTE HEAD MOVING OPERATION
F6F6  EE04      1495          XOR     00000100B   ;GET TRUE TRACK 0 STATUS
F6F8  E69C      1496          AND     10011100B   ;MASK TO ERROR BITS
F6FA  C9        1497          RET                 ;RETURN 1771 STATUS IN A
                1498 ;
                1499 ;
                1500 ;
F6FB  CDABF7    1501 SEEK:    CALL    READY       ;CLEAR DISK CONTROLLER
F6FE  C0        1502          RET     NZ          ;EXIT IF DRIVE NOT READY
F6FF  79        1503          LD      A,C         ;GET TRACK# DATA FROM C AND
F700  FE4D      1504          CP      77          ;CHECK FOR MAXIMUM VALID#
F702  D0        1505          RET     NC          ;FORGET IT IF TRACK# > 76
F703  326DFF    1506          LD      (TRACK),A   ;ELSE STORE TRACK# FOR SEEK
F706  D313      1507          OUT     (DATREG),A  ;OUTPUT TRACK # TO 1771
F708  061C      1508          LD      B,SKCMD     ;LOAD B WITH A SEEK COMMAND AND
F70A  CD93F7    1509          CALL    STEP        ;GO SEEK WITH PROPER STEP RATE
F70D  E698      1510          AND     10011000B   ;MASK TO READY,SEEK & CRC ERROR
F70F  C8        1511          RET     Z           ;BITS AND RETURN IF ALL GOOD
                1512
F710  CDF1F6    1513          CALL    RESTOR      ;ELSE TRY TO RE-CAILBRATE HEAD
F713  C0        1514          RET     NZ          ;ERROR IF WE CAN'T FIND TRACK 0
F714  79        1515          LD      A,C
F715  D313      1516          OUT     (DATREG),A  ;OUTPUT TRACK# TO 1771
F717  061C      1517          LD      B,SKCMD
F719  CD93F7    1518          CALL    STEP        ;TRY TO SEEK THE TRACK AGAIN
F71C  E698      1519          AND     10011000B
F71E  C9        1520          RET                 ;RETURN FINAL SEEK STATUS IN A
                1521 ;
                1522 ;
                1523 ;
```

```
F71F  CDABF7   1524 WRITE:  CALL   READY       ;CLEAR THE DISK CONTROLLER
F722  C0       1525         RET    NZ          ;EXIT IF DRIVE NOT READY
F723  CB77     1526         BIT    6,A
F725  C0       1527         RET    NZ          ;EXIT IF DISK WRITE-PROTECTED
F726  06A8     1528         LD     B,WRTCMD
F728  1806     1529         JR     RDWRT-$
               1530
F72A  CDABF7   1531 READ:   CALL   READY       ;CLEAR DISK CONTROLLER
F72D  C0       1532         RET    NZ          ;EXIT IF DRIVE NOT READY
F72E  0688     1533         LD     B,RDCMD
F730  2271FF   1534 RDWRT:  LD     (IOPTR),HL  ;STORE DISK I/O DATA POINTER
F733  216EFF   1535         LD     HL,SECTOR
F736  71       1536         LD     (HL),C      ;STORE SECTOR# FOR READ/WRITE
F737  23       1537         INC    HL
F738  70       1538         LD     (HL),B      ;SAVE READ/WRITE COMMAND BYTE
F739  23       1539         INC    HL
F73A  3602     1540         LD     (HL),2      ;SET DISK RE-TRY COUNT
F73C  F3       1541 RW1:    DI                 ;NO INTERRUPTS DURING DISK I/O
F73D  216600   1542         LD     HL,NMIVEC   ;SAVE BYTE AT NMI VECTOR LOCAT
F740  56       1543         LD     D,(HL)      ;IN D FOR DURATION OF READ/WRIT
F741  36C9     1544         LD     (HL),RET    ;LOOP AND REPLACE IT WITH A RET
F743  216BFF   1545         LD     HL,RECLEN
F746  46       1546         LD     B,(HL)      ;B=NUMBER OF BYTES/SECTOR
F747  0E13     1547         LD     C,DATREG    ;C=1771 DATA REGISTER PORT#
F749  2A71FF   1548         LD     HL,(IOPTR)  ;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         ;TO TEST HEAD LOAD STATUS
F756  3A6FFF   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         OR     HLOAD       ;ELSE MERGE IN HLD BIT
F75D  CDA3F7   1556 RW2:    CALL   CMDOUT      ;START 1771 DOING IT'S THING
F760  CB6F     1557         BIT    5,A         ;TEST IF COMMAND IS A R OR W
F762  200D     1558         JR     NZ,WLOOP-$  ;AND JUMP TO THE CORRECT LOOP
F764  76       1559 RLOOP:  HALT
F765  EDA2     1560         INI
F767  C264F7   1561         JP     NZ,RLOOP
F76A  CD9CF7   1562         CALL   BUSY        ;LOOP UNTIL 1771 COMES UN-BUSY
F76D  E69C     1563         AND    10011100B   ;MASK OFF TO READY,NOT FOUND,CRC
F76F  180B     1564         JR     RW3-$       ;AND LOST DATA STATUS BITS
               1565
F771  76       1566 WLOOP:  HALT
F772  EDA3     1567         OUTI
F774  C271F7   1568         JP     NZ,WLOOP
F777  CD9CF7   1569         CALL   BUSY
F77A  E6BC     1570         AND    10111100B   ;MASK OFF AS ABOVE + WRT FAULT
F77C  216600   1571 RW3:    LD     HL,NMIVEC
F77F  72       1572         LD     (HL),D      ;RESTORE BYTE @ NMI VECTOR
F780  FB       1573         EI
F781  C8       1574         RET    Z           ;RETURN IF NO DISK I/O ERRORS
F782  2170FF   1575         LD     HL,RETRY
F785  35       1576         DEC    (HL)        ;DECREMENT RE-TRY COUNT AND
F786  2002     1577         JR     NZ,RW4-$    ; EXECUTE COMAND AGAIN IF NOT=0
F788  B7       1578         OR     A
F789  C9       1579         RET                ;ELSE RETURN 1771 ERROR STATUS
               1580
F78A  216DFF   1581 RW4:    LD     HL,TRACK
F78D  4E       1582         LD     C,(HL)      ;GET TRACK# FOR THIS OPERATION
F78E  CDFBF6   1583         CALL   SEEK        ;TRY TO RE-CAILBRATE THE HEAD
F791  18A9     1584         JR     RW1-$       ;BEFORE READ OR WRITE AGAIN
               1585 ;
               1586 ;
               1587 ;
```

```
F793  3A6AFF    1588 STEP:   LD      A,(SPEED)   ;GET STEP SPEED VARIABLE
F796  E603      1589         AND     00000011B
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
                1596 ;
                1597 ;
                1598 ;
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
                1604 ;
                1605 ;
                1606 ;
F7AB  CDB8F7    1607 READY:  CALL    TURNON      ;KEEP THOSE DISKS SPINING FOLKS
F7AE  3ED0      1608 FORCE:  LD      A,FINCMD    ;ISSUE FORCE INTERRUPT COMMAND
F7B0  CDA3F7    1609         CALL    CMDOUT
F7B3  DB10      1610         IN      A,(STSREG)  ;READ STATUS REGISTER CONTENTS
F7B5  CB7F      1611         BIT     7,A         ;TEST DRIVE NOT READY BIT
F7B7  C9        1612         RET
                1613 ;
                1614 ;
                1615 ;
F7B8  3E1E      1616 TURNON: LD      A,30
F7BA  326CFF    1617         LD      (MOTOR),A   ;RE-LOAD MOTOR TURN-OFF TIMER
F7BD  CDA8F7    1618         CALL    PAUSE
F7C0  DB1C      1619         IN      A,(BITDAT)
F7C2  CB57      1620         BIT     2,A         ;TEST IF MOTORS HAVE STOPPED
F7C4  C8        1621         RET     Z           ;AND EXIT IF STILL TURNED ON
F7C5  E6BB      1622         AND     10111011B   ;ELSE RE-ENABLE DRIVE SELECTS
F7C7  D31C      1623         OUT     (BITDAT),A  ;AND ACTIVATE THE MOTOR RELAY
F7C9  C5        1624         PUSH    BC
F7CA  0600      1625         LD      B,0         ;SET READY LOOP MAX TIMEOUT
F7CC  CDDCF7    1626 TURN2:  CALL    WAIT        ;WAIT 1/93 SECOND & TEST READY
F7CF  2802      1627         JR      Z,TURN3-$   ;EXIT LOOP IF DRIVE READY
F7D1  10F9      1628         DJNZ    TURN2-$     ;ELSE TRY AGAIN UP TO 256 TIMES
F7D3  0609      1629 TURN3:  LD      B,9
F7D5  CDDCF7    1630 TURN4:  CALL    WAIT        ;GIVE ABT 1/10 SEC MORE DELAY
F7D8  10FB      1631         DJNZ    TURN4-$
F7DA  C1        1632         POP     BC
F7DB  C9        1633         RET
                1634 ;
                1635 ;
F7DC  DB1B      1636 WAIT:   IN      A,(CTC3)    ;GET CURRENT CTC3 COUNT VALUE
F7DE  4F        1637         LD      C,A
F7DF  DB1B      1638 WAIT2:  IN      A,(CTC3)
F7E1  B9        1639         CP      C           ;SEE IF CTC3 CHANGED BY 1 COUNT
F7E2  28FB      1640         JR      Z,WAIT2-$   ;AND LOOP UNTIL IT CHANGES
F7E4  18C8      1641         JR      FORCE-$     ;THEN TEST DRIVE READY STATUS
                1642 ;
                1643 ;
                1644 ;
                1645 ;
                1646 ;
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
>FF00           1660 SIOVEC: DEFS    16          ;SPACE FOR 8 VECTORS FOR SIO
>FF10           1661 CTCVEC: DEFS    8           ;SPACE FOR 4 VECTORS FOR CTC
>FF18           1662 SYSVEC: DEFS    4           ;SPACE FOR 2 VECTORS FOR SYSTEM
;                                                  PIO
>FF1C           1663 GENVEC: DEFS    4           ;SPACE FOR 2 VECTORS FOR
;                                                  GENERAL PIO
                1664 ;
                1665 ;
                1666 ;KEYBOARD DATA INPUT FIFO VARIABLES
                1667
>FF20           1668 FIFO:   DEFS    16          ;CONSOLE INPUT FIFO
>FF30           1669 FIFCNT: DEFS    1           ;FIFO DATA COUNTER
>FF31           1670 FIFIN:  DEFS    1           ;FIFI INPUT POINTER
>FF32           1671 FIFOUT: DEFS    1           ;FIFO OUTPUT POINTER
>FF33           1672 LOCK:   DEFS    2           ;SHIFT LOCK CHAR+FLAG BYTE
                1673 ;
                1674 ;
                1675 ;STACK POINTER SAVE AND LOCAL STACK FOR INTERRUPT ROUTINES
                1676
>FF35           1677 SPSAVE: DEFS    2           ;USER STACK POINTER SAVE AREA
>FF37           1678 TMPSTK: DEFS    32          ;LOCAL STACK FOR INTERRUPTS
                1679 ;
                1680 ;
                1681 ;'SOFTWARE' VECTORS FOR INTERRUPT SERVICE ROUTINES
                1682
>FF57           1683 TIKVEC: DEFS    2           ;1 SEC INTERRUPT ROUTINE VECTOR
>FF59           1684 PINVEC: DEFS    2           ;PARALLEL CONSOLE INPUT VECTOR
>FF5B           1685 SINVEC: DEFS    2           ;SERIAL CONSOLE INPUT VECTOR
                1686 ;
                1687 ;
                1688 ;CLOCK-TIMER INTERRUPT VARIABLES
                1689
>FF5D           1690 TIKCNT: DEFS    2           ;BINARY CLOCK TICK COUNTER
>FF5F           1691 DAY:    DEFS    1           ;CALENDAR DAY
>FF60           1692 MONTH:  DEFS    1           ;  MONTH
>FF61           1693 YEAR:   DEFS    1           ;  YEAR
>FF62           1694 HRS:    DEFS    1           ;CLOCK HOURS REGISTER
>FF63           1695 MINS:   DEFS    1           ;      MINUTES RETISTER
>FF64           1696 SECS:   DEFS    1           ;      SECONDS REGISTER
                1697 ;
                1698 ;
                1699 ;DISK I/O DRIVER VARIABLES
                1700
>FF65           1701 UNIT:   DEFS    1           ;CURRENTLY SELECTED DISK#
>FF66           1702 TRKTAB: DEFS    4           ;4 DRIVE HEAD POSITION TABLE
>FF6A           1703 SPEED:  DEFS    1           ;SEEK SPEED FOR 1771 COMMANDS
>FF6B           1704 RECLEN: DEFS    1           ;SECTOR RECORD LENGTH VARIABLE
>FF6C           1705 MOTOR:  DEFS    1           ;DRIVE MOTOR TURN-OFF TIMER
>FF6D           1706 TRACK:  DEFS    1
>FF6E           1707 SECTOR: DEFS    1
```

```
>FF6F           1[redacted] CMDTYP: DEFS    1          ;COMMAND BYTE FOR READS/WRITES
>FF70           1709 RETRY:  DEFS    1          ;DISK OPERATION RE-TRY COUNT
>FF71           1710 IOPTR:  DEFS    2          ;DISK I/O BUFFER POINTER
                1711 ;
                1712 ;
                1713 ;
                1714 ;CRT OUTPUT DRIVER VARIABLES
                1715
>FF73           1716 CURSOR: DEFS    2          ;CURSOR POINTER
>FF75           1717 CHRSAV: DEFS    1          ;CHAR OVERLAYED BY CURSOR
>FF76           1718 CSRCHR: DEFS    1          ;CHAR USED FOR A CURSOR
>FF77           1719 BASE:   DEFS    1          ;CURRENT CONTENTS OF SCROLL
;                                                 REGISTER
>FF78           1720 LEADIN: DEFS    1          ;STATE OF LEAD-IN SEQUENCE
;                                                 HANDLER
                1721 ;
                1722 ;
                1723 ;NULL PAD COUNT FOR SERIAL OUTPUT DELAY
                1724
>FF79           1725 NULLS:  DEFS    1          ;# OF NULLS SENT AFTER CONTROL
;                                                 CHARS.
                1726 ;
                1727 ;
                1728 ;LISTHEAD POINTER FOR DYNAMIC MEMORY ALLOCATION SCHEME
                1729
>FF7A           1730 FREPTR: DEFS    2
                1731 ;
                1732 ;
                1733 ;CONSOLE MONITOR PROGRAM VARIABLES
                1734
>FF7C           1735 PARAM1: DEFS    2          ;STORAGE FOR NUMBERS READ
>FF7E           1736 PARAM2: DEFS    2          ;FROM LINE INPUT BUFFER
>FF80           1737 PARAM3: DEFS    2          ;BY 'PARAMS' SUBROUTINE
>FF82           1738 PARAM4: DEFS    2
>FF84           1739 ESCFLG: DEFS    1          ;CONSOLE ESCAPE FLAG
>FF85           1740 COFLAG: DEFS    1          ;CONSOLE OUTPUT TOGGLE
>FF86           1741 LAST:   DEFS    2          ;LAST ADDRESS USED BY 'MEMDMP'
>FF88           1742 LINBUF: DEFS    64         ;CONSOLE LINE INPUT BUFFER
                1743 ;
                1744 ;
                1745
                1746 ;
                1747         END

ERRORS=0000
```

BIGBOARD MISC. ADDENDA

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 followins 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. Referins to the PFM user's manual, output this hex value to Port OC Hex usins 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.
Selectins a new baud rate MUST is done BEFORE usins 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 srounded.
The component legend on the PC board is incorrect at connector J2. The "14" should read "2", the "Keyboard Connector Pin Assisnments" 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 neslected 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 senerator must be installed at U107.

VIDEO OUTPUT

Some brands of video monitors may require adjustment of the Vertical Heisth 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 Big Board.

IMPORTANT!

When using the Big 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 Big 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 Big Board with ALL I/O options added REQUIRES NO JUMPERS.

U52 IS A 14 PIN DEVICE!
THE SILK SCREEN LEGEND IS NOT CORRECT.

GEORGE RISK INDUSTRIES, INC.

MODEL 753 APPLICATION NOTES

Read carefully before assembling or using your keyboard!

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 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 converter 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 <u>unless</u> 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.
5. 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!
6. 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.

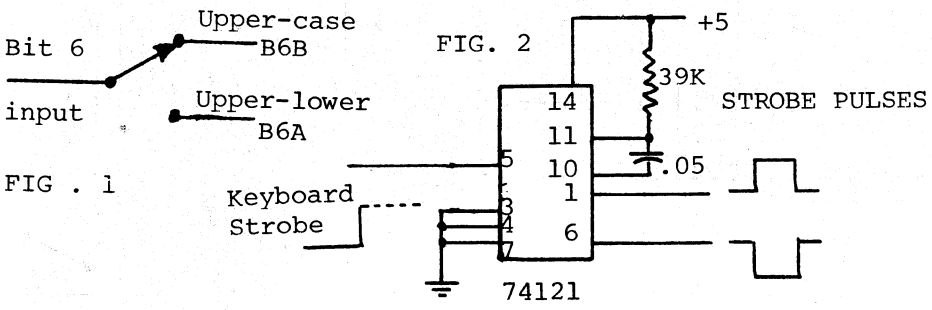
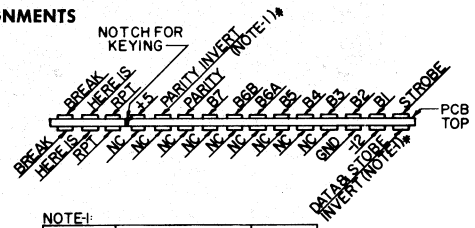


FIG . 1

FIG. 2

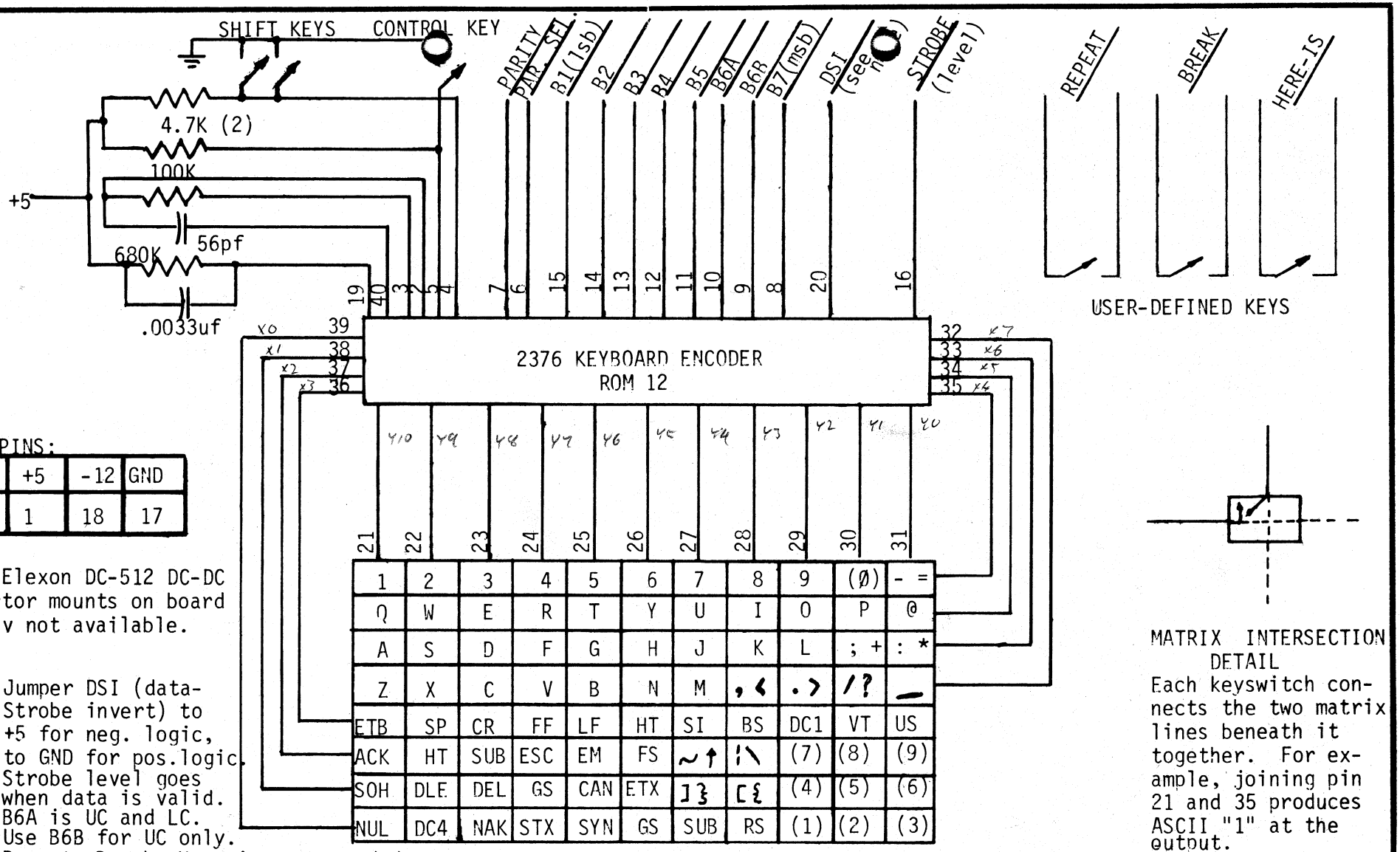
PIN ASSIGNMENTS



NOTE-1:

JUMPER*	DATA AND STROBE OUTPUTS	PARITY OUTPUT
+5	NEG. LOGIC	EVEN
GND.	POS. LOGIC	ODD

2-B6A PROVIDES BOTH UPPER AND LOWER CASE OPTION
B6B PROVIDES UPPER CASE ONLY
3-RPT, BRK., HERE IS ARE EXTERNAL CONTACT CLOSURES.
4-DRIVE CAPABILITY IS ONE TTL UNIT LOAD
5-TYP CONNECTOR CINCH 251-15-30 (NOT INCL.)



POWER PINS:

	+5	-12	GND
AY-5-2376	1	18	17

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

MATRIX INTERSECTION DETAIL

Each keyswitch connects the two matrix lines beneath it together. For example, joining pin 21 and 35 produces ASCII "1" at the output.

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 1/4watt, carbon comp.

NOTES:

DASH NO.	NEXT ASSY	FINAL ASSY	NEXT ASSY	FINAL ASSY
APPLICATION			QTY REQD	

UNLESS OTHERWISE SPECIFIED
DIMENSIONS ARE IN INCHES TOLERANCES ON
FRACTIONS ± 1/64 DECIMALS XX ± .01 XXX ± .005 ANGLES ± 0°30'

MATERIAL | FINISH

GEORGE RISK INDUSTRIES, INC.
KIMBALL, NEBRASKA

SCALE: NONE | APPROVED BY: | DRAWN BY hs
DATE: 1-16-78 | REVISED

ELECTRICAL SCHEMATIC - 753 KEYBOARD

SIZE A | DRAWING NUMBER 2-753-003

SPECIAL INSTRUCTIONS FOR SINGLE SIDED MODEL 753 KEYBOARD

This sheet supercedes all other instructions or specifications. PLEASE READ CAREFULLY!

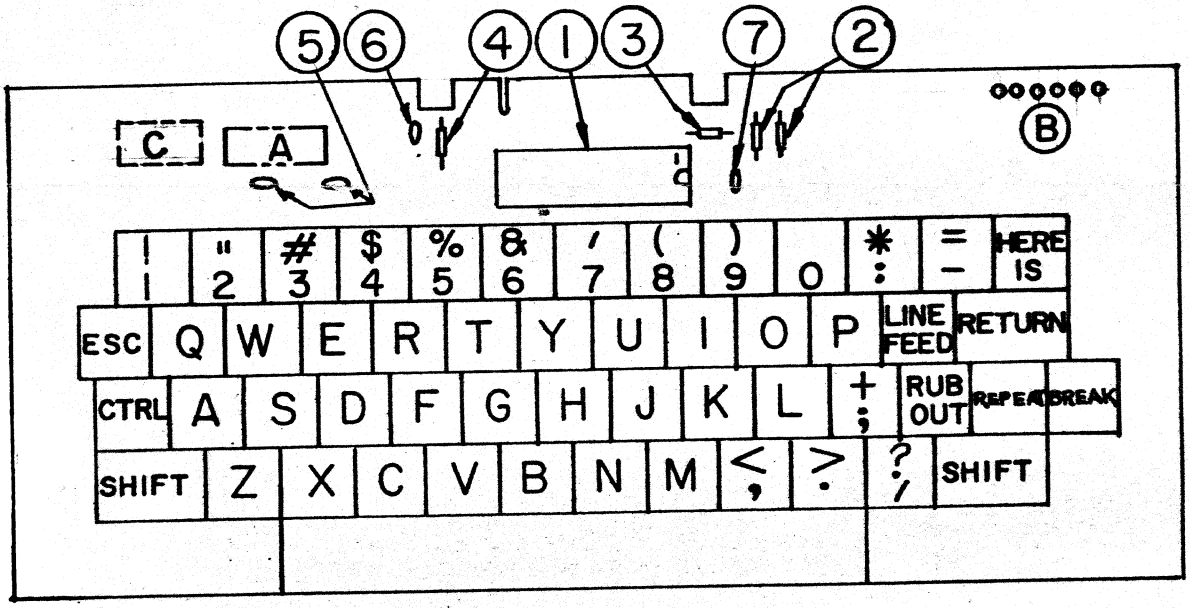
Assembly instructions on your packaging card are NOT correct. Your Model 753 uses a new, single sided printed circuit board, instead of the double sided board shown on the card. Please substitute the instructions on this sheet for those printed on the reverse side of the packaging card. Retain this sheet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

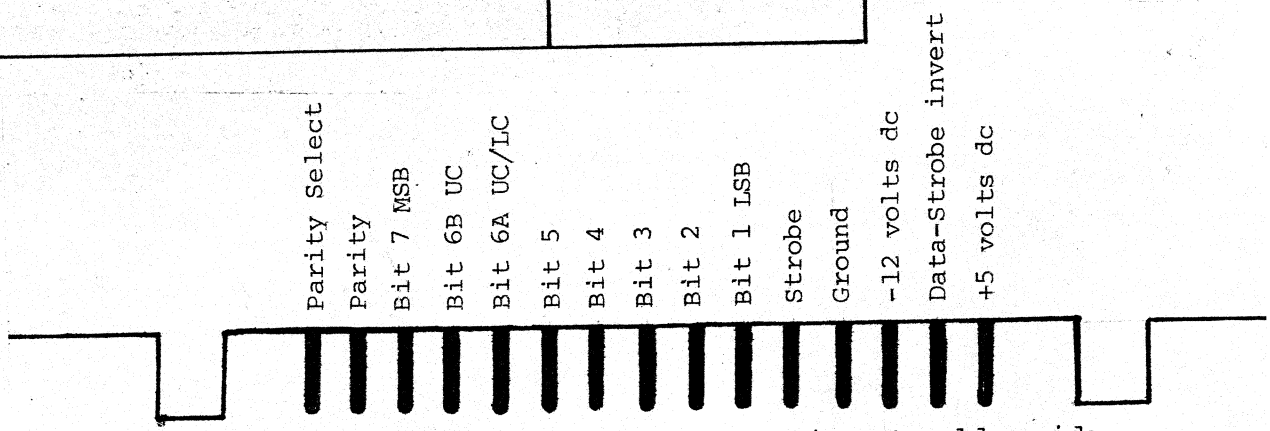
A optional DC512 converter for +5 volt only use

B USER DEFINED KEY termination area

C "Kluge Area"

CARD EDGE CONNECTOR PIN ASSIGNMENTS

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



View of card-edge connector, looking at solder side
-Connect to B6B for Upper case only, to B6A for upper and lower case, selectable by shift keys.
-Jumper Data strobe invert pin to ground for positive logic output, to +5 for negative logic output.

Temporary data sheet
4/79