DUTRONICS P. O. BOX 9160 • STOCKTON, CALIFORNIA 95208

PRICE INCLUDING BINDER $15.00

DZ80-80 CPU MANUAL

PRELIMINARY

DUTRONICS P. O. BOX 9160 • STOCKTON, CALIFORNIA 95208

DZ80-80 MANUAL

TABLE OF CONTENTS

DUTRONICS P. O. BOX 9160 • . STOCKTON, CALIFORNIA 95208

THE DZ80-80 CPU

INTRODUCTION

The DZ80-80 is a 4-inch square "Piggyback" PC card designed to upgrade an 8080/8080A CPU microprocessor based system to a Z-80 CPU system without requiring replacement of the system processor card. The Z-80 CPU is NOT electrically interchangeable with the 8080 CPU and has meant, until the DZ80-80, that to obtain the power of the nearly 690 instruction variations of the Z-80, the 8080 processor card had to be discarded.

Nine integrated circuits and a bevy of passive components provide a network which interface the Z-80 CPU to the system's existing 8080 socket. An umbilical cord connects from the DZ80-80 to the system's existing 8212 status latch socket. Thereby providing Z-80 power by replacing only two ICs.

It is recommended that all included reference material be read prior to the installation of the DZ80-80. Since the Z-80 IC included is an MOS device, improper handling or installation can become an expensive education.

THE Z-80 CPU

Included is the 'Z80 Technical Manual' written by the Z-80 design team. A thorough study and understanding of this Manual is a must to obtain full benefit of Z-80 POWER.

COMPATABILITY

As a subset of the Z-80 instruction set is the 8080 instruction set. Therefore, programs written for the 8080 will execute identically on the DZ80-80 system with one minor exception.

The Parity flag of the 8080 is shared by a new Overflow flag on the Z-80 (see 'Z80 Technical Manual' for description). Some sophisticated software writers have been known to store information in the Parity flag and certain arithmetic instructions cause the Parity flag to react differently on the Z-80 than the 8080. Therefore, in one or two rare instances, where the Parity flag is used for other than Parity, a minor incompatibility could exist (ALTAIR Basic is one rare instance). This is the only inconsistency found after extensive research.

Another difference between the DZ80-80 and the 8080 is that there is no provision for STACK status. As of this writing, no known hardware is available that would be inoperative without STACK status.

As a consolation the DZ80-80 provides the user an option to connect the STACK status line to the Z-80 Refresh signal, thereby allowing the DZ80-80 to perform all necessary refreshing of the system's dynamic memory.

One final note on compatability is when operating the DZ80-80 in an IMSAI, ALTAIR or other systems with a hardware front panel that is supposed to stop (when STOP is pressed) on an M1 cycle only, the DZ80-80 may stop on any random machine cycle. This occurs when the front panel samples the data lines during SYNC to decode M1 status rather than using the STATUS lines themselves. The DZ80-80 does not place STATUS on the data lines.

Panel switches EXAMINE, EXAMINE NEXT, DEPOSIT and DEPOSIT NEXT do not operate correctly unless the processor is in an M1 cycle. It is thus required to single step the processor to an M1 cycle before operating the previously mentioned panel switches after a STOP. (RESET while STOP will always generate an M1 cycle.)

This idiosyncrasy has been found not to be a problem once the operator becomes used to checking for M1 before pressing EXAMINE. It was felt that the extra cost that would have been incurred by the end user did not warrant the addition of hardware to eliminate this inconvenience.

DZ80-80

THEORY OF OPERATION

As noted in the 'Technical Manual' the Z-80 does not provide many of the signals required for the operation of an 8080 system. Namely SYNC, INTE, DBIN, INTA, OUT, INP and MEMR had to be generated from the Z-80 System Control Signals $\overline{IORQ}$, $\overline{MREQ}$, $\overline{RD}$ and $\overline{M1}$.

The system Φ2 clock was chosen to generate the Φ clock for the Z-80 since the specification for Φ2 is compatible with the Φ clock specification and no system timing change occurs for this choice. Φ2 is a 12v clock, unlike the 8080 the Z-80 requires a single 5V supply and no high voltage clocking. Diode CR1 and resistor R1 shift the Φ2 clock to a 5V signal which is double inverted by IC5 and derives Φ with pull-up resistor R2. R2 is included to insure that Φ has a High of 5V as required by the Z-80.

System SYNC (beginning of each machine cycle) is created as one Φ period beginning when both IORQ and MREQ are False by NAND gate IC3, inverter IC5 and JK flip flop IC6 clocked by Φ. IC5 output, $\overline{SYNC}$, is used to gate $\overline{WR}$ and $\overline{WO}$ to insure that time is available for the system to decode OUT status before WR becomes active during a write operation.

Status INP is the AND of RD and IORQ implemented by NOR gate IC7. Status OUT is IORQ ANDED with WR using NOR gate IC4. The status signal INTA is M1 AND IORQ with NOR gate IC4 acting as the AND function.

Status MEMR is formed with NOR gate IC7 as RD AND MREQ. The remaining two implemented status signals, M1 and HLTA, are merely the inversion of Z-80 outputs $\overline{M1}$ and $\overline{HALT}$ by IC2 and IC3, respectively.

The Z-80 does not provide any indication when it is performing a STACK operation, therefore, the STK status has not been provided. The DZ80-80 is assembled with a jumper from the STACK status input to ground. At the user's option this jumper may be connected to the Z-80 $\overline{RFSH}$ output thus providing the system with automatic dynamic memory refresh. See 'Z80 Technical Manual' for a discussion of this subject.

No external indicator is provided by the Z-80 as to the state of the internal Interrupt Flip-Flop. Thus NAND gate IC1, Inverters (2) IC2 and NORs (2) IC4 decode each EI and DI instruction on the falling edge of M1 and store this information in Flip-Flop IC6 providing the INTE signal. System

RESET or Status INTA will set IC6 through NOR gate IC7 indicating INTE False. IC7 then parrots the state of the internal Z-80 Interrupt Flip-Flop.

DBIN is implemented as RD OR INTA by NAND gate IC3 and NOR gate IC7. Notice that DBIN is True also during System RESET. This is not a system requirement but included only to save an IC package. It was determined that DBIN True at RESET time would not degrade performance and create a physically smaller DZ80-80 assembly.

A potentially powerful feature of the Z-80 is its handling of the high order address lines during I/O operations, refer to the 'Technical Manual' for a discussion. Most existing 8080 systems, however, have used A8 through A15 for I/O addressing and this feature could not be included in the DZ80-80. (If this feature is desired IC8 and IC9 can be removed and A8 through A15 strapped straight through.)

Multiplexers IC8 and IC9 are connected between the Z-80 and 8080 system address lines such that during status INP or OUT NOR Gate IC7 causes A8 through A15 to contain the same data as A0 through A7.

The Z-30 $\overline{\text{NMI}}$ (Non-Maskable Interrupt) line has been brought to a solder pad on the DZ80-80 so the user may connect this to, say VI0. This connection would mean VI0 is the ultimately highest priority interrupt.

As noted on the DZ80-80 schematic the remainder of the Z-80 to 8080 system interface is either by straight connection or by simple inversion and need not be dwelled upon.

With the exception of the eight status lines (and $\overline{\text{NMI}}$) all DZ80-80 connections are made through the system's 8080 socket. Connector J2 provides the connection of the status to the system via Plug P2, 8 conductor flat cable, 24-pin connector, connector J3 to the system 8212 status latch socket (8212 is removed). The flat cable is permanently attached and wired to J3 pins 4, 6, 8, 10, 15, 17, 19, and 21, the output pins of the 8212.

Marked on J2 are two different positions P2 can be plugged in, POSition A and POSition B,

J2 Pins 9 and 10 unconnected,

J2 Pins 1 and 2 unconnected,

respectively.

There is no conformed to standard which sets the status signal to 8212 pin relationship. However, two different and often used pin-outs have been observed and those are included as POSition A and POSition B:

POS A - ALTAIR Position

POS B - IMSAI Position

In the event another pin-out is required, the pins of plug P2 maybe removed and scrambled to fit the user's requirements (see INSTALLATION and CHECKOUT section).

INSTALLATION AND CHECKOUT

INSTALLATION CAUTION

THE Z-80 CPU IN THE DZ80-80 IS AN MOS DEVICE AND IT IS IMPERATIVE THAT THE MOS PRECAUTIONS ON THE REVERSE SIDE OF THE PARTS LIST BE FOLLOWED. THE DEVICE WARRANTY WILL BECOME VOID IF THESE ARE NOT ADHERED TO.

1. If the DZ80-80 was obtained as a semi-kit (integrated circuits not installed in their sockets and shipped in separate containers) remove J3 from IC10 socket (shipped there to protect the pins) by carefully prying it up with a small screwdriver or pen knife. DO NOT remove the conductive foam from J1 until ready to install the DZ80-80 on the processer card. Skip to Step 4 if the IC's are already installed.

2. Install IC1 through IC9 in their sockets as directed by the DZ80-80 PC card legend silk screen, the dot indicates Pin 1. Refer to DIP INSERTION on the reverse side of the Parts List.

3. Install IC10, the Z-80 CPU, observing MOS PRECAUTIONS.

4. Turn the Process System Power Switch to OFF and remove the Processor Card.

5. From the Processor Card remove the 8080 CPU IC (store in a static free carrier) and the 8212 IC. If the 8212 is soldered-in, cut off each pin at the IC and unsolder the pins one at a time. (It is NOT recommended that the 8212 be saved by desoldering it. There is too big a change of board damage unless the proper desoldering tools are available.) Install a 24-pin socket if required.

6. Orient the DZ80-80 connector J1 over the processor card CPU socket (Observe Pin 1 orientation) and check for any bypass capactiors that may interfere with the DZ80-80 installation. Bend these over on the processor card as required.

7. Attach P2 to J2 with top of P2 showing (no holes) as follows: (Pins 9 and 10 of J2 not connected) POS A - ALTAIR configuration; (Pins 1 and 2 of J2 not connected) POS B - IMSAI configuration. Skip to Step 8 if interfacing to ALTAIR or IMSAI.

When it is required to interface to other than the two systems shown, it will be necessary to reconfigure P2 by removing its pins and re-inserting them in the dictated order. Refer to the Processor Card documentation and check the signal name to 8212 output pin relationship and compare to the table below.

J2 Pin Number	Status Signal	Flat Cable Wire Color	J3 Pin Number (8212 Output Pin)
1	INTA	GRY	15
2	$\overline{WO}$	PUR	17
3	OUT	BLU	10
4	MI	GRN	8
5	HLTA	YEL	21
6	STACK	ORG	19
7	MEMB	RED	4
8	INP	BRN	6
9	INTA		
10	$\overline{WO}$		

The P2 pins are removed by pressing on the locking tab and sliding them out. (Not an easy operation, but possible.) Re-insert the P2 Pins such that the order matches the 1 to 8 order of J2 as seen in Table left, above. Connect P2 in POS A.

8. Remove the conductive foam from the DZ80-80 J1 (save and reinstall any time the unit is not connected to the processor card) and observing MOS PRECAUTIONS install the DZ80-80 in the 8080 socket.

9. Connect J3 to the 8212 socket, insure proper Pin 1 orientation, pin numbers are molded into J3.

10. Recheck all previous steps, any error could result in Z-80 damage.

11. Install the Processor Card in the system and dress the flat cable. It may be necessary to leave one blank card slot in front of the Processor Card if there is interference.

CHECKOUT

A quick check of operation can be made by testing the front panel functions DEPOSIT and EXAMINE. With these operational the

DUTZ MONITOR can be loaded and the system given a workout.

In the event the system does not perform as indicated on initial start-up, power down immediately and recheck methodically every step of the installation procedure. Performance will be unpredictable if the status cable or ICs are installed incorrectly.

If all attempts at curing a problem fail, contact the supplier or the Factory for assistance.

THE DUTZ MONITOR

GENERAL

V1.0 of DUTZ MONITOR is a 1K page relocatable (able to be loaded at the beginning of any 1K memory page) program to be used for initial DZ80-80 check out and as a system and program debug and evaluation tool. DUTZ V1.0 executes thirteen commands and is capable of expansion limited only to memory availability and the users desires and ingenuity.

It has been assumed that the minimum user I/O configuration will be a Teletype with paper tape option. Two sets of I/O drives are included to allow for a Command Console I/O and File I/O.

The Command I/O is the Input-Output device from which the user issues and receives response from the instruction Command Set. This device is a keyboard input I/O such as a Teletype or CRT with keyboard.

The File I/O is a serial Input-Output device such as a paper tape reader and punch or a tape recorder. The I and O Commands use the File drivers and the remainder use the console drivers. The MONITOR as received is, as indicated in the MONITOR LOADING section, for Console and File I/O to be the same device, but may be changed to the users configuration.

COMMAND SET

The following 13 one or two character commands direct the DUTZ V1.0 Monitor to perform the described operations.

Definitions:

TR - Terminator any of CR (carriage return) or SP (space bar) or , (comma) or ↑ (^).

(A1) or (A2) or (A3) - *4 Hexidecimal (Hex) digit memory address, will default to zero if none entered, if more than four digits are entered the last four will be used.

(OA) - Same as above except optional (need not be entered). When option is not taken the preceeding TR must be replaced with a CR.

---

*If other than Hex digits are entered, a BELL is output to the Command Console and the character is ignored.

(H) - *2 Hex digit number, if one entered a preceding zero is assumed, if more than one is entered the last 2 will be used.

COMMAND	DESCRIPTION
Z	Abort present directive and return control to Monitor, a ? will be output to the console indicating an abort and on a new line the prompter Z will be displayed to indicate the monitor is ready for a new command. Note: Z is not effective during an I (Input) operation (command I/O could be the same as File I/O) an abort is automatically entered if there is no File data received for about 3 seconds.
D	D(A1)TR(A2)TR DUMP memory locations (A1) through (A2) on the command device. Each location is presented as two Hex digits 16 per line with each line beginning with the Hex address of the first location in that line.
E	E(A1)CR EXECUTE starting at memory location (A1), if the program that is executed ends with a RET instruction control will be returned to the Monitor.
H	H(A1)TR(A2)TR HEX arithmetic, the following will be display: $\lceil(A1) + (A2)\rceil$, $\lceil(A1) - (A2)\rceil$
IB	IB(A1)TR(OA)CR INPUT BINARY file starting at memory location (A1) and Abort if location greater than (OA) is reached. If the Optional (OA) is not used CR must replace TR. Place the Binary File (Paper tape etc.) in the File Input device starting with the NULL leader, input the command and start the device. The BELL will sound once to indicate the file is loading. (See Appendix for Binary File format.)

*If other than Hex digits are entered, a BELL is output to the Command Console and the character is ignored.

COMMAND	DESCRIPTION
IL	IL(OA)CR INPUT LOADER file (also called HEX file or Intel format file, see Appendix for format). If the Optional Address (OA) is included (OA) will be added to the address indicated on the file to compute the load address of the data as it is loaded, thus allowing the user to load a file anywhere in memory. This command will load either check summed or non check summed Loader files. An abort is executed if a check summed file contains a check sum error or a non Hex character other than a: Record cue in the correct position. Use the IB load procedure for IL. Command will return to the Monitor when an EOF is found. (See Appendix).
M	M(A1)TR MODIFY memory location (A1). The Monitor will respond with the address (A1) followed by the two Hex digit value stored at (A1) and wait for the users next entry. To modify the displayed location enter (H)SP and the Monitor will respond with a * to indicate the location has been altered and display the address and value of the next sequential location and wait. If only SP is entered the next location will be displayed without altering the first. Return to Monitor can be accomplished at any time by entering CR or Z. To display the location one less than the one displayed enter ↑ (^). No alteration of memory will occur on a ↑ command.
OB	OB(A1)TR(A2)CR OUTPUT BINARY format to the File I/O from memory address (A1) thru (A2). (See Appendix for format) Command may be aborted by entering Z. Enter the command except for the CR, turn on the File Output device (punch etc.) and enter CR.

COMMAND	DESCRIPTION
OL	OL(A1)TR(A2)TR(OA)CR OUTPUT LOADER format to File I/O device from memory location (A1) through (A2), if (OA) is included output an EOF containing (OA) (for ID or starting address), if (OA) is not included no EOF or Null Trailer will be output. (More records to come.) Command may be Aborted by entering Z. Follow the OB procedure.
R	R(A1)TR(A2)TR(A3)CR RELOCATE memory locations (A1) through (A2) to (A3). CAUTION: This command has the potential of overlaying the MONITOR with GARBAGE, double check (A1), (A2) and (A3) before entering the final CR. As a partial guard against error the Monitor will abort if (A2) is less than (A1).
S	S(A1)TR(H)TR SEARCH memory starting at location (A1) for character (H). When (H) is found the M (MODIFY) routine is entered displaying the address and (H). All function of M then become active. To return Searching enter CR, to return to Monitor enter Z. An automatic return to Monitor occurs only if (H) is not contained anywhere in memory.
T	T TR TOP Stack. The address of the top of the stack is displayed by this command. The Monitor automatically assigns this value on every Abort (Z) or at load time as the highest working memory location less than the starting address (Load Address) of the Monitor.
V	V(A1)TR(A2)CR VERIFY memory locations (A1) through (A2) for hardware errors. Upon finding an error the M (modify) routine is entered and the error location is displayed with all functions of the M routine active; the user can then

COMMAND	DESCRIPTION
V (continued)	evaluate the nature of the error. To complete verification enter CR, to return to the Monitor before address (A2) is reached enter Z. This command is not intended to be used as a comprehensive memory test, it may be used to locate gross memory errors such as a stuck bit, protect on, or no memory at an address. DO NOT Verify the memory containing the Verify routine as it can modify itself.

LOADING

The Monitor has been punched on paper tape in a modified Hexidecimal Format preceded by a Binary Format Relocating Load Routine. The Loader is bootstrapped in using the following 21 word Binary Loader.

```
0000  21 AF 01        LXI H,01AFH   REVERSE LOAD ADDR
0003  DB 00    INCH   IN STP        STATUS PORT
0005  EE FF          XRI PMK       FF FOR RDAV TRUE
0007  E6 40          ANI BMK       BIT MASK
0009  20 F8          JRNZ INCH     JMP NONE AVAILABLE
000B  DB 01          IN IPT        INPUT PORT
000D  BD             CMP L         TEST FOR CUE
000E  28 F3          JRZ INCH      JMP STILL CUE
0010  2D            DCR L
0011  77            MOV M,A       STORE IT
0012  20 EF          JRNZ INCH     JMP NOT DONE
0014  E9             PCHL          EXECUTE REL LOADER
```

The preceeding BOOT must be manually loaded at 0000. The user must supply the proper Status Port (STP), Polarity Mask (PMK) (FF for Data Available-True, 00 for Data Available-False), Bit Mask (BMK) and Input Port (IPT) for the loading device.

Once the BOOT is entered and verified:

1. Select the desired load address on the processor panel sense switches. (Only SSA10 through SSA15 are sampled to obtain a 1K page boundry.) The Monitor was designed to be located at the highest available memory location, but maybe located at any 1K boundary except 0000 (0000 contains the Relocating Load Routine).

2. Press Processor RESET.

3. Place the DUTZ MONITOR tape in the Reader with the Hex AF record cue under the read head.

4. Start Reader.

5. Press processor RUN.

The section of tape following the Hex AF record cue is the Relocating Loader which is read in at location 0100 and self-relocates to 0000 around the input routine manually loaded.

Once the Loader relocates to 0000 the INTE LED on the processor panel will light indicating that the Loader has entered correctly. If the INTE LED does not light at the section of NULLS on the Monitor tape (about 2 feet in) the Loader did not get in correctly and the whole Load Procedure must be repeated.

Once the tape has read to the NULLS and the INTE LED is lighted a checksumed Relocatable Hex File is being read. At this time an error is indicated by the flashing of the INTE LED (about 2 cycles per second). This can occur for the following reasons:

1. The Sense Switches are set at zero (remember SSA8 and SSA9 are not used).
   REMEDY: Stop Reader, set switches to a non-zero value, back tape to NULL, restart reader and press RESET while reader is still reading NULLS.

2. Due to a read error, a non-Hex or wrong character is read indicated by a checksum or non-Hex character error.
   REMEDY: Stop Reader, back tape 2-3 feet (a guess of where the record before the error occurred is located), restart Reader and press RESET.

A properly loading program is indicated (not positively) by a non-flash lighted INTE LED.

Chances are quite good that DUTZ MONITOR as received will not have the Command Console and File I/O configuration required by the user's Processor System. Each user will have made assignments to fit system needs that may follow no universal standard (even if there were one).

With this in mind the EOF (End Of File) indicator on the DUTZ MONITOR tape has been separated from the end of the last record by about 6 inches of NULLS. The EOF causes the Loader to branch to the beginning of the MONITOR for automatic start-up.

When the tape reaches these NULLS the reader may be stopped and a patch tape inserted to patch the four I/O drives to match the user's system assignments. If the user doesn't stop the tape before EOF is read and the system is set for different I/O and Status Ports, Polarity and Bit Masks the program will be in a waiting loop. At this point the I/O drivers may be modified via the Front Panel Switches or the patch tape maybe put in the reader and with NULLS being read followed a press RESET. If the patch contains an EOF the MONITOR will sign on with a ? followed by the MONITOR name and version number.

See Appendix for patch tape preparation.

I/O ASSIGNMENT

The following table defines the I/O assignments of the DUTZ MONITOR as received and the addresses and values to be changed when reconfiguring the MONITOR to fit the user's system.

FUNCTION	ADDRESS*	VALUE AS RECEIVED	CHANGE TO
Console Input Status Port	0004	00	Desired Port
Console Data Available Polarity Mask	0006	FF	FF for True High 00 for True Low
Console Data Available Status Bit Mask	0008	40	Set to zero all bits but the desired (uses an ANI instruction)
Console Input Port	000B	01	Desired Port
Console Output Status Port	002E	00	Desired Port
Console Output Not Busy Polarity Mask	0030	FF	FF for True High 00 for True Low
Console Output Status Bit Mask	0032	80	Set to zero all bits but the desired (uses an ANI instruction)
Console Output Port	0037	01	Desired Port
File Input Status Port	003E	00	Desired Port
File Data Available Polarity Mask	0040	FF	FF for True High 00 for True Low
File Data Available Status Bit Mask	0042	40	Set to zero all bits but the desired (uses an ANI instruction)

APPENDIX

FUNCTION	ADDRESS*	VALUE AS RECEIVED	CHANGE TO
File Input Port	0054	01	Desired Port
File Output Status Port	0019	00	Desired Port
File Output Not Busy Polarity Mask	001B	FF	FF for True High 00 for True Low
File Output Status Bit Mask	001D	80	Set to zero all bits but the desired (uses an ANI instruction)
File Output Port	0022	01	Desired Port

---

* Add to these addresses the relocation factor (Sense Switch Value) when modifying by hand. When modifying with a Patch Tape, the Loader will add the relocation, so punch the address shown.

PATCH TAPE PREPARATION

A patch tape is prepared by punching a tape with one or more of the following:

PATCH NUMBER (Don't punch reference only)		NOTES	
		Replace XX in HEX by	Replace YY in HEX by
1	:01000403XXYY	Console Input Status Port Number	-(08 + XX)
2	:01000603XXYY	Console Input Polarity	-(0A + XX)
3	:01000803XXYY	Console Input Status Bit Mask	-(0C + XX)
4	:01000B03XXYY	Console Input Port Number	-(0F + XX)
5	:01002E03XXYY	Console Output Status Port Number	-(22 + XX)
6	:01003003XXYY	Console Output Polarity	-(34 + XX)
7	:01003203XXYY	Console Output Status Bit Mask	-(36 + XX)
8	:01003703XXYY	Console Output Port Number	-(3B + XX)
9	:01003E03XXYY	File Input Status Port Number	-(42 + XX)
10	:01004003XXYY	File Input Polarity	-(44 + XX)
11	:01004203XXYY	File Input Status Bit Mask	-(46 + XX)
12	:01005403XXYY	File Input Port Number	-(58 + XX)
13	:01001903XXYY	File Output Status Port Number	-(1D + XX)

PATCH NUMBER (Don't punch reference only)		NOTES	
		Replace XX in HEX by	Replace YY in HEX by
14	:01001B03XXYY	File Output Polarity	-(1F + XX)
15	:01001D03XXYY	File Output Status Bit Mask	-(21 + XX)
16	:01002203XXYY	File Output Port Number	-(26 + XX)

Only those I/O assignments that differ from the MONITOR Tape as received need be patched. As an example suppose that a particular system with Teletype I/O only requires the following I/O configuration:

A. Input and Output Status Port : Port 00

B. Input and Output Port : Port 01

C. Read Data Available : Bit 4, Low

D. Transmit Buffer Empty : Bit 5, Low

A check of the I/O Assignment Section of this Appendix shows that the MONITOR is now set for

A. Port 00
B. Port 01
C. Bit 6, High
D. Bit 7, High

Therefore only C and D need be changed. Scanning the Patch List it is seen that this requires Patch Nos. 2, 3, 6, 7, 10, 11, and 14, 15.

Patches 2 and 10 set the Status Bit Polarity, the MONITOR requires an active low indicator thus the status word is exclusive OR'd with either 00 or FF to achieve this. Therefore both Patches 2 and 10 require an 00 substituted for the XX in the list. Next the check sum YY must be calculated, this is defined as the negative (ignoring carry outs, i.e. maintain an 8-bit word) of the sum of the HEX digit pairs (1 Byte) in the record.

The Patch List shows the sum of the bytes with the exception of XX, i.e. for Patch 2:

```
  01
  00
  06
  03
  XX
 ----
  0A + XX
```

thus YY = - (0A + XX) = - (0A + 00), for XX = 00 and YY = -(0A) = F6 and Patch 2 becomes

:0100060300F6

and likewise Patches 3, 6, 7, 10, 11 and 14, 15 are

Patch 3	:0100080310E4
Patch 6	:0100300300CC
Patch 7	:0100320320AA
Patch 10	:0100400300BC
Patch 11	:0100420310AA
Patch 14	:01001B0300E1
Patch 15	:01001D0320BF
EOF	:000000

In the preparation of the above example the DUTZ MONITOR H command was used to calculate the checksum, we cheated!

The foregoing procedure is tedious and time consuming, however, once completed and verified it need not be repeated until the system configuration is changed and the MONITOR will automatically start after loading.

One way to limit the amount of work required is to calculate by hand the Console Patches and get the MONITOR on line and use it to aid in the calculation of the File Patches.

FILE RECORD FORMATS

1. Binary Record read by DUTZ IB command or Output by OB command:

   Record Cue (identify beginning of record) - 4 or more FF (all 8 bits 1 words as single binary 8-bit word per FF.

   Record Data - Single binary 8-bit word per data word, a direct copy of the binary data word.

   EOF (End of File, in this case Record) - Exactly 8 binary FF words as in Cue.

2. HEX or Loader Record read by DUTZ IL command or Output by OL command:

   Record Cue -:(single word 3A in binary)

   Header - NN (2-ASCII HEX characters defining the number of Data Bytes in the record in Hexidecimal)

   AAAA (4-ASCII HEX characters defining the 2-byte starting address the data is to be stored.)

   Record Type - 00 (2-ASCII zeros, absolute record type 0)

   Data - HH (2-ASCII HEX characters per byte of data)

   Checksum - CC (2-ASCII HEX characters equal to the negative of the binary sum of all of the Data bytes the Record Type bytes and the three Header bytes during the summation all carry outs are ignored, i.e. modulo 256.)

3. DUTZ Sense Switch Relocatable Record read by relocating BOOT STRAP LOADER.

   Record Cue -:(single word 3A in binary)

   Header - Same as Loader Record

   Record Type - 00 or 03 (In ASCII)

   Data - Same as Loader Record except an ASCII R is the first character of a Data byte that is to be relocated (the value of the 6-MSB Sense Switches are added to the Data bytes before storing.)

Checksum - Same as Loader Record with R taken as 0.

4. End of File (EOF) Record indicating EOF of a string of Loader or Relocatable Records.

Record Cue - Same as Loader Record

Header - 00 (2-ASCII zeros, no-data)

AAAA (4-ASCII HEX characters indicating program entry address or program identification, most often zeros).

DUTRONICS P. O. BOX 9160 • STOCKTON, CALIFORNIA 95208

STATEMENT OF WARRANTY

DUTRONICS, in recognition of its responsibility to provide quality components and adequate instruction for their proper assembly, warrants its products as follows:

All components sold by Dutronics are obtained through recognized factory distribution channels and any part which fails due to defects in manufacture or material will be replaced on an exchange basis, free of charge, for a period of 90 days following the date of purchase.

Any malfunctioning module returned to Dutronics within the warranty period, which in the judgement of Dutronics has been installed and used with care and not subjected to electrical or mechanical absue, will be restored or replaced at Dutronics discretion and returned, with a minimal charge to cover packaging and shipping.

This warranty is made in lieu of all other warranties expressed or implied and is limited in any case to repair or replacement of the module involved.

DUTRONICS P. O. BOX 9160. • STOCKTON, CALIFORNIA 95208

DZ80-80 SEMI-KIT PARTS LIST

1 each	DZ80-80 Manual
1 each	Assembled (No ICs installed) DZ80-80 PC Card
1 each	IC1 - 74LS30 IC
1 each	IC2 - 74LS04 IC
1 each	IC3 - 74LS00 IC
2 each	IC4, IC7 - 74LS02 IC
1 each	IC5 - 7404 IC
1 each	IC6 - 74LS112 IC
2 each	IC8, IC9 - 74LS157 IC
1 each	IC10 - Z80 CPU (MOS)
1 each	Z02 Status Cable Assembly
1 each	1-1/2 inch Binder

DZ80-80 SPECIFICATIONS

Size: 4 inches x 4 inches (10.16 cm x 10.16 cm)

Weight: 3 oz (85.05 gm)

Operating Temperature: 0 to 70°C.

Power Requirements: VCC = 5V ± 5%

ICC = 190mA TYP, 280mA MAX,
Net increase when replacing 8080A and 8212 = 50mA TYP

Interface: J1 - 8080/8080A Pin-Out
J3 - 8212 Pin-Out

DUTRONICS P. O. BOX 9160 • STOCKTON, CALIFORNIA 95208

CONSTRUCTION HINTS

SOLDER DO'S AND DON'TS

1. MOST IMPORTANT: The solder supplied with your kit is the highest quality 60/40, resin-core solder. DO NOT under any circumstances use acid core solder or paste or liquid flux. Any of these will cause unrepairable damage to your PC board and components.

2. DO use a small diameter, low wattage soldering iron, 18 watts is sufficient--no greater than 25 watts. DO NOT use a soldering gun, they are too hot and clumsy!

3. DO keep your soldering iron clean. A damp sponge is a handy cleaning aid.

4. DO NOT PRESS a hot iron on a PC board land or trace. If you do you will watch the copper peel off the board!

5. DO keep the working part of the iron tip well tinned. Touch tip of iron to parts to be soldered while applying solder to combination, solder will flow and heat joint. Use enough solder to just cover the contour of the joint, remove solder and leave iron until resin boils out, about 2–3 seconds. ICs are speced at a max soldering time of 10 seconds.

6. DO be careful of solder bridges between traces -- bridges are the most frequent problem when the initial equipment test fails -- even with professionals! After every step in assembly, carefully inspect for bridges and missed solder points -- the second most frequent problem. Use a solder wick or vacuum bulb to remove bridges and blobs.

DIP INSERTION

1. Dual In-line Packages are uniquely embossed or marked to indicate Pin No. 1. This marking can take the form of a small dot over Pin 1, elongated half circle or full circle at the end of the package containing Pin 1. Whatever the marking, it will be over Pin No. 1 when the DIP is oriented such that the marking is on the left. Pin count then increases counterclockwise around the DIP.

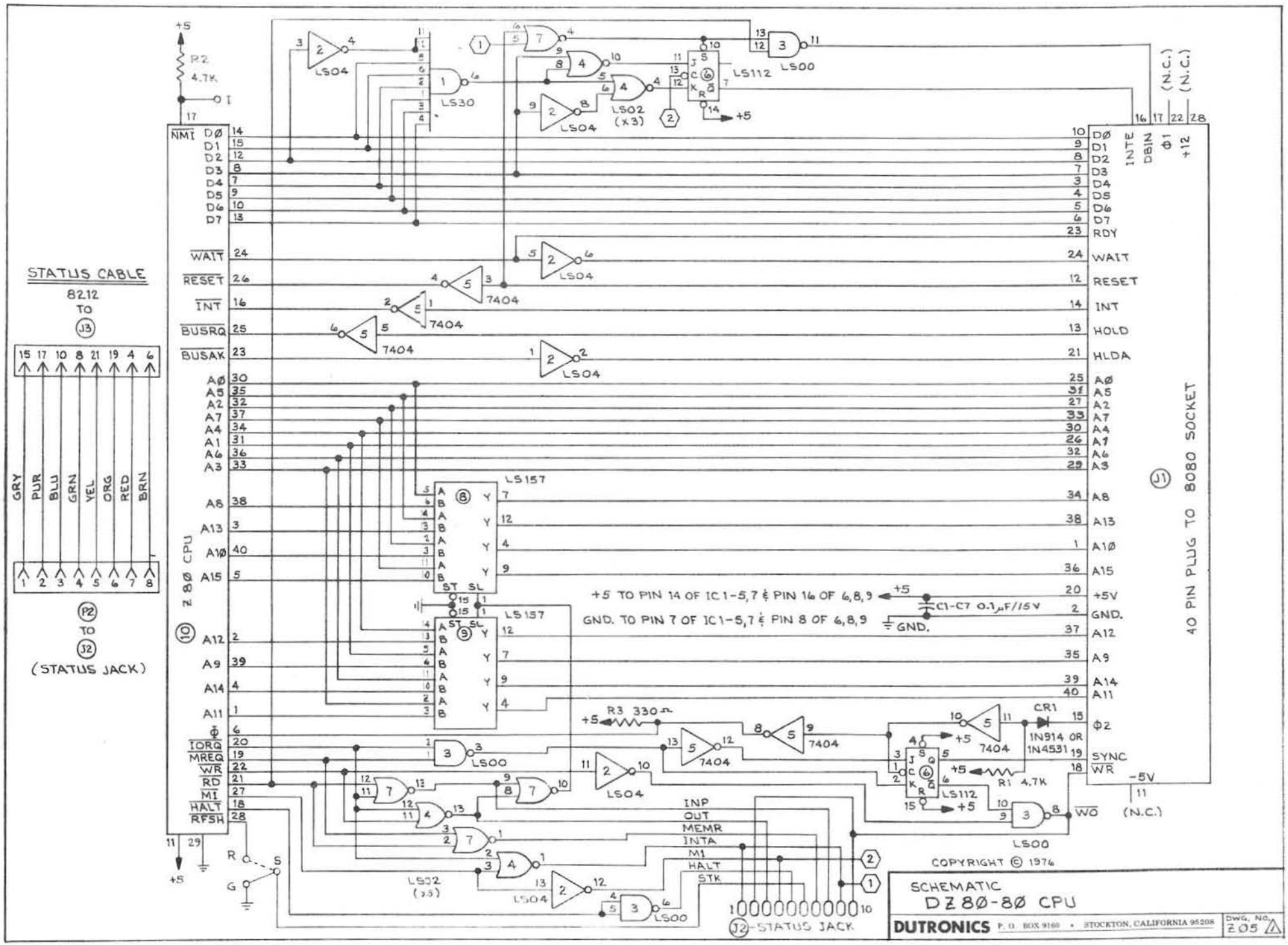
2. Most DIP packages are manufactured with the pin spacing greater than the board lay-out spacing to facilitate automatic machine insertion and soldering. If you have access to an insertion tool, use it; if not, insert the pins on one side about 1/3 and exert pressure on the opposite side until the pins align with the holes, then press down. Use a strip of masking tape to hold several packages down while soldering from the bottom of the board. All components are inserted from the top of the board and soldered from the bottom unless specifically stated otherwise.

3. BE SURE to check DIP orientation and location before soldering. If one is soldered in the incorrect position, it is tedious to remove. If this happens, it is usually best to remove the IC by cutting off the pins and unsoldering the cut pins one at a time. Unless you have had experience unsoldering DIPs you will probably destroy the PC board as well as the DIP anyway. The DIP is inexpensively replaced compared to the whole circuit.

MOS PRECAUTIONS

All MOS devices, so indicated on the parts list as (MOS), are more or less susceptible to destruction by static electric discharge. The following precautions should be taken:

1. Wear cotton rather than synthetic clothing while handling MOS devices.
2. DO NOT allow household pets in the vicinity while working with MOS devices.
3. Insure that your body, the PC board and the MOS device are at the same potential before removing the device from the (shipping) carrier and inserting it into the PC board. This is easily accomplished by simultaneously bringing your working hand (and tool if used), the IC carrier and the PC board in contact immediately before removal and insertion.
4. Use a grounded soldering iron (3-wire) to solder MOS devices, if not available, connect the metal part of the iron to the ground bus on the PC card with aligator clips and flexible wire while soldering.


STATUS CABLE
8212
TO
J3
15 17 10 8 21 19 4 6
GRY PUR BLU GRN YEL ORG RED BRN
1 2 3 4 5 6 7 8
P2
TO
J2
(STATUS JACK)
+5
R2
4.7K
17
NMI
D0 D1 D2 D3 D4 D5 D6 D7
WAIT
RESET
INT
BUSRQ
BUSAK
A0 A5 A2 A7 A4 A1 A6 A3
A8 A13 A10 A15
A12 A9 A14 A11
Φ
IORQ
MREQ
WR
RD
M1
HALT
RFSH
Z80 CPU
10
11 29
+5
R S G
LS04
LS30
LS00
LS02 (x3)
LS112
7404
LS157
ST SL
+5 TO PIN 14 OF IC 1-5,7 & PIN 16 OF 6,8,9
GND. TO PIN 7 OF IC 1-5,7 & PIN 8 OF 6,8,9
C1-C7 0.1μF/15V
GND.
R3 330Ω
CR1
1N914 OR 1N4531
R1 4.7K
-5V
(N.C.)
WO
INP
OUT
MEMR
INTA
M1
HALT
STK
J2-STATUS JACK
COPYRIGHT © 1976
DO D1 D2 D3 D4 D5 D6 D7
RDY
WAIT
RESET
HOLD
HLDA
A0 A5 A2 A7 A4 A1 A6 A3
A8 A13 A10 A15
+5V
GND.
A12 A9 A14 A11
Φ2
SYNC
WR
INTE
DBIN
Φ1
+12
40 PIN PLUG TO 8080 SOCKET
J1
SCHEMATIC
DZ80-80 CPU
DUTRONICS P. O. BOX 9160 • STOCKTON, CALIFORNIA 95208
DWG. NO. Z05

```
                        *          DUTZ DZ80-80 V1.0 MONITOR
                        *          COPYRIGHT DECEMBER 1976
                        *          BY
                        *          DUTRONICS
                        *          P.O. BOX 9160
                        *          STOCKTON, CA 95208
0000  C3 A4 01"         STRT JMP ENTR
                        *IS CONSOLE CHARACTER WAITING
0003  DB 00             ICWC IN CSTS  STATUS PORT
0005  EE FF                  XRI MSK1 POLARITY
0007  E6 40                  ANI MSK2 BIT
0009  C9                    RET
                        *INPUT CONSOLE CHARACTER
000A  DB 01             ICC  IN CONI  INPUT PORT
000C  3C                     INC A
000D  C8                     RZ       NO RUBOUTS
000E  3D                     DCR A
000F  E6 7F                  ANI 7FH  STRIP PARITY
0011  C9                    RET
0012  00 00                  NOP,NOP  LEAVE ROOM
                        *OUTPUT FILE WITH ABORT TEST
0014  F5                OFAT PUSH AF  SAVE AF
0015  CD 26 00"         OFA1 CALL CONT  ABORT?
0018  38 34                  JRC ABRJ
001A  DB 00                  IN FSTS  STATUS
001C  EE FF                  XRI MSK3  POLARITY
001E  E6 80                  ANI MSK4  BIT
0020  20 F3                  JRNZ OFA1
0022  F1                    POP AF   UNSAVE
0023  D3 01                  OUT FLEO  OUTPUT PORT
0025  C9                    RET
                        *CONSOLE ABORT TEST
0026  CD 03 00"         CONT CALL ICWC
0029  C0                     RNZ  RET NONE WAITING
002A  18 2F                  JR GET1  TEST FOR ABORT
                        *OUTPUT CHARACTER TO CONSOLE
002C  F5                OCC  PUSH AF  SAVE AF
002D  DB 00             OCC1 IN CSTS  STATUS
002F  EE FF                  XRI MSK5  POLARITY
0031  E6 80                  ANI MSK6  BIT
0033  20 F8                  JRNZ OCC1
0035  F1                    POP AF  UNSAVE
0036  D3 01                  OUT CONO   OUTPUT PORT
0038  C9                    RET
                        *INPUT FILE WITH ABORT TIMER --2.5 SEC
0039  C5                IFAT PUSH B
003A  01 00 00               LXI B,0
003D  DB 00             IFA1 IN FSTS  STATUS PORT
003F  EE FF                  XRI MSK7  POLARITY
0041  E6 40                  ANI MSK8  BIT
0043  28 0C                  JRZ OUTI  READY
0045  DD CB 00 46            BIT 0,X  EAT TIME
0049  0D                    DCR C  MORE TIME
004A  20 F1                  JRNZ IFA1  AGAIN
```

```
004C  10 EF          DJNZ IFA1  AGAIN AGAIN
004E  C3 A0 01"  ABRJ JMP ABRT  ABORT JUMP OFF
0051  00           OUTI NOP  ROOM
0052  C1               POP B
0053  DB 01            IN FLEI  INPUT PORT
0055  C9               RET
                 *GET CONSOLE CHARACTER, TEST FOR TERM
                 *AND ABORT ON Z
0056  CD 03 00"  GETC CALL ICWC  CONSOLE READY?
0059  20 FB            JRNZ GETC  JMP NO
005B  CD 0A 00"          CALL ICC  GET IT
005E  28 F6              JRZ GETC  NO NULLS OR RUBOUTS
0060  FE 2C        GET1 CPI ','
0062  C8                RZ
0063  FE 20              CPI 20H   SPACE
0065  C8                RZ
0066  FE 5E              CPI 5EH  UP ARROW
0068  C8                RZ
0069  FE 0D              CPI 0DH  CR
006B  3F                CMC  IF CR RETURN WITH CY SET
006C  C8                RZ
006D  FE 5A              CPI 'Z'  ABORT
006F  28 DD              JRZ ABRJ
0071  37                STC
0073  3F                CMC  CLEAR CY
0073  C9                RET
                 *INPUT FILE AND COMPARE WITH C
0074  CD 39 00"  IFCC CALL IFAT  GET A CHARACTER
0077  B9              CMP C  COMPARE TO C
0078  C9                RET
                 *GET FILE BYTE--PACKS IN A
0079  CD 3E 01"  GTFB CALL GTFC  GET A HEX CHAR
007C  D5           GFB1 PUSH D
007D  07                RLC
007E  07                RLC
007F  07                RLC
0080  07                RLC
0081  5F                MOV E,A  SAVE IT
0082  CD 3E 01"         CALL GTFC  GET ANOTHER
0085  B3                ORA E  PACK IT
0086  5F                MOV E,A   SAVE IT
0087  81                ADD C  UPDATE CHECK SUM
0088  4F                MOV C,A  STORE CHECK SUM
0089  7B                MOV A,E  BYTE IN A
008A  D1                POP D
008B  C9                RET
                 *CONVERT A TO HEX IN A
008C  E6 7F      HEX  ANI 7FH  STRIP PARITY
008E  D6 30           SUI 30H
0090  FE 0A           CPI 0AH
0092  3F              CMC
0093  D0              RNC  CY = ERROR
0094  FE 11           CPI 11H
0096  D8              RC
0097  FE 17           CPI 17H
```

```
0099  3F              CMC
009A  D8              RC
009B  D6 07           SUI 7
009D  C9              RET
                    *STORE AND ABORT ON HL GT DE
009E  77            STRT MOV M,A
009F  CD A7 00"       CALL HGTD
00A2  DA A0 01"       JC ABORT
00A5  C9              RET
                    *INX H AND SET CY IF HL GT DE
00A6  2B            HDVD DCX H
00A7  23            HGTD INX H
00A8  7B              MOV A,E
00A9  95              SUB L
00AA  7A              MOV A,D
00AB  9C              SBB H
00AC  C9              RET
                    *OUTPUT TO FILE A AS TWO ASCII
00AD  F5            OFAS PUSH AF
00AE  81              ADD C
00AF  4F              MOV C,A  UPDATE CHECK SUM
00B0  F1              POP AF
00B1  C5              PUSH B
00B2  CD 06 01"       CALL ASC2
00B5  CD 14 00"       CALL OFAT
00B8  79              MOV A,C
00B9  CD 14 00"       CALL OFAT
00BC  C1              POP B
00BD  C9              RET
                    *CONSOLE MESSAGE
00BE  3F            MESG '?'
00BF  00 0D 0A 00     00,0DH,0AH,00  CR, LF
00C3  00 00 00 00     00,00,00,00  ROOM
00C7  44 55 54 5A     'DUTZ'
00CB  20 56 31 2E     ' V1.'
00CF  30              '0'
                    *OUTPUT MESSAGE TO CONSOLE
00D0  7E            PRNT MOV A,M
00D1  CD 2C 00"       CALL OCC
00D4  23              INX H
00D5  10 F9           DJNZ PRNT
00D7  C9              RET
                    *OUTPUT NULLS TO FILE
00D8  C5            NULS PUSH B
00D9  01 00 30        LXI B,3000H
00DC  CD E1 00"       CALL OFBC
00DF  C1              POP B
00E0  C9              RET
                    *OUTPUT C TO FILE, B TIMES
00E1  79            OFBC MOV A,C
00E2  CD 14 00"       CALL OFAT
00E5  10 FA           DJNZ OFBC
00E7  C9              RET
                    *GET TOP OF MEMORY IN HL
00E8  21 FF 00"     TOPM LXI H,0FFH"
```

```
00EB  25           TOP1 DCR H
00EC  7E                MOV A,M
00ED  2F                CMA
00EE  77                MOV M,A
00EF  BE                CMP M
00F0  2F                CMA
00F1  77                MOV M,A
00F2  20 F7             JRNZ TOP1
00F4  C9                RET
                   *PRINT TOP OF MEMORY
00F5  CD E8 00"    TOPP CALL TOPM
00F8  CD 1B 01"         CALL OCHL
00FB  C9                RET
                   *CONVERT LOW ORDER A TO ASCII
00FC  E6 0F        ASC  ANI 0FH
00FE  C6 30             ADI 30H
0100  FE 3A             CPI 3AH
0102  D8                RC
0103  C6 07             ADI 7H
0105  C9                RET
                   *CONVERT A TO TWO ASCII
0106  47           ASC2 MOV B,A  SAVE A
0107  CD FC 00"         CALL ASC  CONVERT LOW ORDER
010A  4F                MOV C,A  AND SAVE IN C
010B  78                MOV A,B
010C  07                RLC
010D  07                RLC
010E  07                RLC
010F  07                RLC
0110  CD FC 00"         CALL ASC  CONVERT HIGH ORDER
0113  C9                RET
                   *GET COMMAND FROM CONSOLE AND ECHO
0114  CD 56 00"    GTCC CALL GETC
0117  CD 2C 00"         CALL OCC
011A  C9                RET
                   *OUTPUT TO CONSOLE HL
011B  7C           OCHL MOV A,H
011C  CD 24 01"         CALL OCA
011F  7D                MOV A,L
0120  CD 24 01"         CALL OCA
0123  C9                RET
                   *OUTPUT TO CONSOLE A AS TWO ASCII
0124  C5           OCA  PUSH B
0125  CD 06 01"         CALL ASC2
0128  CD 2C 00"         CALL OCC
012B  79                MOV A,C
012C  CD 2C 00"         CALL OCC
012F  C1                POP B
0130  C9                RET
                   *OUTPUT CR LF TO CONSOLE
0131  E5           CRLF PUSH H
0132  C5                PUSH B
0133  06 04              MVI B,4H
0135  21 BF 00"         LXI H,MESG+1
0138  CD D0 00"         CALL PRNT
```

```
013B  C1              POP B
013C  E1              POP H
013D  C9              RET
                    *GET FILE CHARACTER, ABORT IF NOT ASCII HEX
013E  CD 39 00"     GTFC CALL IFAT
0141  CD 8C 00"         CALL HEX
0144  38 5A             JRC ABRT
0146  C9                RET
                    *OUTPUT TO CONSOLE ONE SP OR TWO SP
                    *OR ONE * OR ONE ,
0147  3E 2A        OC1A MVI A,'*'
0149  18 0B            JR OUT1
014B  3E 2C        OC1C MVI A,','
014D  18 07             JR OUT1
014F  3E 20        OC2S MVI A,' '
0151  CD 2C 00"         CALL OCC
0154  3E 20        OC1S MVI A,' '
0156  CD 2C 00"     OUT1 CALL OCC
0159  C9                RET
                    *GET FROM THRU PARAMETERS
015A  CD 80 01"     PAFT CALL PARA
015D  42                MOV B,D
015E  4B                MOV C,E  FROM IN BC
015F  CD 4B 01"         CALL OC1C
0162  CD 80 01"     PARH CALL PARA
0165  EB                XCHG  THRU IN HL
0166  C9                RET
                    *GET PARAMETER IN DE ABORT IF NO CR TERMINATOR
0167  CD 4B 01"     PARD CALL OC1C
016A  CD 80 01"     PARG CALL PARA
016D  30 31             JRNC ABRT
016F  C9                RET
                    *ROTATE ASCII IN A AS HEX THRU DE
0170  CD 8C 00"     RODE CALL HEX
0173  D8                RC
0174  EB                XCHG
0175  29                DAD H
0176  29                DAD H
0177  29                DAD H
0178  29                DAD H
0179  EB                XCHG
017A  B3                ORA E
017B  5F                MOV E,A
017C  C9                RET
                    *GET PARAMETER IN DE
017D  CD 4B 01"     PARC CALL OC1C
0180  C5            PARA PUSH B
0181  11 00 00           LXI D,0
0184  43                MOV B,E  DE AND B ZEROED
0185  CD 56 00"     PAR1 CALL GETC  GET CONSOLE CHARACTER
0188  28 11             JRZ POUT  OUT ON TERMINATOR
018A  4F                MOV C,A  SAVE IT
018B  CD 70 01"         CALL RODE  ROTATE INTO DE
018E  38 07             JRC PBAD  DING NON-HEX
0190  79                MOV A,C  UNSAVE IT
```

```
0191  04              INR B  B NON ZERO FOR ANY ENTERED
0192  CD 2C 00"   PAR2 CALL OCC  ECHO IT
0195  18 EE           JR PAR1  AGAIN
0197  3E 07       PBAD MVI A,7H  BELL
0199  18 F7           JR PAR2  OUTPUT DING AND DO AGAIN
019B  04           POUT INR B
019C  05           DCR B  SET ZERO IF NONE ENTERED
019D  C1           POP B
019E  C9           RET
019F  00           NOP  ROOM
                   *ABORT ROUTINE
01A0  06 07        ABRT MVI B,7H
01A2  18 02           JR ENT1  SKIP OVER SIGN ON MESSAGE
                   *PRINT MESSAGE AND SET STK POINTER
01A4  06 12        ENTR LXI B,12H
01A6  AF           ENT1 XRA A
01A7  ED 47           MOV I,A  SET INT MODE 0
01A9  FB              EI
01AA  31 00 00        LXI SP,0
01AD  DD E1           POP X  SAVE 0000 AND 0001
01AF  CD E8 00"       CALL TOPM  GET TOP OF MEMORY
01B2  DD E5           PUSH X
01B4  23              INX H
01B5  F9              SPHL  STK POINTER SET
01B6  21 A0 01"       LXI H,ABRT
01B9  E5              PUSH H  ABORT ON TOO MANY RETURNS
01BA  21 BE 00"       LXI H,MESG
01BD  CD D0 00"       CALL PRNT  OUTPUT ABORT OR SIGN ON MESSAGE
                   *MAIN COMMAND READ LOOP
01C0  21 C0 01"    MAIN LXI H,MAIN  HERE ON A RETURN
01C3  E5              PUSH H
01C4  CD 31 01"    MAN1 CALL CRLF
01C7  3E 5A           MVI A,'Z'
01C9  CD 2C 00"       CALL OCC  OUTPUT PROMPTER
01CC  CD 14 01"       CALL GTCC  GET COMMAND CHARACTER
                   *EXECUTE ROUTINE
01CF  FE 45        E     CPI 'E'
01D1  20 05           JRNZ MT
01D3  CD 6A 01"       CALL PARG  GET EXEC ADDRESS
01D6  EB              XCHG
01D7  E9              PCHL  GO
                   *MODIFY ROUTINE
01D8  FE 4D        MT    CPI 'M'
01DA  20 2C           JRNZ DT
01DC  CD 80 01"    M     CALL PARA  GET START ADDRESS
01DF  EB              XCHG
01E0  CD 31 01"    ME1   CALL CRLF
01E3  CD 1B 01"       CALL OCHL  OUTPUT ADDRESS
01E6  CD 54 01"       CALL OC1S
01E9  7E              MOV A,M
01EA  CD 24 01"       CALL OCA  OUTPUT ADDRESS CONTENTS
01ED  CD 4F 01"       CALL OC2S
01F0  CD 80 01"       CALL PARA  ANY CHANGE?
01F3  D8              RC  RETURN TO MONITOR ON CR
01F4  06 00           MVI B,0
```

```
01F6  2B              DCX H
01F7  28 01           JRZ ME2  JMP NONE ENTERED
01F9  04              INR B  REMEMBER TO WRITE
01FA  FE 5E     ME2   CPI 5EH  UP ARROW
01FC  28 E2           JRZ ME1  DISPLAY PREVIOUS LOCATION
01FE  23              INX H
01FF  10 04           DJNZ ME3  JMP NO WRITE
0201  73              MOV M,E  WRITE IT
0202  CD 47 01"        CALL OC1A  WRITE INDICATOR
0205  23        ME3   INX H
0206  18 D8           JR ME1  DO AGAIN
                *DUMP ROUTINE
0208  FE 44     DT    CPI 'D'
020A  20 22           JRNZ OUTT
020C  CD 5A 01"  D     CALL PAFT  GET FROM AND THRU
020F  EB              XCHG
0210  60              MOV H,B
0211  69              MOV L,C
0212  CD 31 01"  D1    CALL CRLF
0215  CD 1B 01"        CALL OCHL
0218  CD 54 01"  D2    CALL OC1S
021B  CD 26 00"        CALL CONT  HAD ENOUGH?
021E  D8              RC
021F  7E              MOV A,M
0220  CD 24 01"        CALL OCA  OUTPUT LOCATION
0223  CD A7 00"        CALL HGTD
0226  D8              RC  RETURN ON COMPLETE
0227  7D              MOV A,L
0228  E6 0F           ANI 0FH
022A  28 E6           JRZ D1  NEW LINE
022C  18 EA           JR D2  SAME LINE
                *OUTPUT ROUTINES
022E  FE 4F     OUTT  CPI 'O'
0230  C2 CF 02"       JNZ INTT
0233  CD 14 01"        CALL GTCC  GET SECOND COMMAND
0236  FE 4C           CPI 'L'
0238  28 2C           JRZ OL
023A  FE 42           CPI 'B'
023C  C0              RNZ  INPUT GARBAGE
                *OUTPUT BINARY ROUTINE
023D  CD 80 01"  OB    CALL PARA
0240  EB              XCHG
0241  CD 67 01"        CALL PARD
0244  CD A6 00"        CALL HDVD
0247  DA A0 01"        JC ABRT  FROM GT THRU
024A  CD D8 00"        CALL NULS  LEADER
024D  01 FF 08         LXI B,8FFH
0250  CD E1 00"        CALL OFBC  CUE
0253  7E        OBL1  MOV A,M
0254  CD 14 00"        CALL OFAT  OUTPUT A LOCATION
0257  CD A7 00"        CALL HGTD
025A  30 F7           JRNC OBL1  JMP NOT DONE
025C  01 FF 08         LXI B,8FFH
025F  CD E1 00"        CALL OFBC  END OF FILE
0262  CD D8 00"        CALL NULS  TRAILER
```

```
0265  C9                    RET
                      *OUTPUT LOADER FORMAT ROUTINE
0266  CD 5A 01"       OL    CALL PAFT
0269  11 FF FF              LXI D,0FFFFH  NO EOF
026C  38 03                 JRC OLE0  JMP NO EOF
026E  CD 67 01"             CALL PARD  GET EOF PARAMETER
0271  B7              OLE0  ORA A  ZERO CY
0272  ED 42                 SBC B
0274  DA A0 01"             JC ABRT  FROM GT THRU
0277  23                    INX H
0278  C5                    PUSH B
0279  EB                    XCHG
027A  E3                    XTHL
027B  CD D8 00"             CALL NULS  LEADER
027E  7B              OLE1  MOV A,E
027F  B2                    ORA D
0280  28 42                 JRZ OLE4  JMP IF DE ZERO
0282  01 00 18              LXI B,1800H  18 PER RECORD, ZERO CHECKSUM
0285  7B                    MOV A,E
0286  98                    SBB B
0287  7A                    MOV A,D
0288  99                    SBB C
0289  30 01                 JRNC OLE2  DE GT 18H
028B  43                    MOV B,C  DE LT 18H
028C  3E 0D           OLE2  MVI A,0DH
028E  CD 14 00"             CALL OFAT  OUTPUT CR
0291  3E 0A                 MVI A,0AH
0293  CD 14 00"             CALL OFAT  OUTPUT LF
0296  3E 00                 MVI A,00
0298  CD 14 00"             CALL OFAT  OUTPUT NULL
029B  3E 3A                 MVI A,':'
029D  CD 14 00"             CALL OFAT  CUE
02A0  78                    MOV A,B
02A1  CD AD 00"             CALL OFAS  LENGTH
02A4  7C                    MOV A,H
02A5  CD AD 00"             CALL OFAS  HIGH ADDRESS
02A8  7D                    MOV A,L
02A9  CD AD 00"             CALL OFAS  LOW ADDRESS
02AC  04                    INR B
02AD  05                    DCR B
02AE  28 1B                 JRZ OLE5  JMP END OF FILE
02B0  AF                    XRA A
02B1  CD AD 00"             CALL OFAS  TYPE 00 RECORD
02B4  7E              OLE3  MOV A,M  GET IT
02B5  CD AD 00"             CALL OFAS  OUTPUT IT
02B8  23                    INX H  NEXT
02B9  1B                    DCX D
02BA  10 F8                 DJNZ OLE3  JMP NOT DONE
02BC  79                    MOV A,C
02BD  ED 44                 NEG
02BF  CD AD 00"             CALL OFAS  CHECKSUM
02C2  18 BA                 JR OLE1  AGAIN
02C4  E1              OLE4  POP H
02C5  7D                    MOV A,L
02C6  A4                    ANA H
```

```
02C7  3C              INR A
02C8  C8              RZ  NO EOF
02C9  18 C1           JR OLE2  GO OUTPUT EOF
02CB  CD D8 00"  OLE5 CALL NULS  TRAILER
02CE  C9              RET
                 *INPUT ROUTINES
02CF  FE 49      INTT CPI 'I'
02D1  C2 7C 03"       JNZ VERT
02D4  CD 14 01"       CALL GTCC  GET SECOND COMMAND
02D7  FE 4C           CPI 'L'
02D9  28 4B           JRZ IL
02DB  FE 42           CPI 'B'
02DD  C0              RNZ  GARBAGE
                 *INPUT BINARY ROUTINE
02DE  CD 62 01"  IB   CALL PARH  GET LOAD ADDRESS
02E1  38 07           JRC IBE0  JMP NO STOP ADDRESS
02E3  CD 67 01"       CALL PARD  GET STOP ADDRESS
02E6  7B              MOV A,E
02E7  B2              ORA D
02E8  20 03           JRNZ IBE1
02EA  11 FF FF   IBE0 LXI D,0FFFFH  NO STOP
02ED  01 FF 04   IBE1 LXI B,04FFH    CUE
02F0  CD 74 00"  IBE2 CALL IFCC
02F3  20 F8           JRNZ IBE1
02F5  10 F9           DJNZ IBE2  FIND 4 CUE WORDS
02F7  CD 74 00"  IBE3 CALL IFCC
02FA  28 FB           JRZ IBE3  LOOP TILL NO CUE WORDS
02FC  47              MOV B,A
02FD  3E 07           MVI A,7H  BELL
02FF  CD 2C 00"       CALL OCC  DING
0302  78              MOV A,B
0303  CD 9E 00"  IBE5 CALL STRT  STORE IT
0306  CD 74 00"       CALL IFCC  GET ONE
0309  20 F8           JRNZ IBE5  LOOP NOT FFH
030B  06 01           MVI B,1  ONE FFH FOUND
030D  CD 74 00"  IBE6 CALL IFCC  GET ANOTHER
0310  28 0D           JRZ IBE8  MORE FFH
0312  4F              MOV C,A  SAVE NON FFH
0313  3E FF       IBE7 MVI A,0FFH
0315  CD 9E 00"       CALL STRT
0318  10 F9           DJNZ IBE7  STORE THE FFH WORDS
031A  79              MOV A,C
031B  0E FF           MVI C,0FFH
031D  18 E4           JR IBE5  GO STORE NON FFH
031F  04         IBE8 INR B  COUNT FFH WORDS
0320  3E 08           MVI A,8H
0322  B8              CMP B  8 YET?
0323  20 E8           JRNZ IBE6  JMP NO
0325  C9              RET  YES
                 *INPUT LOADER FORMAT
0326  CD 62 01"  IL   CALL PARH  GET START ADDRESS
0329  11 00 00        LXI D,0
032C  38 03           JRC ILE1
032E  C3 A0 01"       JMP ABRT  CHG TO CALL PARD FOR REL
0331  19         ILE1 DAD D
```

```
0332  E5              PUSH H
0333  FD E1           POP Y
0335  CD 74 00"  ILE2 CALL IFCC  LOOK FOR CUE
0338  E6 7E       ILE3 ANI 7EH
033A  D6 3A            SUI ':'
033C  20 F7            JRNZ ILE2
033E  4F               MOV C,A  ZERO CHECKSUM
033F  CD 79 00"        CALL GTFB
0342  47               MOV B,A  LENGTH
0343  CD 79 00"        CALL GTFB
0346  67               MOV H,A  HIGH ADDRESS
0347  CD 79 00"        CALL GTFB
034A  6F               MOV L,A  LOW ADDRESS
034B  04               INR B
034C  05               DCR B
034D  C8               RZ  EOF
034E  FD E5            PUSH Y
0350  DD E1            POP X
0352  EB               XCHG
0353  DD 19            DADX D
0355  EB               XCHG  X=STORE ADDRESS
0356  CD 79 00"        CALL GTFB  GET RECORD TYPE
0359  3D               DCR A
035A  CA A0 01"        JZ ABRT  REL LD JMP OFF
035D  CD 79 00"  LDAB  CALL GTFB  GET DATA BYTE
0360  DD 77 00         MOV MX,A  STORE IT
0363  DD 23            INX X  SET FOR NEXT
0665  05               DCR B  UPDATE COUNT
0366  20 F5            JRNZ LDAB  AGAIN IF MORE
0368  CD 74 00"  CKSM  CALL IFCC  GET CHECKSUM
036B  47               MOV B,A  SAVE IT
036C  CD 8C 00"        CALL HEX  TEST FOR CHECKSUM
036F  30 03            JRNZ CKS1  JMP FOR CHECKSUM
0371  78               MOV A,B
0372  18 C4            JR ILE3  NO CHECKSWM
0374  CD 7C 00"  CKS1  CALL GFB1  GET REST OF SUM
0377  28 BC            JRZ ILE2  JMP OK
0379  C3 A0 01"        JMP ABRT  CHECKSUM ERROR
                  *VERIFY ROUTINE
037C  FE 56       VERT CPI 'V'
037E  20 1E            JRNZ SERT
0380  CD 62 01"  V     CALL PARH  GET FROM
0383  CD 67 01"        CALL PARD  GET TO
0386  CD 26 00"  V1    CALL CONT  ENOUGH?
0389  7E               MOV A,M
038A  47               MOV B,A
038B  2F               CMA
038C  77               MOV M,A
038D  AE               XRA M
038E  70               MOV M,B
038F  28 07            JRZ V2  JMP OK
0391  D5               PUSH D
0392  C5               PUSH B
0393  CD E0 01"        CALL ME1  M ROUTINE ENTRY
0396  C1               POP B
```

```
0397  D1              POP D
0398  CD A7 00"   V2  CALL HGTD
039B  30 E9           JRNC V1  JMP NOT DONE
039D  C9              RET
                    *SEARCH ROUTINE
039E  FE 53         SERT CPI 'S'
03A0  20 16           JRNZ HEXT
03A2  CD 62 01"     S   CALL PARH  GET FROM
03A5  CD 7D 01"         CALL PARC  GET TO
03A8  01 00 00      SE1 LXI B,0  B USED BY M ROUTINE
03AB  7B              MOV A,E  SEARCH IN E
03AC  ED B1           CPIR  LOOK FOR IT
03AE  E0              RPO  RET NONE FOUND
03AF  2B              DCX H  STOPS ONE HIGHER
03B0  D5              PUSH D  SAVE D
03B1  CD E0 01"       CALL ME1  M ROUTINE ENTRY
03B4  D1              POP D  UNSAVE D
03B5  23              INX H  SET HL FOR NEXT
03B6  18 F0           JR SE1  AGAIN
                    *HEXIDECIMAL ROUTINE
03B8  FE 48         HEXT CPI 'H'
03BA  20 18           JRNZ RELT
03BC  CD 5A 01"     H   CALL PAFT  GET TWO NUMBERS
03BF  E5              PUSH H  SAVE FIRST
03C0  09              DAD B  ADD THEM
03C1  CD 4F 01"       CALL OC2S
03C4  CD 1B 01"       CALL OCHL  OUTPUT SUM
03C7  C5              PUSH B
03C8  E1              POP H
03C9  C1              POP B
03CA  B7              ORA A  ZERO CY
03CB  ED 42           SBC B  SUBTRACT THEM
03CD  CD 4B 01"       CALL OC1C
03D0  CD 1B 01"       CALL OCHL  OUTPUT DIFFERENCE
03D3  C9              RET
                    *RELOCATE ROUTINE
03D4  FE 52         RELT CPI 'R'
03D6  20 22           JRNZ TOPT
03D8  CD 5A 01"     R   CALL PAFT  GET FROM THRU
03DB  DA A0 01"     RE1 JC ABORT  NO TO
03DE  CD 67 01"         CALL PARD  GET TO
03E1  B7              ORA A  ZERO CY
03E2  ED 42           SBC B  NO. LOCATIONS
03E4  38 F5           JRC RE1  ABORT ON FROM GT TO
03E6  C5              PUSH B
03E7  E3              XTHL
03E8  C1              POP B
03E9  CD A6 00"       CALL HDVD
03EC  38 08           JRC RE2  MOVE FORWARD
03EE  09              DAD B
03EF  EB              XCHG
03F0  09              DAD B
03F1  EB              XCHG
03F2  03              INX B
03F3  ED B8           LDDR  MOVE REVERSE
```

```
03F5  C9              RET
03F6  03          RE2 INX B  FORWARD
03F7  ED B0           LDIR
03F9  C9              RET
                  *TOP MEMORY JUMP OFF
03FA  FE 54       TOPT CPI 'T'
03FC  CA F5 00"       JZ TOPP
03FF  C9              RET  ILLEGAL COMMAND
                  *REPLACE ABOVE RET WITH NEXT ROUTINE
                  *WHEN EXPANDING
```