— CONTENTS —

LINKING ASSEMBLY PROGRAM WITH FDOS

An object program generated with the FDOS editor, assembler and linker can be executed with the RUN command.

Example 1: 1 > RUN GALAXY [CR]

This command loads GALAXY.OBJ into memory from the floppy disk and executes it. Execution of a RET statement in the object program returns control to FDOS. The contents of the stack pointer must be restored to the value contained when the object program was called before the RET statement is executed. CF must be reset before control is returned because an error message will be output on the assumption that the ACC contains a systen, error code if CF is set.

Global switches and/or arguments can be assigned after the file name in the RUN command as shown below.

Example 2: 2 > RUN ASMZ8/P CONTROL–A [CR]

Global switch Argument

In this case, FDOS converts the entire command line into intermediate codes (refer to the LIBRARY/PACKAGE Manual), and loads ASMZ8.OBJ into memory from the floppy disk, then executes it. At this time, the HL register points to the intermediate code corresponding to /P (88H). The RJOB area in FDOS has the same value as the HL register.

Switches and arguments following the file name (ASMZ8) must be decoded by user object program. They can be decoded using FDOS subroutines. When the last character (" : " or 0DH) is decoded, the HL register contents must be stored in RJOB. To return control to FDOS, execute a RET statement in the object program.

The sample program listed on the following pages illustrates command line decoding. It outputs an ASCII file to the CRT display or printer. This program operates in a manner similar to the FDOS TYPE command. The file name of this program is TYPE'. Thus, executing

Example 3: 2 > RUN TYPE' ABC [CR]

outputs file ABC.ASC to the CRT display and executing

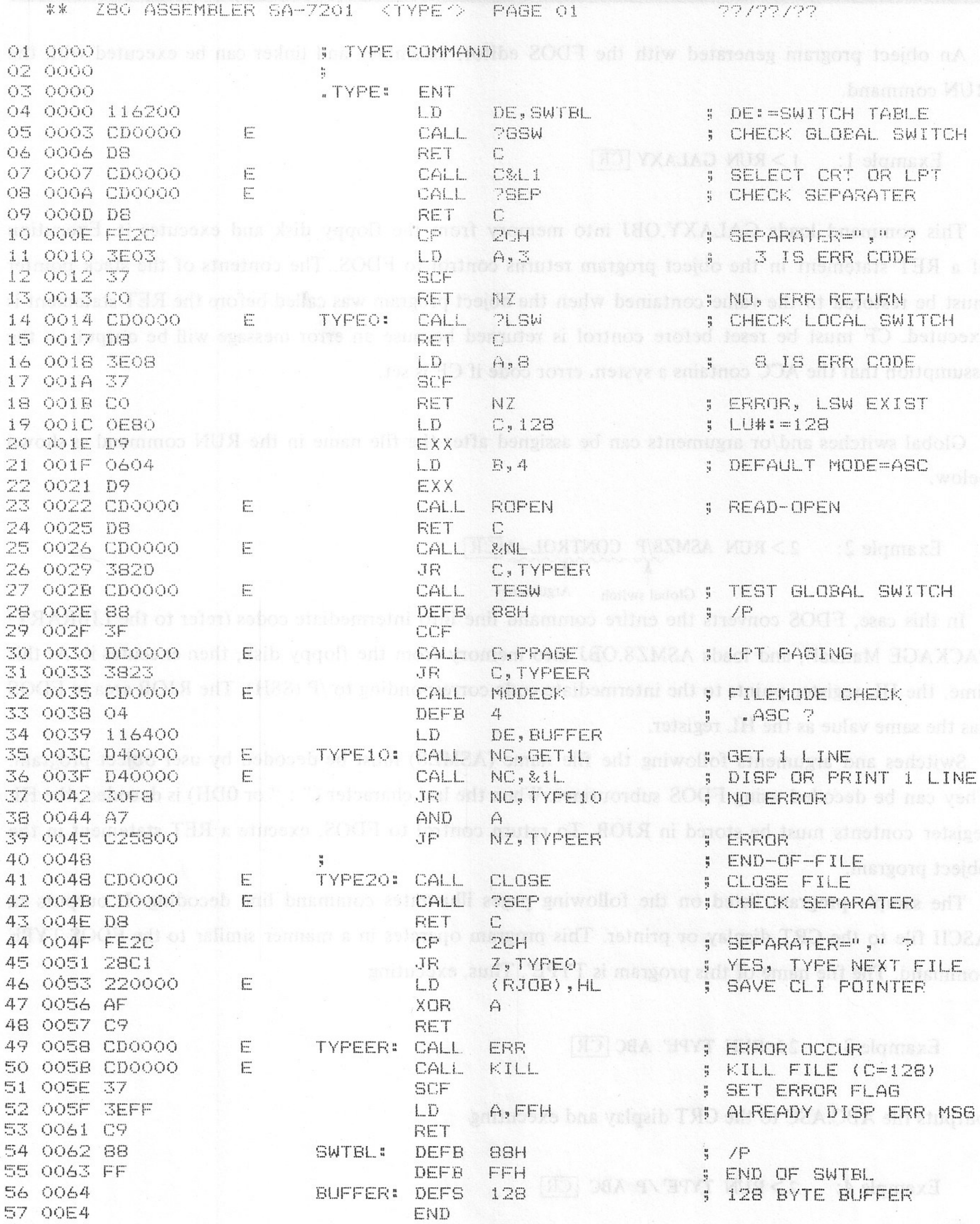
Example 4: 2 > RUN TYPE'/P ABC [CR]

outputs ABC.ASC to the printer.

All external labels (indicated by the E message) in this program list are defined in FDOSEQU. LIB.

See page SYS- 46 in System Command for the RUN command.

—Sample Program (Command)—



```
**  Z80 ASSEMBLER SA-7201  <TYPE>  PAGE 01                 ??/??/??

0000              ; TYPE COMMAND
0000              ;
0000              .TYPE:    ENT
0000 116200                 LD    DE,SWTBL         ; DE:=SWITCH TABLE
0003 CD0000   E             CALL  ?GSW             ; CHECK GLOBAL SWITCH
0006 D8                     RET   C
0007 CD0000   E             CALL  C&L1             ; SELECT CRT OR LPT
000A CD0000   E             CALL  ?SEP             ; CHECK SEPARATER
000D D8                     RET   C
000E FE2C                   CP    2CH              ; SEPARATER="," ?
0010 3E03                   LD    A,3              ;   3 IS ERR CODE
0012 37                     SCF
0013 C0                     RET   NZ               ; NO, ERR RETURN
0014 CD0000   E   TYPE0:    CALL  ?LSW             ; CHECK LOCAL SWITCH
0017 D8                     RET   C
0018 3E08                   LD    A,8              ;   8 IS ERR CODE
001A 37                     SCF
001B C0                     RET   NZ               ; ERROR, LSW EXIST
001C 0E80                   LD    C,128            ; LU#:=128
001E D9                     EXX
001F 0604                   LD    B,4              ; DEFAULT MODE=ASC
0021 D9                     EXX
0022 CD0000   E             CALL  ROPEN            ; READ-OPEN
0025 D8                     RET   C
0026 CD0000   E             CALL  &NL
0029 382D                   JR    C,TYPEER
002B CD0000   E             CALL  TESW             ; TEST GLOBAL SWITCH
002E 88                     DEFB  88H              ; /P
002F 3F                     CCF
0030 DC0000   E             CALL  C,PPAGE          ; LPT PAGING
0033 3823                   JR    C,TYPEER
0035 CD0000   E             CALL  MODECK           ; FILEMODE CHECK
0038 04                     DEFB  4                ;   .ASC ?
0039 116400                 LD    DE,BUFFER
003C D40000   E   TYPE10:   CALL  NC,GET1L         ; GET 1 LINE
003F D40000   E             CALL  NC,&1L           ; DISP OR PRINT 1 LINE
0042 30F8                   JR    NC,TYPE10        ; NO ERROR
0044 A7                     AND   A
0045 C25800                 JP    NZ,TYPEER        ; ERROR
0048              ;                                ; END-OF-FILE
0048 CD0000   E   TYPE20:   CALL  CLOSE            ; CLOSE FILE
004B CD0000   E             CALL  ?SEP             ; CHECK SEPARATER
004E D8                     RET   C
004F FE2C                   CP    2CH              ; SEPARATER="," ?
0051 28C1                   JR    Z,TYPE0          ; YES, TYPE NEXT FILE
0053 220000   E             LD    (RJOB),HL        ; SAVE CLI POINTER
0056 AF                     XOR   A
0057 C9                     RET
0058 CD0000   E   TYPEER:   CALL  ERR              ; ERROR OCCUR
005B CD0000   E             CALL  KILL             ; KILL FILE (C=128)
005E 37                     SCF                    ; SET ERROR FLAG
005F 3EFF                   LD    A,FFH            ; ALREADY DISP ERR MSG
0061 C9                     RET
0062 88           SWTBL:    DEFB  88H              ; /P
0063 FF                     DEFB  FFH              ; END OF SWTBL
0064              BUFFER:   DEFS  128              ; 128 BYTE BUFFER
00E4                        END
```

```
 **   Z80 ASSEMBLER SA-7201  <TYPE >  PAGE 02           ??/??/??

.TYPE   0000  BUFFER 0064  SWTBL  0062  TYPE0   0014  TYPE10 003C
TYPE20 0048  TYPEER 0058
```

USER CODED I/O ROUTINES

FDOS supports control programs not only for the floppy disk drive but for the printer ($LPT) and the paper tape reader ($PTR), etc. Other I/O devices can be operated under the control of FDOS by means of user coded control programs.

—User I/O Routine—

A user I/O routine consists of the following sections.

A. Device table (57 bytes)

B. ROPEN or WOPEN procedure

C. Data transfer program

D. CLOSE and KILL procedure

These sections are explained below using the FDOS paper tape reader control program ($PTR) as an example.

A. **Device Table** (line 9 through 27, bytes 0 through 56)

∗ FDOS uses bytes 0 through 2 (FLAG 0 ~ FLAG 2).
This area must be written exactly as it is shown.

∗ Byte 3 (FLAG 3) represents the attribute of the I/O device.

Bit 7 : 0

Bit 6 : 1 indicates that tabulation is possible. (This bit is set to 1 for the printer. See Note 1 on page 7.)

Bit 5 : 1 indicates that parity specification ($PTR/PE, etc.) can be made.

Bit 4 : 1 indicates that only .ASC-mode files can be transferred.

Bit 3 : 0

Bit 2 : 0

Bit 1 : 1 indicates that WOPEN is possible. (See note 2 on page 7.)

Bit 0 : 1 indicates that ROPEN is possible. (See note 2 on page 7.)

∗ Byte 4 indicates the data transfer format. (described later)

∗ Bytes 5 and 6 are the starting address of the subroutine to be called during ROPEN execution.

∗ Bytes 7 and 8 are the starting address of the subroutine to be called during WOPEN execution. (WOPEN is not executed for $PTR so DEFW 0 is specified in this program.)

∗ Bytes 9 and 10 are loaded with data by the FDOS STATUS command. (Not used for $PTR.)

∗ Bytes 11 through 14 (0BH ~ 0EH) are the starting addresses of the subroutines for CLOSE and KILL processing.

∗ Bytes 15 through 22 (0FH ~ 16H Procedures 1 ~ 4) are loaded with data transfer routine addresses. The data transfer procedure differes according to the transfer format.

ROPEN

Transfer format	1	4
Procedure 1	Input 1 character (ACC ← data)	Input 1 line (From the address indicated by DE to a CR code.)
Procedure 2 ~ 4	Unused	Unused

WOPEN

Transfer format	1	5
Procedure 1	Unused	Carriage return
Procedure 2	Output 1 character (ACC : data)†	Output 1 character (ACC : data)†
Procedure 3	Unused	Output 1 line (Corresponds to monitor subroutine MSG)
Procedure 4	Unused	Output line (Corresponds to monitor subroutine MSGX)

† On .ASC mode, 0DH means carriage-return, and 0CH means form-feed.

✻ Byte 23 and 24 (17H and 18H) are used by FDOS.

✻ Byte 25 (19H) is used only when bit 6 of FLAG 3 is 1, in which case it must be loaded with the number of characters of the line which have been output by I/O routine.

✻ Byte 26 (1AH) is loaded with the file mode by FDOS.

✻ Bytes 27 through 56 (1BH ~ 38H) are the device name (up to 16 characters); the rest area must be reserved with DEFS.

✻ When the transfer format is 4, a buffer area for 1 line is reserved after the byte 56 with DEFS.

B. **ROPEN or WOPEN procedure** (lines 50 through 61)

Only ROPEN is neede for the paper tape reader ($PTR). The tape feeder is skipped by this procedure. WOPEN is also used to start a new page during output of an assembly listing.

C. **Data transfer program** (lines 62 through 95)

Program which performs actual transfer of data.

D. **CLOSE and KILL procedures** (lines 60 through 61)

No function with $PTR.

To return control to FDOS from the ROPEN, WOPEN, Procedure 1 ~ 4, CLOSE or KILL routines, set registers as follows before executing the RET statement.

Normal : CR ← 0

Error : CF ← 1, ACC ← error code (refer to the System Error Messages in the System Command Manual.)

File end : CF ← 1, ACC ← 0

The contents of the IY, BC′, DE′ and HL′ registers must be saved in any case.

—Relocating User I/O Routines—

First, assemble the program coded (the program name DVM is used below).

```
Example 1:      2>ASM  DVM,  $LPT/L [CR]
```

Next, relocate the file to generate the object program. A higher loading address must be specified at this time because of factors related to the LIMIT command described later. Take care to ensure that addresses do not overlap when two or more user I/O programs are used. If necessary, link MONEQU.LIB or FDOSEQU.LIB with the user I/O programs.

```
Example 2:      2>LINK  $C000, DVM [CR]
Example 2':     2>LINK  $C400, CDISP,  $FD1 ; FDOSEQU.LIB [CR]
```

—Linking User I/O Routines with FDOS—

User I/O routines must be linked with FDOS I/O controller every time FDOS is activated.

First, use the LIMIT command to reserve an area in memory for loading the object program (DVM.OBJ).

```
Example 3:      2>LIMIT  $C000 [CR]
```

Next, load the object program.

```
Example 4:      2>LOAD  DVM [CR]
```

Finally, link the routine to the FDOS I/O controller. $USR1 through 4 are provided in FDOS as devide names for user I/O routines; assign the user I/O control routine to one of these device names.

```
Example 5:      2>ASSIGN  $USR1,  $C000 [CR]
```

Now the user program is linked with FDOS and can be called by specifying $USR1 (~4). It is convenient to prepare EXEC files which include LIMIT, LOAD and ASSIGN commands such as those shown above. (Refer to the System Command Manual).

User I/O programs are called as shown below.

Example of use by FDOS commands

```
    2>TYPE  $USR1 [CR]
    2>XFER  DATA4,  $USR2 [CR]
```

Example of use by BASIC compiler

```
10  ROPEN   #2, "$USR1"
20  INPUT   #2, A$
30  IF EOF (#2) THEN 100
40  PRINT   A$
:
:
:
999 CLOSE  #2
```

Note 1: Bit 6 determines the functions of BASIC statements PRINT # and INPUT #.

When bit 6 = 1, data is treated in the same manner as with the PRINT and INPUT statements. When bit 6 = 0, separators (" , " and " ; ") in the PRINT # statement are replaced with $\boxed{\text{CR}}$ and commas included in the input character string for the INPUT # statement are treated as data; only $\boxed{\text{CR}}$ is regarded as a data separator. (This is the same as with the PRINT # statement supported by SA-6510 and the PRINT/T statement supported by SA-5510.)

Note 2: Both ROPEN and WOPEN are possible, when both bit 1 and bit 0 are set, but they cannot be executed simultaneously.

—Sample Program (I/O Driver)—

```
   **   Z80 ASSEMBLER SA-7201  <PTRP>  PAGE 01             12.25.81
01 0000                       ; ------------------------------
02 0000                       ; PTR/PTP DRIVER FOR MZ-80A FDOS.
03 0000                       ; COPYRIGHT 1981 BY SHARP CORP.
04 0000                       ;
05 0000                       ; ------------------------------
06 0000   P                   ;
07 0000                       IOWAIT: EQU    1311H           ;I/O ERR,WAIT SP KEY
08 0000                       ;
09 0000 0000                  $PTR:   ENT
10 0002 00                            DEFW   0               ;FLAG0,1
11 0003 21                            DEFB   0               ;FLAG2
12 0004 01                            DEFB   21H             ;FLAG3
13 0005 7900                          DEFB   1               ;TRANSFER FORMAT
14 0007 0000                          DEFW   $PTRO           ;ROPEN
15 0009 0000                          DEFW   0               ;WOPEN
16 000B 8600                          DEFW   0               ;STATUS
17 000D 8600                          DEFW   CLC             ;CLOSE
18 000F 8800                          DEFW   CLC             ;KILL
19 0011 0000                          DEFW   $PTR1           ;PROC1
20 0013 0000                          DEFW   0               ;PROC2
21 0015 0000                          DEFW   0               ;PROC3
22 0017                               DEFW   0               ;PROC4
23 0019                               DEFS   2
24 001A                               DEFS   1
                                      DEFS   1               ;FILEMODE
25 001B 24505452              $PTRNM: DEFM   '$PTR'          ;FILENAME
26 001F 0D                            DEFB   0DH
27 0020                               DEFS   25
28 0039                       ;
29 0039                       $PTP:   ENT
30 0039 0000                          DEFW   0H              ;FLAG0,1
31 003B 00                            DEFB   0H              ;FLAG2
32 003C 22                            DEFB   22H             ;FLAG3
33 003D 01                            DEFB   1               ;TRANSFER FORMAT
34 003E 0000                          DEFW   0               ;ROPEN
35 0040 2901                          DEFW   $PTPFD          ;WOPEN
36 0042 0000                          DEFW   0H              ;STATUS
37 0044 2901                          DEFW   $PTPFD          ;CLOSE
38 0046 8600                          DEFW   CLC             ;KILL
39 0048 0000                          DEFW   0               ;PROC1
40 004A BE00                          DEFW   $PTP1C          ;PROC2
41 004C 0000                          DEFW   0               ;PROC3
42 004E 0000                          DEFW   0               ;PROC4
43 0050                               DEFS   2
44 0052                               DEFS   1
45 0053                               DEFS   1               ;FILEMODE
46 0054 24505450              $PTPNM: DEFM   '$PTP'          ;FILENAME
47 0058 0D                            DEFB   0DH
48 0059                               DEFS   25
49 0072                       ;
50 0072 111B00                $PTRNR: LD     DE,$PTRNM
51 0075 CD1113                        CALL   IOWAIT
52 0078 D8                            RET    C
53 0079 CD9500                $PTRO:  CALL   $PTRIN          ;ROPEN
54 007C 38F4                          JR     C,$PTRNR
55 007E 78                            LD     A,B
56 007F A7                            AND    A
57 0080 28F7                          JR     Z,$PTRO
58 0082 78                            LD     A,B
59 0083 32BD00                        LD     ($PTRD),A
60 0086 AF                    CLC:    XOR    A
```

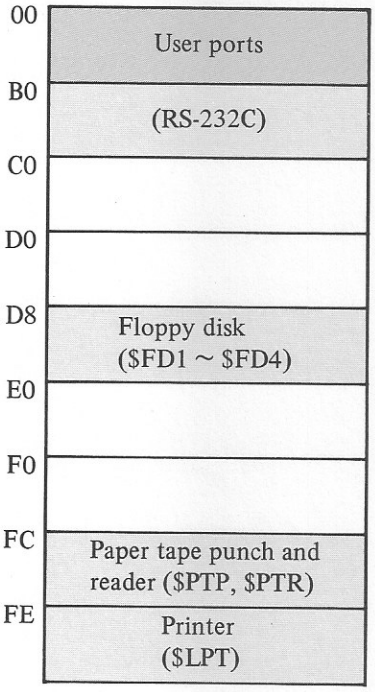
```
0087 C9                      RET
0088 CD9500       $PTR1:     CALL    $PTRIN          ;GET1C
008B D8                      RET     C
008C 21BD00                  LD      HL,$PTRD
008F 7E                      LD      A,M
0090 70                      LD      M,B
0091 A7                      AND     A
0092 C0                      RET     NZ
0093 37                      SCF                     ;EOF
0094 C9                      RET
0095              ;
0095 3EEF         $PTRIN:    LD      A,EFH
0097 D3FD                    OUT     (FDH),A
0099 CDB000       $PTR2:     CALL    $PTRCK
009C CB67                    BIT     4,A
009E 28F9                    JR      Z,$PTR2
00A0 CDB000       $PTR3:     CALL    $PTRCK
00A3 CB67                    BIT     4,A
00A5 20F9                    JR      NZ,$PTR3
00A7 DBFC                    IN      A,(FCH)
00A9 2F                      CPL
00AA 47                      LD      B,A
00AB 3EFF         $PTR5:     LD      A,FFH
00AD D3FD                    OUT     (FDH),A
00AF C9                      RET
00B0 DBFD         $PTRCK:    IN      A,(FDH)
00B2 CB6F                    BIT     5,A
00B4 C8                      RET     Z
00B5 F1                      POP     AF
00B6 CDAB00                  CALL    $PTR5
00B9 37                      SCF
00BA 3E3C                    LD      A,60            ;NOT READY
00BC C9                      RET
00BD              $PTRD:     DEFS    1
00BE              ;
00BE              ;
00BE F5           $PTP1C:    PUSH    AF
00BF DBFD                    IN      A,(FDH)
00C1 E601                    AND     1
00C3 2818                    JR      Z,$PTP20
00C5 3EFE                    LD      A,FEH
00C7 D3FD                    OUT     (FDH),A
00C9 210000                  LD      HL,0H
00CC 2B           $PTP10:    DEC     HL
00CD DBFD                    IN      A,(FDH)
00CF E601                    AND     1
00D1 280A                    JR      Z,$PTP20
00D3 7C                      LD      A,H
00D4 B5                      OR      L
00D5 00                      NOP
00D6 00                      NOP
00D7 3E3C                    LD      A,60            ;NOT READY
00D9 2837                    JR      Z,$PTP60
00DB 18EF                    JR      $PTP10
00DD              ;
00DD 3EFE         $PTP20:    LD      A,FEH
00DF D3FD                    OUT     (FDH),A
00E1 DBFD         $PTP30:    IN      A,(FDH)
00E3 CB47                    BIT     0,A
00E5 20FA                    JR      NZ,$PTP30
```

```
** Z80 ASSEMBLER SA-7201 <PTRP> PAGE 03                         12.25.81

00E7 F1                         POP    AF
00E8 F5                         PUSH   AF
00E9 2F                         CPL
00EA D3FC                       OUT    (FCH),A
00EC 3EFC                       LD     A,FCH
00EE D3FD                       OUT    (FDH),A
00F0 210000                     LD     HL,0H
00F3 2B               $PTP40:   DEC    HL
00F4 7C                         LD     A,H
00F5 B5                         OR     L
00F6 3E4E                       LD     A,78              ;TIME OUT
00F8 2818                       JR     Z,$PTP60
00FA DBFD                       IN     A,(FDH)
00FC CB4F                       BIT    1,A
00FE 20F3                       JR     NZ,$PTP40
0100 DBFD                       IN     A,(FDH)
0102 CB4F                       BIT    1,A
0104 20ED                       JR     NZ,$PTP40
0106 DBFD             $PTP50:   IN     A,(FDH)
0108 CB4F                       BIT    1,A
010A 28FA                       JR     Z,$PTP50
010C DBFD                       IN     A,(FDH)
010E CB4F                       BIT    1,A
0110 28F4                       JR     Z,$PTP50
0112 E1               $PTP60:   POP    HL
0113 F5                         PUSH   AF
0114 3EFE                       LD     A,FEH
0116 D3FD                       OUT    (FDH),A
0118 CD3601                     CALL   DLY80U
011B 3D                         DEC    A                 ;A<--FFH
011C D3FC                       OUT    (FCH),A
011E F1                         POP    AF
011F C0                         RET    NZ
0120 37                         SCF
0121 C9                         RET
0122                  ;
0122 115400           $PTPNR:   LD     DE,$PTPNM
0125 CD1113                     CALL   IOWAIT
0128 D8                         RET    C
0129 0696             $PTPFD:   LD     B,150             ;FEEDER
012B C5                         PUSH   BC
012C AF                         XOR    A
012D CDBE00                     CALL   $PTP1C
0130 C1                         POP    BC
0131 38EF                       JR     C,$PTPNR
0133 10F6                       DJNZ   $PTPFD+2
0135 C9                         RET
0136                  ;
0136 111000           DLY80U:   LD     DE,16
0139 1B                         DEC    DE
013A 7A                         LD     A,D
013B B3                         OR     E
013C 20FB                       JR     NZ,-3
013E C9                         RET
013F                            END
```

```
  **  Z80 ASSEMBLER SA-7201  <PTRP>  PAGE 04             12.25.81

$PTP    0039  $PTP10 00CC  $PTP1C 00BE  $PTP20 00DD  $PTP30 00E1
$PTP40  00F3  $PTP50 0106  $PTP60 0112  $PTPFD 0129  $PTPNM 0054
$PTPNR  0122  $PTR   0000  $PTR1  0088  $PTR2  0099  $PTR3  00A0
$PTR5   00AB  $PTRCK 00B0  $PTRD  00BD  $PTRIN 0095  $PTRNM 001B
$PTRNR  0072  $PTRO  0079  CLC    0086  DLY80U 0136  IOWAIT 1311
```

I/O MAP

I/O ports with addresses equal to or higher than B0 are reserved by the manufacturer for control of external devices; those used by FDOS are assigned device names such as $LPT.



00	User ports
B0	(RS-232C)
C0	
D0	
D8	Floppy disk ($FD1 ~ $FD4)
E0	
F0	
FC	Paper tape punch and reader ($PTP, $PTR)
FE	Printer ($LPT)

PAPER TAPE PUNCH AND READER INTERFACE

FDOS has built-in paper tape punch and reader control programs. These are assigned the device names $PTP and $PTR, respectively. In actuality, however an interface circuit must be established with a universal interface I/O card to connect the paper tape punch and reader with the MZ-80 series microcomputer. The circuit diagram is shown on page 40.

The method for controlling the paper tape punch and reader is not standardized. A paper tape punch and reader which can be controlled by FDOS must have the following signal timing system. The signal names and timing charts shown below are based on the RP-600 paper tape punch and reader manufactured by Nada Electronics Laboratory. (For details, refer to the manual included with the paper tape punch and reader.)

—Signal Name—

< Puncher >

$\overline{\text{DT}}_1 \sim \overline{\text{DT}}_8$	: Data (PTP ⟵ CPU)
MI*	: Motor ON/OFF control signal (PTP ⟵ CPU)
$\overline{\text{ST}}$	: START/STOP control signal (PTP⟵ CPU)
$\overline{\text{TO}}$	: Timing signal (PTP ⟶ CPU)
($\overline{\text{RDY}}$)**	: Ready state signal (PTP ⟶ CPU) (This signal is not output from the RP-600 since it can be used in remote operation. Ground it when the RP-600 is used.)

< Reader >

$\overline{\text{RD}}_1 \sim \overline{\text{RD}}_8$	: Data (PTR ⟶ CPU)
$\overline{\text{STA}}$	: START/STOP control signal (PTR ⟵ CPU)
$\overline{\text{SPR}}$	: Sprocket signal (PTR ⟶ CPU)
$\overline{\text{RB}}$	: Tape end signal (abnormal stop signal) (PTR ⟶ CPU)

* Do not connect when the motor is not remotely controlled.

** The DPT26A manufactured by the Anritsu Electric Co. outputs this signal, but the RP-600 does not.

—I/O Ports—

Port FC$_H$ is used for data by both the punch and the reader. Port FD$_H$ is used for control signals. See Table 1.

< Punch >

[Data]

O_{10}							O_{17}
$\overline{\text{DT}}_1$	$\overline{\text{DT}}_2$	$\overline{\text{DT}}_3$	$\overline{\text{DT}}_4$	$\overline{\text{DT}}_5$	$\overline{\text{DT}}_6$	$\overline{\text{DT}}_7$	$\overline{\text{DT}}_8$

[Control signals]

O_{20}							O_{27}
MI	$\overline{\text{ST}}$						

I_{20}							I_{27}
($\overline{\text{RDY}}$)	$\overline{\text{TO}}$						

< Reader >

[Data]

I_{10}							I_{17}
$\overline{\text{RD}}_1$	$\overline{\text{RD}}_2$	$\overline{\text{RD}}_3$	$\overline{\text{RD}}_4$	$\overline{\text{RD}}_5$	$\overline{\text{RD}}_6$	$\overline{\text{RD}}_7$	$\overline{\text{RD}}_8$

[Control signals]

O_{20}							O_{27}
				$\overline{\text{STA}}$			

I_{20}							I_{27}
				$\overline{\text{SPR}}$	$\overline{\text{RB}}$		

Table 1. Port allocation

—Timing Chart—

Punch

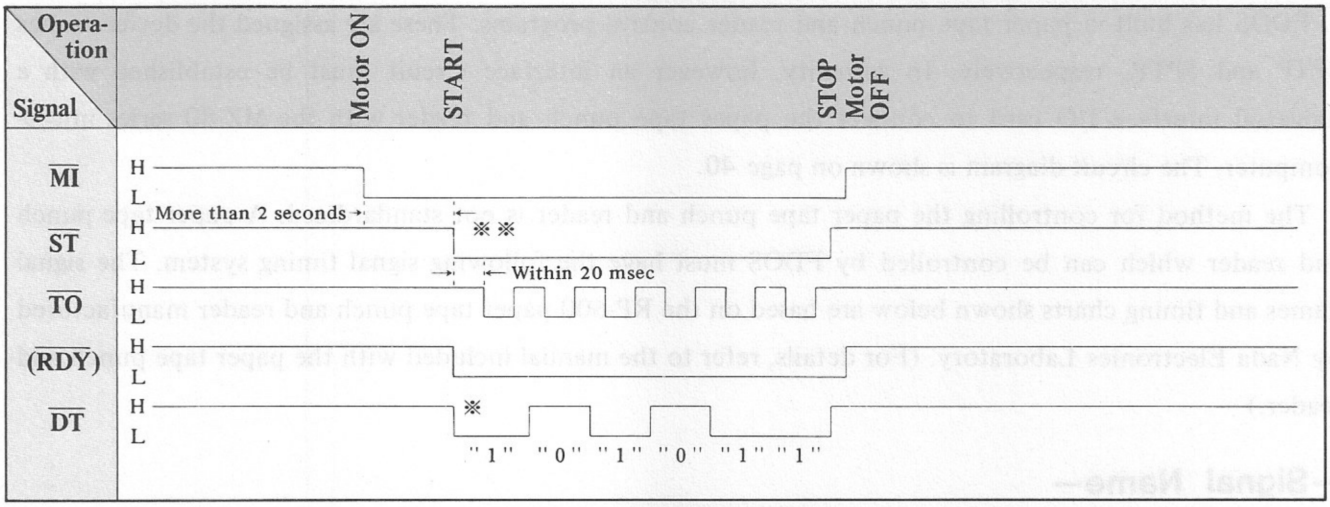


Figure 1. Punch timing chart

* The next data to be punched is readied while $\overline{\text{TO}}$ is H and maintained while $\overline{\text{TO}}$ is L.

** $\overline{\text{ST}}$ is set to L 2 or more seconds after the motor has been started, and is set to H after $\overline{\text{TO}}$ has risen from L to H for the last data.

Reader

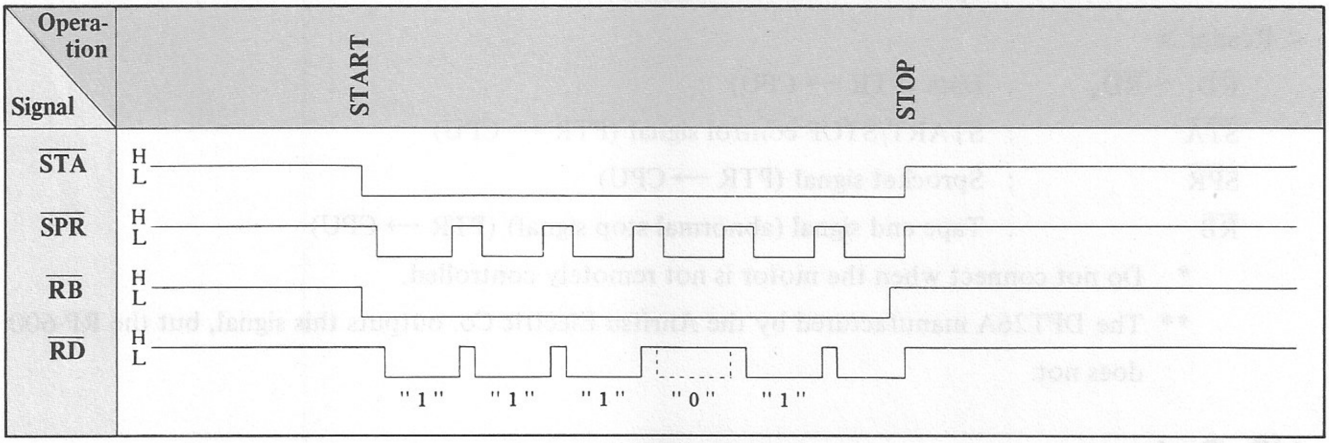


Figure 2. Reader timing chart

—Preparing a Paper Tape Punch/Reader I/O Card—

It is convenient to use a universal I/O card (MZ-80IO2) for preparing a paper tape punch and reader I/O interface circuit. Markings such as O_{10} or O_{17} in the port allocation table on page 13 match those on the universal I/O card. See page 16 for setting the universal I/O card switches to select port addresses FC and FD.

The RP-600 internal interface circuit and input and output pin connections are shown below for reference. (For details, refer to the manual included with the RP-600).

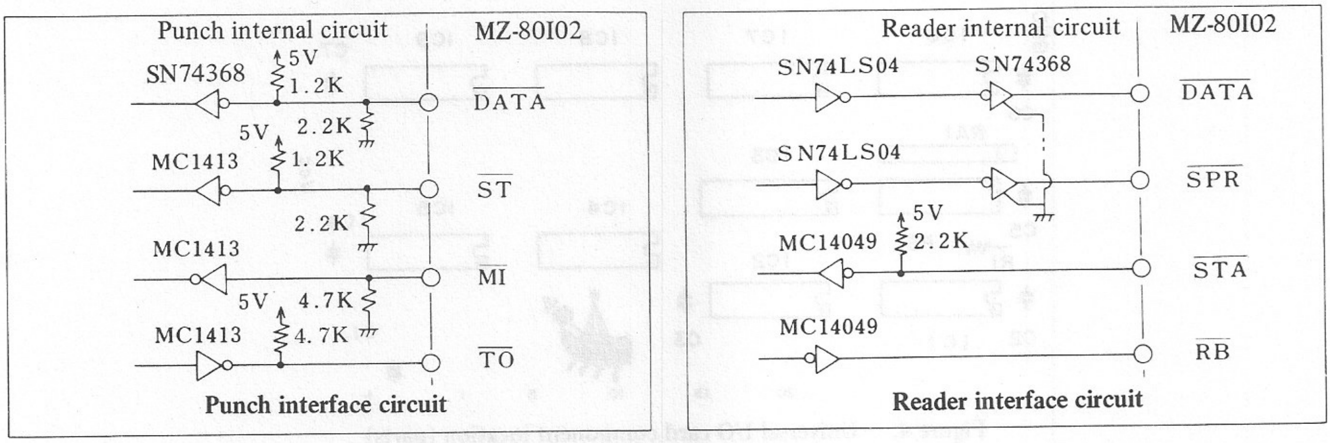


Figure 3. Interface circuit (RP-600)

Punch I/O connector

Pin	Signal		Pin	Signal
1	$\overline{DT}_1$	Data	14	
2	$\overline{DT}_2$		15	
3	$\overline{DT}_3$		16	
4	$\overline{DT}_4$		17	
5	$\overline{DT}_5$		18	
6	$\overline{DT}_6$		19	
7	$\overline{DT}_7$		20	
8			20	
9	$\overline{DT}_8$		21	
10			22	$\overline{\text{MI}}$ Motor ON/OFF signal
11	$\overline{\text{TO}}$	Timing signal		
12	GND		23	$\overline{\text{ST}}$ START/STOP signal
13			24	FG Frame ground

Reader I/O connector

Pin	Signal		Pin	Signal
1	$\overline{RD}_1$	Data	20	
2	$\overline{RD}_2$		21	
3	$\overline{RD}_3$		22	
4	$\overline{RD}_4$		23	
5	$\overline{RD}_5$		24	
6	$\overline{RD}_6$		25	
7	$\overline{RD}_7$		26	
8	$\overline{\text{SPR}}$	Sprocket signal	27	
9	$\overline{RD}_8$	Data	28	$\overline{\text{STA}}$ START/STOP signal
10				
11			29	$\overline{\text{RB}}$ Operating state
12	GND		30	FG Frame ground
13			31	
14			32	
15			33	
16			34	
17			34	
18			35	
19			36	

Table 2. Connector pin connections

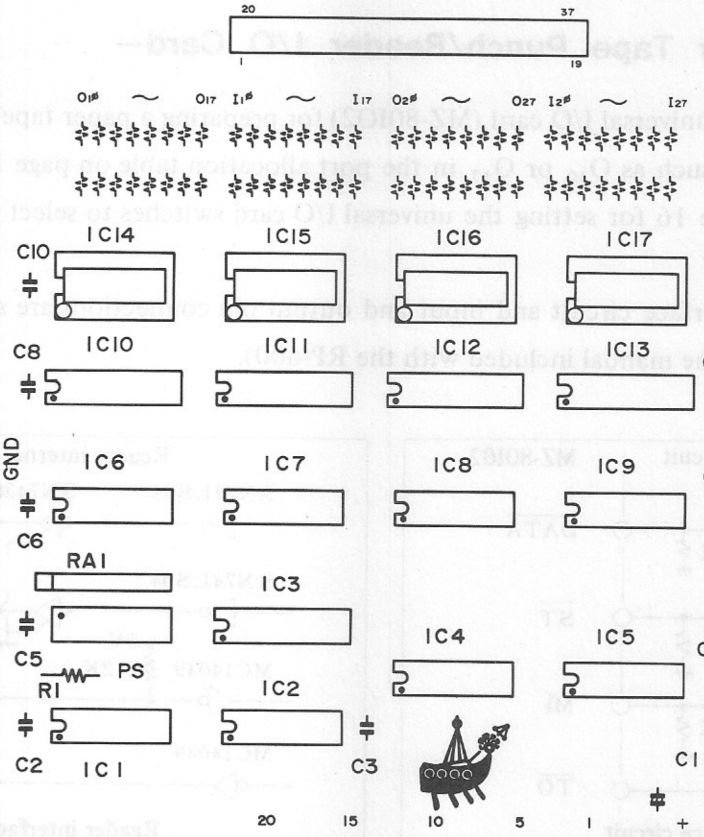


Figure 4. Universal I/O card component location (parts)

Universal I/O card port address setting

(1) Number of ports

Input : 2 ports Output : 2 ports

(2) Port address

All port addresses can be set. (However, FDOS uses $B0_H$ and higher locations.)

The input port for $I_{10} \sim I_{17}$ is set to an even address.

The input port for $I_{20} \sim I_{27}$ is set to an odd address.

The output port for $O_{10} \sim O_{17}$ is set to an even address.

The output port for $O_{20} \sim O_{27}$ is set to an odd address.

(3) Port address setting switches (PS)

Numbers marked on the PS switches correspond to the address bus lines shown below. Turning a PS switch OFF sets the corresponding address bit to logical "1" and turning it ON to logical "0".

Switch No.	7	6	5	4	3	2	1
Address bit	A_7	A_6	A_5	A_4	A_3	A_2	A_1

Example: Setting the PS switches as shown below sets the port address to FC_H.

1 1 1 1 1 1 0 0

↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑

S_7 S_6 S_5 S_4 S_3 S_2 S_1

When the PS switches are set as shown above, ports FC_H and FD_H are used for this card.

S_7 .... OFF

S_6 .... OFF

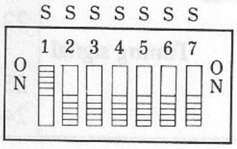
S_5 .... OFF

S_4 .... OFF

S_3 .... OFF

S_2 .... OFF

S_1 .... ON



Caution: Installing two or more interface cards which have the same port address settings will result in destruction of ICs.

COMMAND TABLES & ERROR MESSAGES

–FDOS Built-in Commands–

BOOT

Terminates the FDOS and activates sytem IPL.
Example: BOOT ↲

CHATR sign, filename1, attribute [, ...filenameN, attribute]

Matches the password's sign and changes the file attribute(s) of the matching file(s) identified by filename to attribute(s).

P: Permanent file	R: Read inhibit
0: No protection	W: Write inhibit

Examples: CHATR KEY, ABC, 0, XYZ, P ↲ : Deletes the file attribute of file ABC and changes the file attribute of file XYZ to PERMANENT if matches occur with the password KEY.
CHATR KEY, $FD2 ; UVW, R ↲ : Changes the file attribute of file UVW in FD2 to READ INHIBIT if a match occurs with the password KEY.
CHATR ↲ : This allows the programmer to interactively specify the password, file name and attribute.

DATE [MM . DD . YY]

Displays the current date or sets the specified date in month, date, year format. The set information is used as file information when new files are created.
Global switch /P : Specifies that the date is to be printed on the LPT.
Examples: DATE/P↲ : Lists the current date on the LPT.
DATE 12 . 25 . 81↲ : Sets the current date to December 25, 1981.

DELETE filename1 [, ..., filenameN] (? ,*)

Deletes the file(s) specified by filename(s).
Global switch /C : Specifies that each file name is to be displayed on the screen for verification. The programmer must enter Y to delete it or N to suppress deletion.
Examples: DELETE ABC . *↲ : Deletes all files identified by ABC. * .
DELETE/C A * . * ↲ : Displays files identified by A * . * on the screen for verification before deletion.
filename : ABC.ASC deleted ← Indicates that the file is deleted since "Y" is entered.
filename : ABC.RB ← Indicates that the file is not deleted "N" is entered.
filename : AXY.OBJ permanent ← Indicates that the file is not deleted because it is assigned the PERMANENT file attribute.

DIR [$FDn] or [filename] (? ,*)

Displays file information in the directory specified by $FDn or of the file specified by filename on the screen.
Global switch /P : Specifies that the file information is to be output to LPT. The file information is displayed on the screen when this switch is not specified.
Examples: DIR ↲ : Displays all file information in the current directory on the screen.
DIR/P $FD2 ↲ : Outputs all FD2 file names to LPT and switches the currently logged disk to FD2.
DIR $FD2 ; ABC. * ↲ : Displays the file information of files in FD2 identified by ABC. * .

EXEC filename

Executes the contents of the file identified by filename as FDOS commands.
Example: EXEC ABC . ASC ↲ : Sequentially executes the FDOS commands in file ABC.

FREE [$FDn]

Lists statistical information about the disk identified by $FDn on the screen or on the LPT.

Example: FREE $FD2 ↵

$FD2 master left : XXXX used : YYYY

Indicates that the diskette on FD2 is a master diskette, that the number of unused sectors is XXXX and that the number of used sectors is YYYY.

HCOPY control-code

Copies a data frame from the CRT screen to the LPT.

Examples: HCOPY ↵

MON

Terminates FDOS processing and returns control to the monitor.

Example: MON ↵

POKE $nnnn, data [, ..., $uuuu, dataN]

Stores data in the specified addresses in memory.

Example: POKE $000D, 2010, $000F, 40 ↵

RENAME oldname1, newname1 [, ..., oldnameN, newnameN]

Renames the file specified by oldname to newname.

Examples: RENAME ABC, XYZ ↵ : Renames file ABC to XYZ.

RENAME ABC, DEF, UVW, XYZ ↵ : Renames file ABC to DEF and UVW to XYZ.

RUN filename

Executes the program in the object file identified by filename.

Example: RUN ABC ↵ : Executes the program in file ABC, assuming it ot be ABC.OBJ.

TIME [HH : MM : SS]

Displays the current time or sets specified time in hour, minute, second format.

The current time is set to 00 : 00 : 00 upon system start.

Global switch /P : Specifies that the current time is to be listed on the LPT.

Examples: TIME/P ↵ : Lists the current time on the LPT.

TIME 16 : 30 : 30 ↵ : Sets the current time to 16 : 30 : 30

TYPE filename1 [, ..., filenameN] (?, ∗)

Lists the contents of the file(s) identified by filename(s) on the screen or on LPT.

Global switch /P : Lists the file contents on LPT.

Examples: TYPE ABC, DEF ↵ : Displays the contents of files ABC and DEF on the screen.

TYPE/P $FD3 ; XYZ ↵ : Lists the contents of file XYZ in FD3 on LPT.

TYPE $PTR ↵ : Reads paper tape data from PTR and displays it on the screen.

XFER sourcefile1, destinationfile2 [, ..., sourcefileN, destinationfileN] (sourcefile only ?, ∗)

Transfers the source file(s) to the destination file(s).

Examples: XFER ABC, XYZ ↵ : Copies file ABC to XYZ.

XFER $PTR, DEF ↵ : Transfers the file at the PTR to file DEF.

XFER XYZ, $PTP/PE ↵ : Transfers file XYZ to the PTP with even partiy in ASCII code.

–FDOS Transient Commands–

ASM filename

Assembles the source file identified by filename and produces a relocatable file and an assembly listing.

Global switch (none) : Specifies that the relocatable file is to be output.
Global switch/N : Suppresses generation of the relocatable file.
Local switch/O : Specifies that the relocatable file is to be output with the specified file name.
Local switch/E : Specifies that error statements are to be output to the specified file.
Local switch/L : Specifies that the listing is to be directed to the specified file.

Examples:
ASM ABC ↲ : Assembles source file ABC and generates relocatable file ABC.RB.
ASM/N/ABC, $CRT/E ↲ : Assembles source file ABC and displays error statements on the screen (no relocatable file is created).
ASM/ABC, XYZ/O, $LPT/L ↲ : Assembles source file ABC and generates relocatable file XYZ.RB and an assembly listing on the LPT.
ASM/ABC, $FD2 ; XYZ/L, $LPT/E ↲ : Assembles source file ABC outputs the assembly listing to file XYZ.ASC in FD2 and outputs error statements on the LPT.

ASSIGN devicename, address

Sets the address of a user device drive routine.

Examples: ASSIGN $USR1, $B000 ↲ : Sets the drive routine address of user device $USR1 to B000 (hexadecimal).

BASIC filename

Invokes the BASIC compiler to compile the source program identified by filename.

Example: BASIC XYZ ↲ : Invokes the BASIC compiler, compiles source file XYZ.ASC and generates relocatable file XYZ.RB.

CONVERT

Converts a file generated with the SA-5000 series BASIC interpreter or the D-BASIC SA-6000 series into a file which can be used under FDOS, or converts a file generated with FDOS into a file which can be used under the SA-5000 series BASIC interpreter or the D-BASIC SA-6000 series.

Example: CONVERT ↲

COPY

Copies the files on the diskette in drive 1 to the diskette in drive 2. The system matches the passwords in these diskettes before carrying out a copy operation.

Example: COPY ↲

DEBUG filename [, ..., filenameN]

Invokes the symbolic debugger and links and loads relocatable file(s).

Global switch /T : Specifies that the symbol table information is to be output.
Global switch /P : Specifies that the listing is to be directed to the LPT (the listing is displayed on the screen if omitted).
Local switch /O : Specifies that the object file is to be generated with the specified file name.

Example: DEBUG ABC, DEF ↲ : Invokes the symbolic debugger, links and loads relocatable files ABC and DEF and waits for a symbolic debugger command.

EDIT [filename]

Loads the text editor and reads in the file (if specified). The file must be an ASC mode file.

Examples:
EDIT ↲ : Loads the text editor and waits for an editor command.
EDIT $FD2 ; ABC ↲ : Loads the text editor and reads in file ABC from FD2.

FORMAT [$FDn]

Initializes the diskette in $FDn in the system format. The pasword set by the SIGN command is checked before execution.

Examples: FORMAT ↲ : Initializes the currently logged-on diskette.
FORMAT $FD2 ↲ : Initializes the diskette in FD2.

LIBRARY filename1 [, ..., filenameN]

Links specified file(s) into a library file.

Global switch (none) : Specifies that the link information is to be displayed on the screen.
Global switch /P : Specifies that the link information is to be printed on the LPT.

Examples: LIBRARY ABC, DEF, ↲ : Links relocatable files ABC and DEF and stores their contents into library file ABC.LIB
LIBRARY ABC, DEF, XYZ /O ↲ : Links relocatable files ABC and DEF and stores their contents into library file XYZ.LIB.

LIMIT address

Sets or changes the end address of the memory area managed by FDOS.

Examples: LIMIT $B000 ↲ : Sets the FDOS area to B000 (hexadecimal).
LIMIT MAX ↲ : Sets the FDOS area to the maximum available address.

LINK filename1 [, ..., filenameN]

Links relocatable files identified by filename1 through filenameN and outputs an object file with a link table listing.

Global switch /T : Specifies that the symbol table information is to be listed.
Global switch /P : Specifies that the listing is to be directed to the LPT (the listing is displayed on the screen if the switch is omitted).
Global switch /S : Specifies that a system file is to be generated.

Examples: LINK ABC, DEF ↲ : Links relocatable files ABC and DEF and outputs object file ABC.OBJ
LINK/T/P ABC, DEF, XYZ /O↲: Links relocatable files ABC and DEF and outputs object file XYZ.OBJ with the link table information on the LPT.

LOAD filename

Loads the object file identified by filename into the area immediately following the area established by the LIMIT command.

Example: LOAD ABC.OBJ ↲ : Loads object file ABC.OBJ into memory.

MLINK filename1 [, ..., filenameN]

Links relocatable files identified by filename1 through filenameN and outputs an object file with a link table listing. This command can link files to generate an object file of up to 30K bytes, although the LINK command can only deal with up to 20K bytes.

Global switch /T : Specifies that the symbol table information is to be listed.
Global switch /P : Specifies that the listing is to be output on the LPT (the listing is displayed on the screen if this switch is omitted).

Example: MLINK ABC, DEF ↲ : Links relocatable files ABC and DEF and outputs object file ABC.OBJ.

PAGE [output-device] or nn

Performs a form feed operation on the output device identified by output-device, or sets the number of lines per page on the LPT.

Examples: PAGE↵ : Moves the print position to the home position of the printer form.
PAGE 22↵ : Sets the number of lines per page on the LPT to 22. The print form is fed to the top of the next page when a page feed code is issued or the TOP OF FORM button is pressed.

PROM

Generates formatted code on the paper tape punch from an object file. Applicable PROM writers are those which are supplied by Britronics, Intel, Takeda and Minato Electronics.

Example: PROM↵

SIGN [$FDn]

Changes the password of the diskette in $FDn.
During a diskette copy or formatting operation, the system checks the programmer-specified password with that stored in the diskette directory for a match and carries out the specified operation only when a match occurs.

Example: SIGN↵ : Changes the password of the diskette currently logged on.

STATUS devicename, status

Sets the status of the I/O device identified by devicename to status.

Example: STATUS $SIA, $1234↵ : Sets the control status of serial input port A to 1234 (hexadecimal).

VERIFY filename1, filename2 [, ..., filenameN-1, filenameN] **(?, ∗ only for filename1, ..., filenameN-1])**

Compares the contents of files filename1 through filenameN.

Global switch /P : Specifies that the results of the comparison are to be listed on the LPT.

Example: VERIFY $CMT, $FD2 ; ABC↵ : Compares the first file on the cassette tape with source file ABC in FD2.

—System Error Messages—

There are four system error message formats.

— ERR: error message

Pertains mainly to coding errors. Issued when invalid commands are detected.

— ERR: filename (device name) : error message

Indicates errors pertaining to file or device specifications.

— ERR: logical number: error message

Indicates errors pertaining to logical number specifications.

— ERR: logical number file name (device name): error message

Indicates errors pertaining to logical number specifications and file (or device) specifications.

The system error messages are listed below. The error numbers are not output.

ERR- 1	syntax	
2	il command	
3	il argument	
4	il global switch	
5	il data	
6	il attribute	; Illegal file attribute found
7	different file mode	
8	il local switch	
9	il device switch	
10		
11	no usable device	; Device unavailable
12	double device	
13	directory in use	
14		
15		
16	not enough arguments	
17	too many argument	
18		
19		
20	no memory space	
21	memory protection	
22	END ?	
37	Break	
38	system id	; Diskette not conforming to FDOS format.
39	System error	; System malfunction, user program error, diskette replaced improperly, etc.

50	not found	
51	too long file	; File size exceeds 65535 bytes
52	already exist	
53	already opened	; The file has been already opened or the logical number is already used.
54	not opended	
55	read protected	
56	write protected	
57	permanent	
58	end of file	
59	no byte file	
60	not ready	
61	too many files	; Number of files exceeds 96
62	disk volume	; Diskette replaced improperly
63	no file space	
64	unformat	; Diskette unformatted
65	FD hard error	; Hardware related disk error
66	il data	
67	no usable diskette	
68	(sub)master diskette	
69	mismatch sign	
70	il file name	; Invalid file name
71	il file attribute	; Invalid file attribute
72	il file type	; Invalid file type
73	il file mode	; Invalid file mode
74	il lu#	; Invalid logical number
75	not ready	; Printer error
76	alarm	
77	paper empty	
78	time out	; Paper tape reader or punch error
79	parity	
80	check sum	
89	lu table overflow	; Attempt made to open too many files
90	source ?	
91	destination	
92	can't xopen	
93	too long line	; Line exceeding 128 bytes
94	end of record	
95	diff record length	

—Editor Commands—

Command type	Command name	Function
Input command	R	Clears the edit buffer and loads it wih the input file indicated by the filename. The CP is positioned at the beginning of the edit buffer after execution of this command.
	A	Appends the input file indicated by the filename to the contents of the edit buffer. The CP position is not changed.
Output command	W	Writes the edit buffer contents to the output file specified by the filename in ASCII code.
Page processing command	PR PA	Loads the remainder of a file whose beginning has been loaded with the R or A command. The PR command clears the edit buffer before the data is loaded and the PA command adds the data to the current contents of the edit buffer.
	PW	Same as the W command, except that the maximum amount of data output is 1 page.
	PC	Terminates execution of the processing command. This command is required whenever a PR, PA or PW command is executed.
	PK	Kills the file opened by a page processing command.
Type command	T	Displays the entire contents of the edit buffer. The CP position is not changed.
	nT	Displays n lines starting at the CP position.
CP positioning command	B	Positions the CP at the beginning of the edit buffer.
	nJ	Positions the CP at the beginning of the line indicated by n.
	nL	Moves the CP to the beginning of the line n lines after the current CP position.
	L	Moves the CP to the beginning of the current line. This is the same as when n = 0 in the nL command.
	nM	Changes the CP position by n characters.
	M	Does not move the CP. This is the same as when n = 0 in the nM command.
	Z	Moves the CP to the end of the text in the edit buffer.
Correction command	C	Searches for the specified character string and replaces it with another character string; the search starts at the current CP position and proceeds to the end of the edit buffer. The CP is repositioned to the end of the character string replaced.
	Q	Repeats the C command each time the specified character string is found until the end of the edit buffer is reaches. The CP is repositioned to the end of the character string last replaced.
	I	Inserts the specified character string at the position of the CP. The CP is repositioned to the end of the character stirng inserted. Line numbers are updated when a line is inserted with this command.
	nK	Deletes the n lines following the CP. The CP position is not changed.
	K	Deletes all characters preceding the CP position until a $\boxed{\text{CR}}$ code is detected. The $\boxed{\text{CR}}$ code is not deleted.
	nD	Deletes the n characters following the CP.
	D	No operation
Search command	S	Searches for the specified character string, starting at the CP position and proceeding to the end of the buffer. The CP is repositioned to the end of the character stirng when it is found.
Special command	\	Executes the specified built-in command.
	=	Displays the number of characters stored in the edit buffer (including spaces and CRs).
	.	Displays the number of the line at which the CP is located.
	&	Deletes the entire contents of the edit buffer.
	#	Changes the list mode for listing to the printer.
	!	Transfers control to the FDOS.

Most of the above commands are compatible with those used in the NOVA editor program manufactured by the Data General Corporation.

—Editor Error Messages—

Error message	Explanation	Related commands
Full buffer	The edit buffer is full.	R, A, PR, PA
? ? ?	n < 0 in the nT or nJ command.	T, J
Large	n greater than 65535 was specified.	T, J, L, M, K, D, B, Z
Not found	The string specified in the command was not found.	S, C, Q
Invalid	Other than an editor command was entered or an incorrect format was used. Ex) * H $\boxed{\text{CR}}$: There is no H command. * S $\boxed{\text{CR}}$: A string should be specified.	Any case
Page opened ?	The file to be subjected to page processing is not defined (or is not opened).	PR, PA, PW, PC
Page opened !	An attempt to execute the ! or \command was made on a file which was subjected to page processing, but which was not closed or killed by the PC or PK command.	! , \
No file is saved after edition End of job?	These messages are displayed where an attempt is made to execute a ! command after the edit buffer contents have been corrected without first executing a W or PW command. Pressing the $\boxed{\text{Y}}$ key in this case executes the ! command. Pressing the $\boxed{\text{N}}$ key causes the system to await entry of a new command.	!

Note: Refer to the System Error Messages in the System Command manual for the system errors.

Display of the message "Already opened" during execution of a W command indicates that there is a PW command which has not been closed.

—Assembler Messages—

Definition status message	Meaning	Example
E (External)	Indicates that a label symbol is being referenced externally; that is, the label is not defined in the current source program unit.	E LD B, CONST0 The data byte "CONST0" is undefined. E CALL SORT The address "SORT" is undefined. EE BIT TOP, (IY+FLAG) The data byte "FLAG" is undefined. The data byte "TOP" is undefined.
P (Phase)	Defines a label symbol with a constant assigned. This message is also output when a label symbol is encountered during pass 2 which was not encountered during pass 1.	P LETNL : EQU 0762H P DATA1 : EQU 3 LETNL and DATA1 are defined by EQU. The P message is displayed in the relocatable binary code column rather than in the assembler message column.

Error message	Meaning	Example
C (illegal Character error)	Indicates that an illegal character is used in the operand.	C JP +1000–3
F (Format error)	Indicates that the instruction format is incorrect.	
N (Non label error)	Indicates that no label symbol is specified for ENT or EQU.	N EQU 0012H No label symbol
L (erroneous Label error)	Indicates that an illegal label symbol is used.	L JR XYZ XYZ is not defined in the current program. No externally defined global symbol can be used as the operand of a JR or DJNZ command. If such a label symbol is specified, the L message is displayed.
M (Multiple label error)	Indicates that a label symbol is defined two or more times.	M ABC : LD DE, BUFFER ≀ M ABC : ENT ABC is defined twice.
O (erroneous Operand)	Indicates that an illegal operand is specified.	
Q (Questionable mnemonic)	Indicates that the mnemonic code is incorrect.	Q CAL XYZ CALL XYZ is correct.
S (String error)	Indicates that single or double quotation mark(s) are omitted.	S DEFM GAME OVER DEFM 'GAME OVER' is correct.
V (Value over)	Indicates that the value of the operand is out of the prescribed range.	V LD A, FF8H V SET 8, A V JR –130
END?	Indicates that the END directive is missing from the source program.	

Note: Refer to the System Error Messages in the System Command manual for other system errors.

—Symbolic Debugger Commands—

Command type	Command name	Function
Symbol table command	T	Displays the contents of the symbol table; i.e., the label symbol name, its absolute address and the definition status for each table entry. (Table Dump)
Debugging commands	B†	Displays, sets or alters a breakpoint. (Breakpoint)
	&	Clears all breakpoints set. (Clear Breakpoints)
	M†	Displays the contents of the specified block in the link area in hexadecimal representation or alters them. (Memory Dump)
	D†	Displays the contents of the specified block in the link area in hexadecimal representation with one instruction on a line. (Memory List Dump)
	W†	Writes hexadecimal data, starting at the specified address in the link area. (Write)
	G†	Executes the program at the specified address. (GOTO)
	I	Executes the program at the address designated by PC with the register buffer data set to the CPU internal reigsters. (Indicative Start)
	A	Displays the contents of registers A, F, B, C, D, E, H and L in hexadecimal representation or alters them. (Accumulator)
	C	Displays the contents of complementary registers A′, F′, B′, C′, D′, E′, H′ and L′ in hexadecimal representation or alters them. (Complementary)
	P	Displays the contents of registers PC, SP, IX, IY and I in hexadecimal representation or alters them. (Program Counter)
	R	Displays the contents of all registers in hexadecimal representation. (Register)
	X	Transfers the specified memory block to the specified address. (Transfer)
File I/O commands	S	Saves the object program in the link area in an output file with the specified name. (Save)
	Y	Reads the object program from the object file with the specified file name into memory. (Yank)
Special commands	\	Executes the specified FDOS built-in command.
	#	Switches the printer list mode for listing printout.
	!	Transfers control to FDOS.

Note: Commands marked by a dagger permit symbolic operations.

—Symbolic Debugger Error Messages—

Error message	Description	Related commands
???	○ The command operand fields does not match the 4-digit hexadecimal format. ○ A symbolic label is missing. ○ A data defining symbol is used as a label.	M, D, W, B, G
Error	○ An invalid number of digits was entered when altering register or memory contents, or a key other than 0 through 9 or A through F was pressed.	A, C, P, M
DJNZ?	A breakpoint was set for a DJNZ instruction.	B
CALL?	A breakpoint was set for a CALL instruction.	B
RST 6?	A breakpoint was set for a RST 6 instruction.	B
Over	An attempt was made to set more than 9 breakpoints.	B
?	○ An attempt was made to access outside the link area. ○ The starting address is greater than the ending address. ○ An attempt was made to clear an undefined breakpoint. ○ The breakpoint counter was set to F (the maximum permissible value is E in hexadecimal).	M, D, W, B, G, X M, D B B

Note: Refer to the System Error Messages in the System Command manual for other system error messages.

—PROM Formatter Commands—

COMMAND		OPERATION
File Input/ Output commands	Y (Yank)	Loads a program (data) from the diskette into the free area.
	S (Save)	Saves the program (data) in the free area on diskette.
	CY (Yank disk)	Loads data in 256-byte units from the specified sector(s) of the specified track on the diskette into RAM.
	CS (Save disk)	Saves data in 256-byte units from RAM memory in the specified sector(s) of the specified track of the diskette.
Format commands	P (Punch)	Punches the specified contents of the free area in the specified format.
	R (Read)	Reads in a paper tape punched in the format specified.
Other commands	M (Memory)	Displays and modifies data in the free area.
	V (Verify)	Reads data from the paper tape reader and companies it with the contents of the RAM free area.
	\ (FDOS)	Executes the specified built-in FDOS command.
	#	Switches the list mode for listing on a printer.
	& (Clear)	Buries all data in the free area in hexadecimal code FFH.
	?	Displays the starting and ending addresses of the free area.
	! (Return)	Returns control to FDOS.

Error message	Error content	Related command
memory protection	An address outside of the free area was specified.	Y, S, P, R, M, V
il command	The command was not entered correctly.	
il data	The format specified does not match the format read.	R, V
check sum	Check sum error.	R, V
$ LPT : not ready	The printer is not ready.	#
$ PTP : not ready	The paper tape punch is not ready.	P
$ PTR : not ready	The paper tape reader is not ready.	R, V

See the "System Error Messages" in System Command for other error messages.

Caution:

Entry of characters other than S, Y, CS, CY, P, R, M, V, \, &, #, ? or ! will cause a return to the command wait state after the command table is displayed.

If a character other than A~H is input while "format?" is displayed and format entry awaited, the format table will be displayed, after which the format entry wait state will be reentered. A return can be made to the command wait state at this time by pressing $\boxed{\text{BREAK}}$.

—FILE Mode—

File mode	Meanings
.ASC	ASCII file. A source file generated by the text editor or a file containing ASCII character strings generated by a BASIC interpreter.
.RB	Relocatable file. A file containing pseudo-machine language code (relocatable binary code) which can be loaded into any location in memory. It is generated by the assembler or the compiler.
.OBJ	Object file. A file containing Z-80 machine language codes.
.LIB	Library file. A file into which FDOS links multiple relocatable files.
.SYS	System file. A file containing a system program runs under FDOS and which contains relocatable binary codes (such as the text editor and the assembler).

—I/O Devices Handled by FDOS—

$KB	MZ-80A system keyboard
$CRT	MZ-80A system display unit
$FD1	Floppy disk drives (MZ-80FB or MZ-80FBK)
$FD2	
$FD3	
$FD4	
$CMT	System cassette tape deck
$LPT	Optional printer
$MEM	A part of MZ-80 main memory
$PTR	Paper tape reader
$PTP	Paper tape punch
$SIA	Serial input port A
$SIB	Serial input port B
$SOA	Serial output port A
$SOB	Serial qutput port B
$USR1	User devices 1 ~ 4
$USR2	
$USR3	
$USR4	

—File Attributes—

File attributes are information pertaining to file protection. There are four types of file attribute: 0, R, W and P. File attribute 0 indicates that a file is not protected. The other attributes inhibit the use of specific commands as indicated below.

File attribute	R	W	P	
Inhibited FDOS commands	TYPE XFER EDIT ASM LINK DEBUG PROM BASIC	DELETE RENAME	TYPE XFER EDIT ASM LINK DEBUG PROM BASIC DELETE RENAME	0 : No file protection R: Read-inhibited file W: Write-inhibited file P: Permanent file
Inhibited BASIC statements	ROPEN# INPUT#( )	PRINT#( )	ROPEN# INPUT#( ) PRINT#( )	

ASCII CODE TABLE

The following are the ASCII codes for characters:

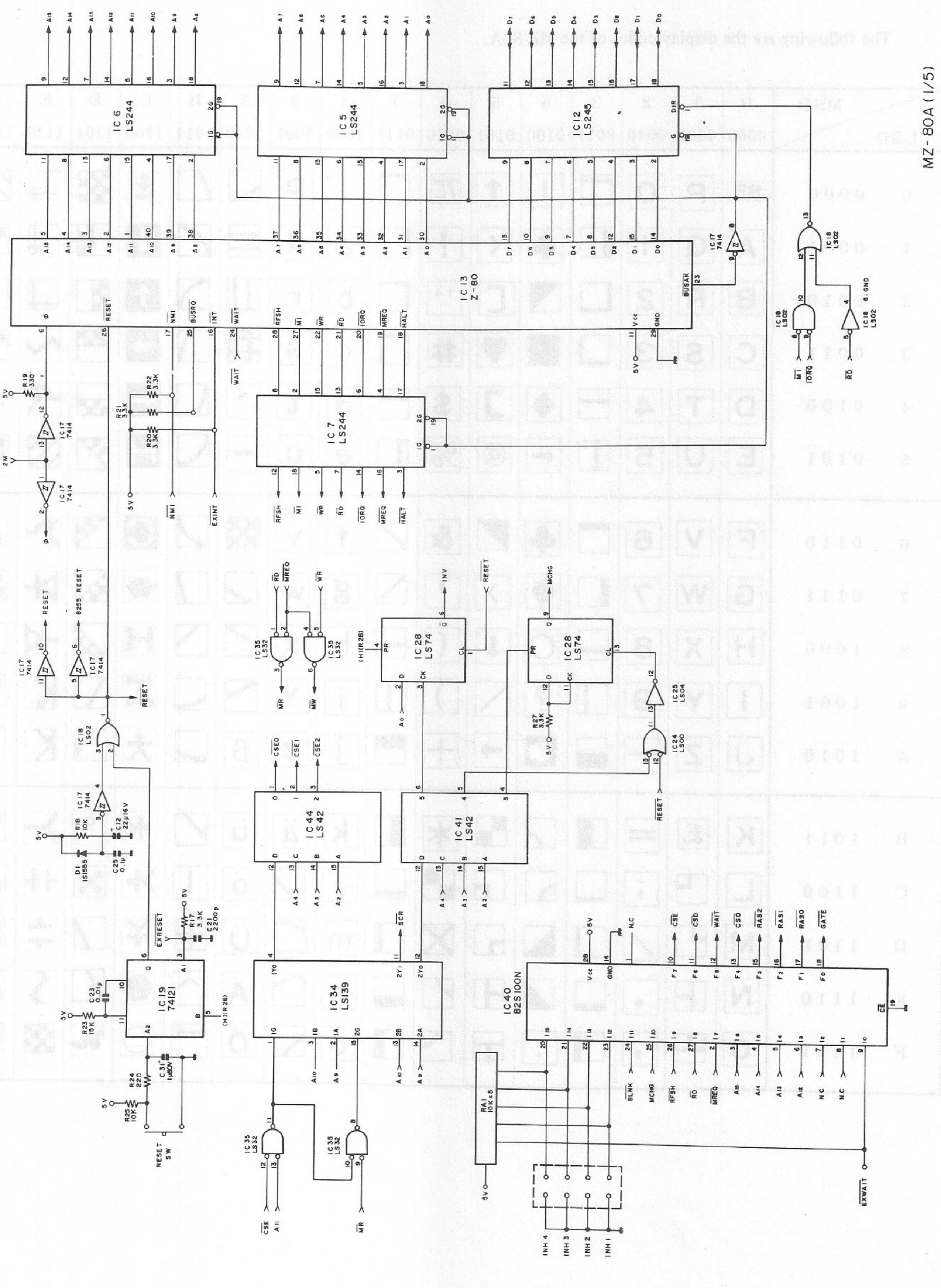
MSD LSD		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0	0000			SP	0	@	P	👾	▒	}	▔	q	n	│	┌	─	┐
1	0001		↓	!	1	A	Q	H	▐	⊞	▦	a	╱	▐	┴	♠	●
2	0010		↑	"	2	B	R	I	▌	╭	e	z	Ü	▄	┬	┃	═
3	0011		→	#	3	C	S	웃	▀	╭	`	w	m	▔	┤	─	♥
4	0100		←	$	4	D	T	K	▄	╰	~	s	⟋	▁	▎	═	‖
5	0101		H	%	5	E	U	⅄	◤	╰	▩	u	⟋	▏	▌	═	◢
6	0110		C	&	6	F	V	¥	◥	(	t	i	╲	→	▐	═	╳
7	0111			'	7	G	W	☻	◣	╮	g	≡	o	▕	▔	▏	○
8	1000			(	8	H	X	☺	◢	╮	h	ö	l	█	▀	▕	♣
9	1001			)	9	I	Y	∿	╲	╯	╭	k	Ä	◤	▄	◣	▕
A	1010			*	:	J	Z	⊳	╲	╯	b	f	ö	▐	┘	╰	♦
B	1011			+	;	K	[	⊲	°	^	x	v	ä	├	╭	╯	£
C	1100			,	<	L	\	K	⁙	╮	d	‖	╲	◙	╮	▁	↓
D	1101	CR		—	=	M	]	K	╱	)	r	ü	y	└	┘	╲	┃
E	1110			.	>	N	↑	⊣⊢	╱	)	p	β	{	┐	▞	╱	◥
F	1111			/	?	O	←	÷		╯	c	j	╮	▂	▚	▔	π

DISPLAY CODE TABLE

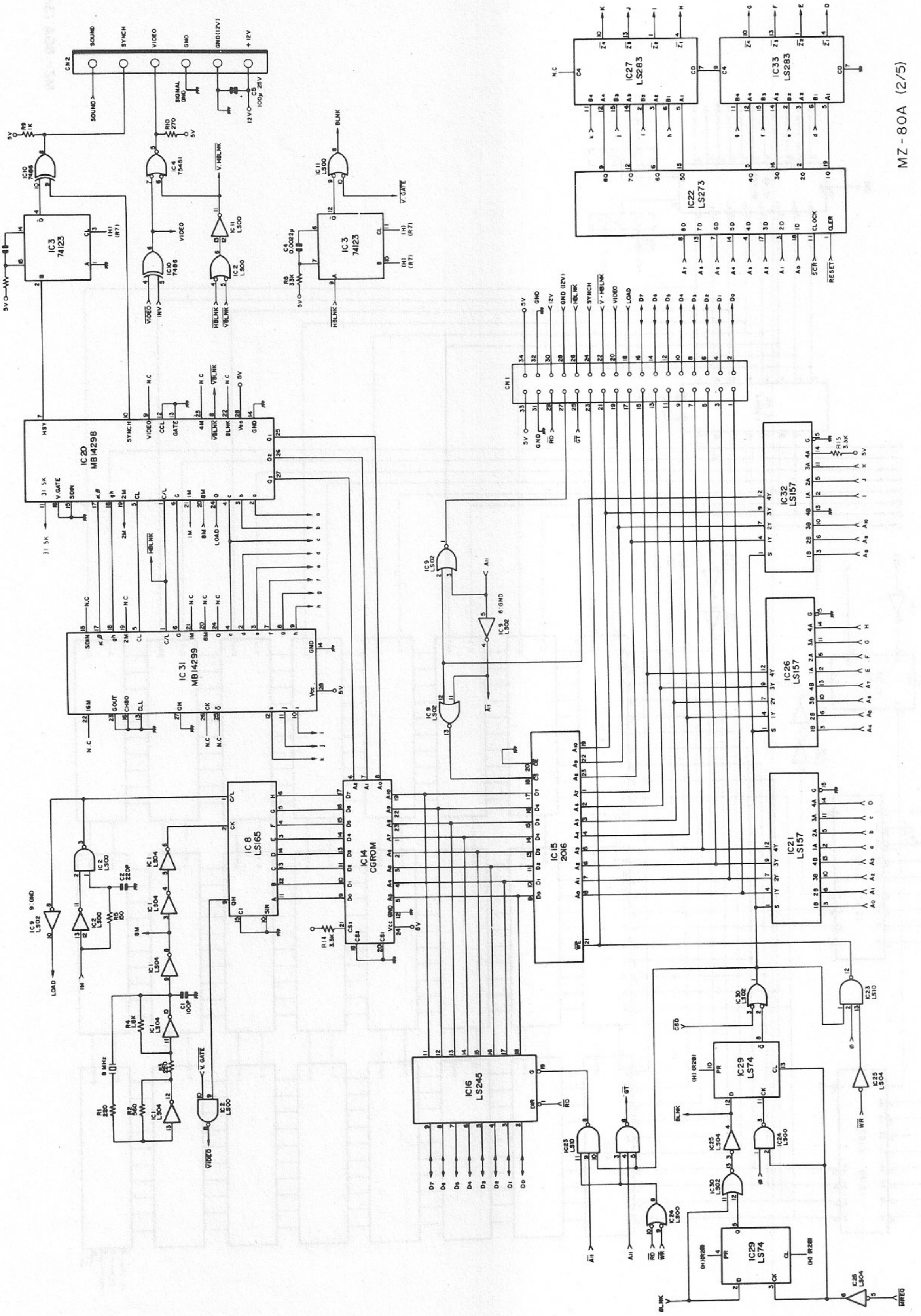
The following are the display codes of the MZ-80A.

LSD \ MSD		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0	0000	SP	P	O	▔	}	↑	π	▔	|	p	◣	◜	⤓	▦	┴	▨
1	0001	A	Q	1	▏	♠	<	!	▏	a	q	≡	◜	↓	▦	┤	▘
2	0010	B	R	2	▁	◥	[	"	▁	b	r	‖	◝	↑	▦	┘	▝
3	0011	C	S	3	▕	■	♥	#	▕	c	s	╬	◝	→	▦	∼	▀
4	0100	D	T	4	─	◆	]	$	─	d	t	`	◟	←	▦	⌐	▖
5	0101	E	U	5	│	←	@	%	│	e	u	~	◟	H	▦	▧	▌
6	0110	F	V	6	▔	♣	◩	&	╱	f	v	▩	◞	C	▦	⋋	▞
7	0111	G	W	7	▏	●	>	'	╲	g	w	◢	◞	✺	▦	⋈	▛
8	1000	H	X	8	▁	○	↓	(	─	h	x	◤	◹	H	▦	⋈	▗
9	1001	I	Y	9	▕	?	╲	)	│	i	y	◥	◟	I	╲	K	▚
A	1010	J	Z	−	▄	◙	→	+	▀	j	z	β	◟	大	╲	K	▐
B	1011	K	£	=	▐	◜	▚	*	▌	k	ä	ü	╱	✱	°	⊱	▜
C	1100	L	⌟	;	▁	◝	▗	▚	▁	l	◢	ö	{	✱	✕	╫	▄
D	1101	M	⌜	/	▕	◣	▖	☒	▕	m	◸	Ü	⊞	¥	╱	÷	▙
E	1110	N	╞	.	▁	◢	╪	◞	▄	n	◹	Ä	^	☻	╱	Ƨ	▟
F	1111	O	╧	,	▕	:	╤	◝	▐	o	◺	Ö	‾	☺	∿	▦	█

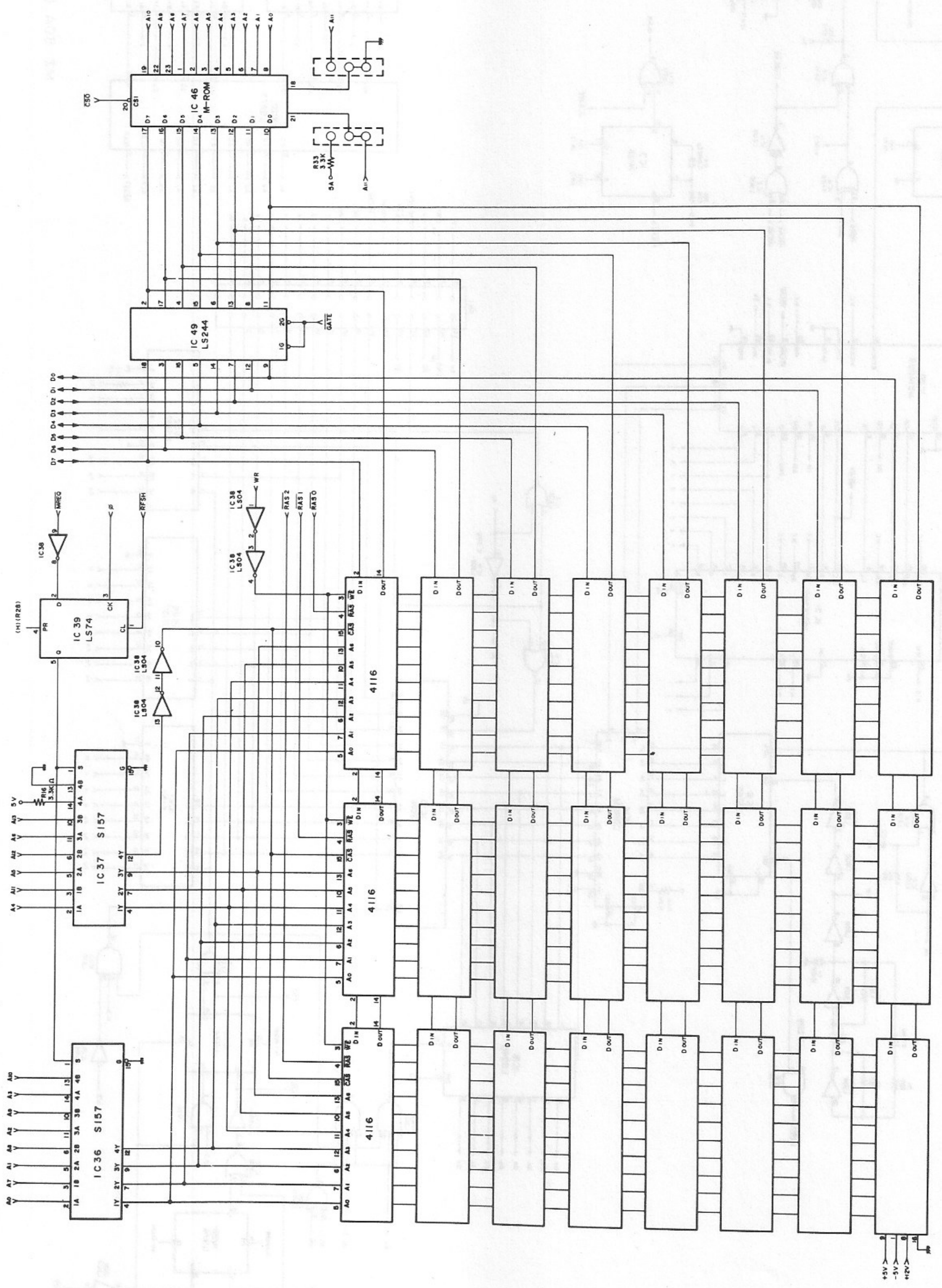
MZ-80A CIRCUIT DIAGRAMS



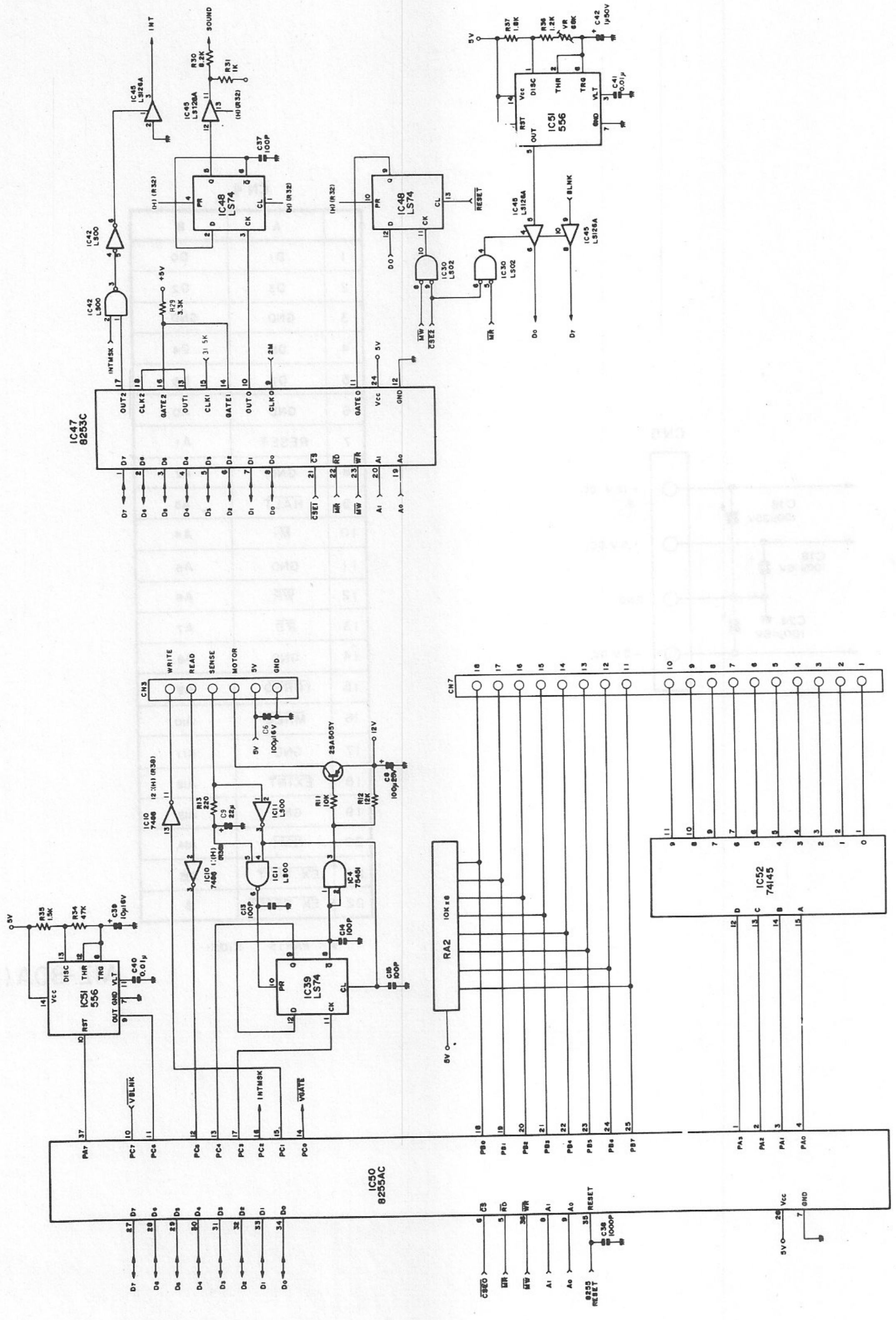
CPU board, block 1 : CPU signal system



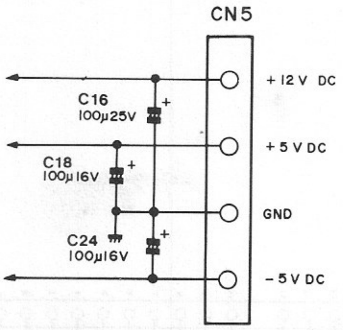
CPU board, block 2



CPU board, block 3: RAM signal system



CPU board, block 4 : 8255 and 8253 signal system



CN4

	A	B
1	D_1	D_0
2	D_3	D_2
3	GND	GND
4	D_5	D_4
5	D_7	D_6
6	GND	A_0
7	RESET	A_1
8	GND	A_2
9	$\overline{HALT}$	A_3
10	$\overline{M_1}$	A_4
11	GND	A_5
12	$\overline{WR}$	A_6
13	$\overline{RD}$	A_7
14	GND	A_8
15	$\overline{IOREQ}$	A_9
16	$\overline{MREQ}$	A_{10}
17	GND	A_{11}
18	$\overline{EXINT}$	A_{12}
19	GND	A_{13}
20	$\overline{NMI}$	A_{14}
21	$\overline{EX\ WAIT}$	A_{15}
22	$\overline{EX\ RESET}$	ϕ

A : PARTS SIDE

MZ-80A(5/5)

CPU board, block 5 : Peripheral I/O port and power terminal

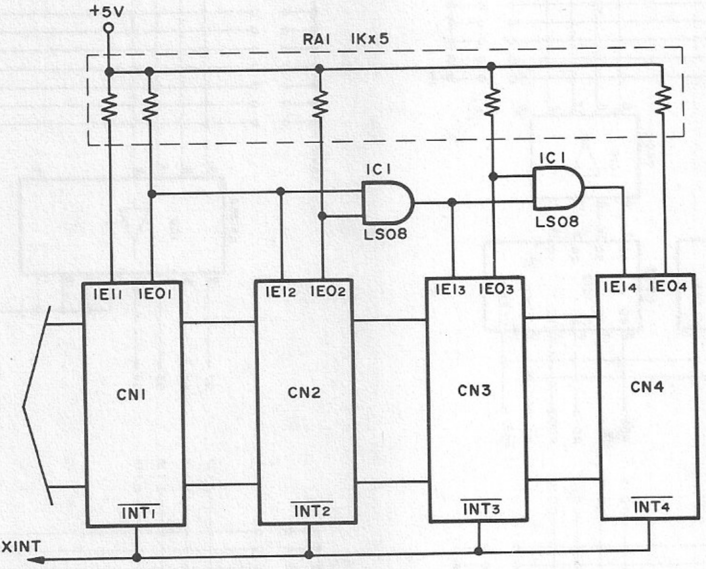
MZ-80AEU CIRCUIT DIAGRAMS

MZ-80AEU I/O MOTHER BOARD

CN5

A		B
D1	1	D0
D3	2	D2
GND	3	G
D5	4	D4
D7	5	D6
GND	6	A0
RESET	7	A1
GND	8	A2
$\overline{\mathrm{HALT}}$	9	A3
$\overline{\mathrm{M1}}$	10	A4
GND	11	A5
$\overline{\mathrm{WR}}$	12	A6
$\overline{\mathrm{RD}}$	13	A7
GND	14	A8
$\overline{\mathrm{IOREQ}}$	15	A9
$\overline{\mathrm{MREQ}}$	16	A10
GND	17	A11
$\overline{\mathrm{EXINT}}$	18	A12
GND	19	A13
$\overline{\mathrm{NMI}}$	20	A14
$\overline{\mathrm{EXWALT}}$	21	A15
$\overline{\mathrm{EXRESET}}$	22	Ø

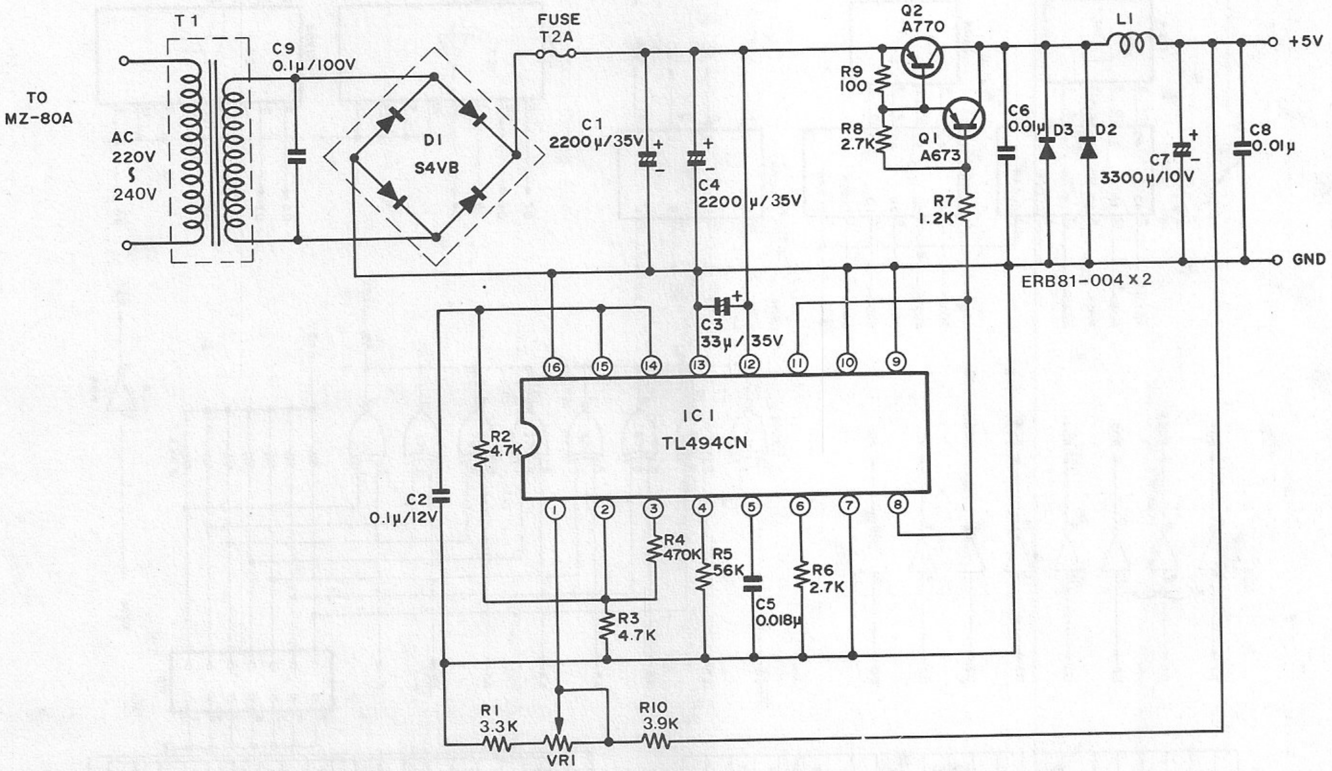
A: PARTS SIDE


+5V
RA1 1Kx5
IC1
LS08
IC1
LS08
IEI1 IEO1
IEI2 IEO2
IEI3 IEO3
IEI4 IEO4
CN1
CN2
CN3
CN4
$\overline{\mathrm{INT1}}$
$\overline{\mathrm{INT2}}$
$\overline{\mathrm{INT3}}$
$\overline{\mathrm{INT4}}$
EXINT

CN1~CN4

A		B
+5V	1	+5V
D2	2	D3
D1	3	D4
D0	4	D5
GND	5	D6
A15	6	D7
A14	7	Ø
A13	8	$\overline{\mathrm{M1}}$
A12	9	$\overline{\mathrm{WR}}$
A11	10	$\overline{\mathrm{RD}}$
A10	11	$\overline{\mathrm{IOREQ}}$
A9	12	$\overline{\mathrm{MREQ}}$
A8	13	GND
A7	14	$\overline{\mathrm{HALT}}$
A6	15	IEI
A5	16	IEO
A4	17	RESET
A3	18	$\overline{\mathrm{EXRESET}}$
A2	19	$\overline{\mathrm{EXINT}}$
A1	20	$\overline{\mathrm{EXWAIT}}$
A0	21	$\overline{\mathrm{NMI}}$
GND	22	GND

A: PARTS SIDE



UNIVERSAL I/O CARD CIRCUIT DIAGRAM

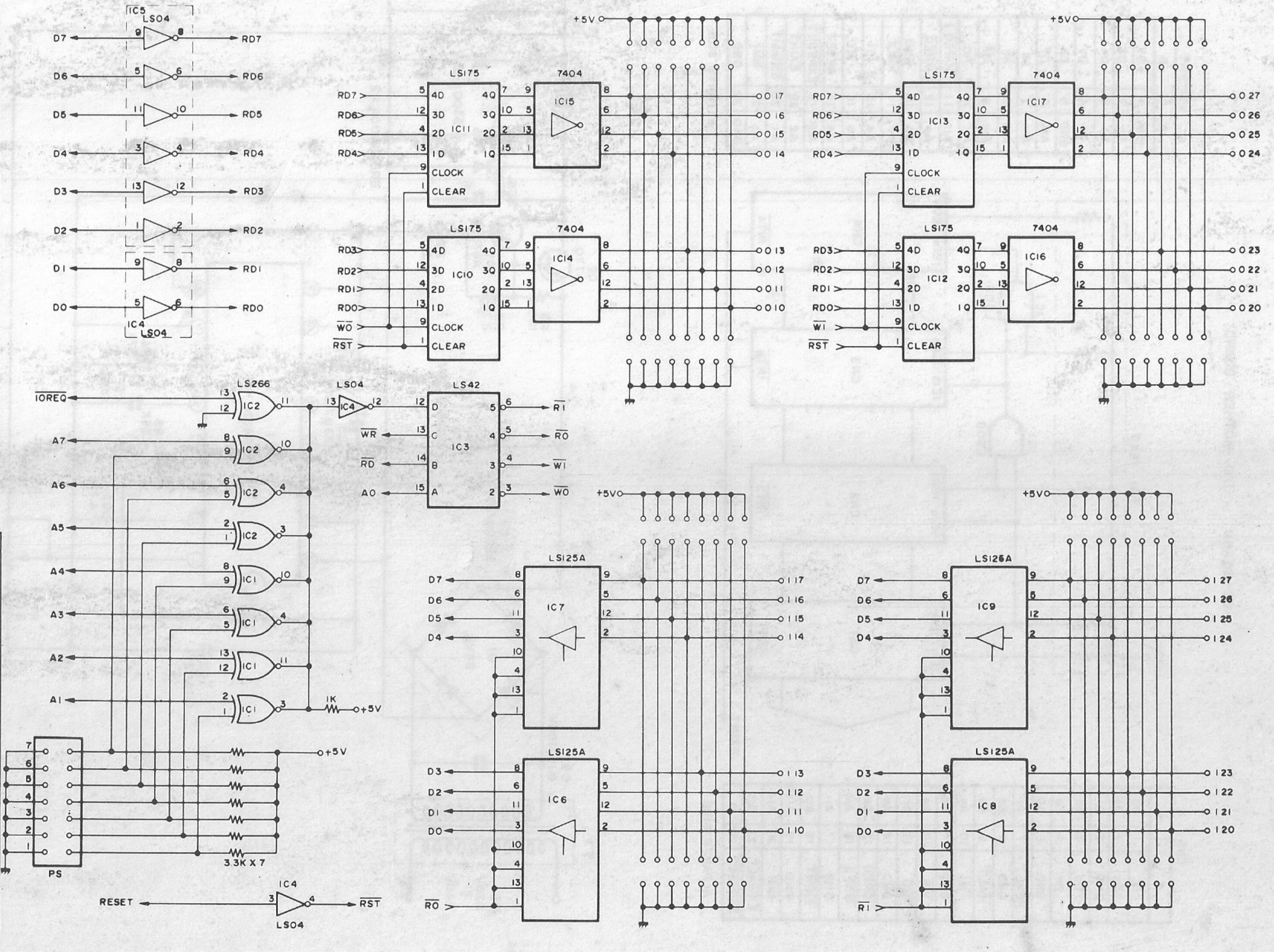
I/O CONNECTOR

A		B
+5V	1	+5V
D2	2	D3
D1	3	D4
D0	4	D5
GND	5	D6
A15	6	D7
A14	7	BUSϕ
A13	8	$\overline{M1}$
A12	9	$\overline{WR}$
A11	10	$\overline{RD}$
A10	11	$\overline{IOREQ}$
A9	12	$\overline{MREQ}$
A8	13	GND
A7	14	$\overline{HALT}$
A6	15	IEI
A5	16	IEO
A4	17	RESET
A3	18	$\overline{EXRESET}$
A2	19	$\overline{EXINT}$
A1	20	$\overline{EX WAIT}$
A0	21	$\overline{NMI}$
GND	22	GND

A : PARTS SIDE

SIGNAL TERMINAL

GND	1	20	GND
O10	2	21	O11
O12	3	22	O13
O14	4	23	O15
O16	5	24	O17
GND	6	25	I10
I11	7	26	I12
I13	8	27	I14
I15	9	28	I16
I17	10	29	GND
O20	11	30	O21
O22	12	31	O23
O24	13	32	O25
O26	14	33	O27
GND	15	34	I20
I21	16	35	I22
I23	17	36	I24
I25	18	37	I26
I27	19		



UNIVERSAL I/O CARD