THE UNIVERSITY OF MICHIGAN

COMPUTING RESEARCH LABORATORY[1]

USER MANUAL FOR ZIP, A Z80 ASSEMBLY LANGUAGE INTERPRETER PROGRAM

G. D. Buzzard
T. N. Mudge

CRL-TR-20-84

MARCH 1984

Room 1079, East Engineering Building
Ann Arbor, Michigan 48109
USA
Tel: (313) 763-8000

[1]Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the funding agencies.

This report is a reissue of Systems Engineering Laboratory (now defunct) report: SEL-TR-154.

engn
VHR1092

SEL-TR-154

User Manual for ZIP, a Z80 Assembly Language Interpreter Program

G. D. Buzzard
T.N. Mudge
The University of Michigan
Department of Electrical and
Computer Engineering
Ann Arbor, Michigan 48109
(313) 764-0203

August 1981

Prepared under
a Grant from the University of Michigan
Center for Research on Learning and Teaching

TABLE OF CONTENTS

Introduction

ZIP (Z80 Interpreter Program) was written by G. D. Buzzard and W. MacLeod based on a design by T. N. Mudge. Designed as a teaching/debugging tool, ZIP is presently used to aid in the teaching of an introductory microcomputer course. Following the User Manual is an assembly listing of ZIP.

ZIP disassembles and interprets segments of memory containing Z80 machine code and/or data. The disassembled code is displayed on the left side of the CRT screen and the CPU registers, top four words of the stack, and 32 contiguous bytes of memory are displayed on the right side. This configuration provides a visual relationship between the assembly language source code and the dynamic state of the CPU registers, stack, and memory locations. Commands can be entered to control the interpretation of the program, modify register or memory values, and to control the display of information.

Program Overview

In order to display the disassembled code in an intelligible format the areas of executable code must be distinguished from data areas and unused memory. This is accomplished by searching the target program code for jump, call, and return statements. The location of these statements, along with their targets, when necessary, are used to develop a table of origin and end point (ORG-END) pairs. This table of ORG-END pairs is then used to determine which segments of memory will be disassembled and displayed as source code, and which will be displayed as data.

Upon completion of the ORG-END table the user is prompted for commands. After the execution of any command which affects the target program PC (program counter), a segment of memory is disassembled and displayed. Within the constraints mentioned later, this dynamic disassembly allows the effects of self-modifying code to be observed. Following the execution of commands which change the state of any of the CPU registers, displayed memory locations, or the stack, the right side of the display is updated.

The user is prompted for new commands until program control by ZIP is terminated by:

1) The user entering the quit command (QU).
2) ZIP interpreting a reset, jump to the operating system, or an unreturned call to the operating system.
3) A recognized (interrupts enabled) mode 0 interrupt which supplies an instruction which changes the PC.
4) A recognized mode 1 interrupt.

Operating Instructions

The file containing the load module for ZIP is available on the ECE 365 system diskette. ZIP is loaded in memory at $C800_{16}$. The target program stack is initialized at $C7FF_{16}$ and proceeds towards low memory. Since no address checking is performed on the target program, the user is cautioned against interpreting programs which occupy or modify memory locations near or above $C800_{16}$.

The suggested method for running ZIP is to load both ZIP and the target program from SPDS (i.e., !A R,D,MYPGM<cr> !R<cr> !A R,D,ZIP<cr> !R<cr>), then issue the GO command for address $C800_{16}$ (!G 0C800) to begin program execution. From this point on, program flow is controlled by ZIP and the user's interactive commands. Immediately, the CRT screen is reformatted and the user is prompted for a "START ADDRESS?". The starting address of the target program code is to be entered as a four digit hexadecimal number followed by a carriage return. This enables ZIP to track the target program's origins and endpoints. The disassembled text, CPU register contents, top four stack words, and 32 memory locations are then displayed and the user prompted for a command.

Screen Layout

The layout of the screen is shown in Figure 1. The column on the left shows memory locations in hex (4 hex digits). Alongside these are one to four byte instruction codes also in hex (Z80 instructions can be from one to four bytes in length). Further to the right, the instruction codes are shown in their disassembled form. For example, consider the line covered by the shaded rectangle in the left center. At the left is a memory location ($90EE_{16}$). This location and the subsequent one contain the bytes 10_{16} and $F7_{16}$ respectively (the Z80 has byte oriented addressing). These disassemble to the Z80 instruction: "DJNZ 90E7" -- decrement register B and jump if B is non- zero to location $90E7_{16}$. Notice that addresses of operands or targets of jumps are not disassembled but are left as absolute addresses. To disassemble these would require access to the symbol table created when the program was assembled. In order to keep the first version of ZIP simple the ability to recover symbolic address was omitted.

As noted above, ZIP automatically determines data areas in memory by examining jumps, subroutine calls, and returns, and when necessary their targets. Memory locations that contain data rather than instructions have their contents displayed as one to four hex digits in the same column as the symbolic instruction codes. Further to the right, in the same column as the disassembled instruction, the contents of the memory is displayed in its ASCII character form.

The right hand side of the screen displays the contents of the Z80's CPU registers, the top four items on the stack, 32 bytes of memory, and the command line.

There are eight 1 byte CPU registers: A, F, B, C, D, E, H, L. These are displayed at the top right of the screen. For example, the second row at the top right shows the contents of A in hex (88), the contents of B in hex (33), followed by the contents of A in binary (10001000), and the contents of F in binary (00110011). The binary display is useful for checking bit operations, shifts, and rotations. The F register is not a general purpose register, instead it holds six 1 bit flags that show the condition codes. Their position is shown in the binary display of F by the header "SZ*H*PNC" at the extreme top right (see [2] for their meaning). Immediately below the 1 byte CPU registers are the 16 bit CPU registers: IX, IY, SP, PC. IX and IY are index registers, SP is the stack pointer (points to the top of the stack), and PC is the program counter. The register pairs BC, DE, and HL can also be regarded as 16 bit registers and the format of the display has been set up to allow this view. To the right of the 16 bit registers appears the two special 1 byte registers I and R. Below the 16 bit registers appears the top four stack items. These items are one word, or two bytes each, thus in Figure 1, for example the top of the stack is at location $F3F8_{16}$ (see contents of SP)

and the top item is the 16 bit quantity $ED08_{16}$. The bytes of the top four words of the stack are shown in the reverse order from which they appear in memory; left-to-right within each word corresponds to high-to-low memory addresses. The stack grows towards low memory. The orientation of the bytes displayed in each word is consistent with the orientation of the bytes displayed in the 16 bit registers and the register pairs BC, DE, and HL. In all cases, 16 bit words are stored with their most significant byte at the higher memory location.

Below the stack display a user selected 32 byte area of memory is displayed in hex. Finally, below that the command currently being entered by the user is shown.

The shaded rectangles in Figure 1 indicate reverse video. Thus the instruction to be executed next is the DJNZ mentioned above. In addition the contents of the H and L registers are shown in reverse video indicating that the most recently executed instruction -- "INC HL" -- caused their contents to be altered. If any of the memory locations already displayed on the screen had been altered they would also be shown in reverse video.

The command line is shown with an reverse video square alongside it to distinguish it. The particular command shown in figure 1 reads: beginning with the current instruction (the DJNZ) execute the program until the contents of registers A and B have been equal three times. The command is terminated with a carriage return; the return initiates ZIP's interpretation of the command line. The left side of the display scrolls so that the next instruction to be interpreted (i.e., the instruction displayed in reverse video) is always kept in the middle of the screen.

Command Syntax

Figure 1 shows ZIP's command syntax in standard BNF (Backus Naur Form) notation. The current version of ZIP enforces rather severe spacing restrictions:

1) Exactly one space is required between the command words GO, SE, DI and the productions which follow them. The command words which are not followed by any productions may be followed by any combination of other characters.
2) Repetition factors (i.e., :12) must be preceded by one or more spaces.
3) No spaces other than those mentioned above are allowed.
4) All commands must be terminated by carriage returns.

   Note: only the first two letters of the command word are interpreted. Therefore, GO_BLUE 5000 is interpreted the same as GO 5000.

The modification to ZIP to accommodate arbitrary spacing is straight forward and will be implemented at a later date.

Command Descriptions

GO --

causes the contents of the target program PC to be replaced by the specified value. The disassembled code is updated to reflect this change, and all reverse video on the right side of the screen, with the exception of

the PC, unless it is left unchanged, is reset to normal video.

SET --
causes the contents of the indicated register or memory location to be replaced by the specified value. Unless the specified value equals the previous value, the indicated register or memory location will be displayed in reverse video. It is possible to change the contents of any memory location regardless of whether or not it is displayed.

DISPLAY --
displays 32 contiguous bytes of memory, beginning with the specified address, in the memory display area of the screen. All resulting memory display screen locations with values differing from their previous ones are displayed in reverse video, while those screen locations remaining unchanged are displayed in normal video. This feature facilitates a quick byte by byte comparison of different memory locations.

Trigger Conditions --
cause the target program to be interpreted until the specified condition is met. All displayed values which have changed since the last command are shown in reverse video, while those which have not changed are shown in normal video. The <tail> production specifies the number of target program instructions to be interpreted, if the number is omitted a default of one is assumed. The second alternative of the <taill> production specifies an optional repetition factor. And, the <memory><relation><rhs_memory> construct, when used, operates on comparisons of 8 bits in length.

ON --
fills the screen with 256 contiguous bytes of memory beginning with the 32 bytes which were displayed in the normal screen format.

OFF --
is used in conjunction with ON. OFF returns the screen to its normal format. All reverse video which appeared in the memory display area of the screen previous to the ON command is reset to normal video.

AU --
swaps the alternate A and F registers for the present ones. This command has a toggling action, and may be repeated several times in succession. Any resulting change (with respect to the values which were displayed at the end of the preceding command) is shown in reverse video.

UA --
performs the same function as AU on the remaining CPU general registers (i.e., B,C,D,E,H,L).

OLD --
displays the right side of the screen exactly as it appeared prior to the last trigger condition command.

NEW --
is used in conjunction with OLD. NEW restores the right side of the screen to reflect the current state of the registers and memory. All reverse video which appeared prior to the OLD command is reset to normal video.

QUIT --
terminates execution of ZIP and returns control to SPDS.

System Routines

Operating system routines are not interpreted. Therefore, the execution of all operating system routines appear transparent to the user. The single step interpretations of some common operating system call statements are described below:

CALL CO (console output) --
The ASCII character representing the contents of the C register is flashed briefly near the lower left corner of the screen, and the user is prompted for the next command.

CALL CICO (console input console output) --
Program execution enters a wait loop until an input from the keyboard is received. The character entered is flashed briefly near the lower left corner of the screen, and the corresponding ASCII value is displayed in the A register.

CALL CI (console input) --
Program execution enters a wait loop until an input from the keyboard is received. Upon receiving an input the ASCII value corresponding to the entry is displayed in the A register.

Self Modifying Code

The target program is tracked only once, establishing the ORG-END table prior to any part of the target program being executed. Hence, if the target program dynamically modifies a memory area which was executable code into a data field, or vice versa, the display on the left side of the screen will periodically become unintelligible. This, however, should not affect the correct execution of the target program. Self modifying code which does not change executable code into data, or vice versa, is interpreted without any adverse affects.

Computed Jumps

Another result of the program being tracked only once is that the run-time targets of computed jumps (i.e., JP(HL), JP(IX), and JP(IY)) cannot be determined. When the tracking routine encounters a computed jump the tracking is aborted and the user queried to provide the entries for the ORG-END table.

The format required for user entry of the ORG-END table is as follows:

1) All entries must appear as four digit hex numbers followed by a carriage return.
2) All entries must be entered as ORG-END pairs, with END's being entered immediately after their corresponding ORG's.
3) The entries must be entered in ascending numerical order of ORG values.
4) The last two entries must be FFFF and 0000 respectively.

Interrupts

Mode 0 interrupts (see [2] for descriptions) cannot be detected by ZIP. But, provided that the instruction supplied by the peripheral device does not change the PC, any resulting changes in the CPU registers, top for stacks words, or displayed memory will be shown.

As stated earlier, mode 1 interrupts result in program control being returned to the operating system, This action is effected via the equivalent of a "RST 38H" instruction.

For mode 2 interrupts the current version of ZIP will not dynamically trace the execution of an interrupt service routine during the interpretation of the target program. The final CPU register status, top four stack values, and displayed memory values will be shown, but, the execution of the interrupt service routine will appear transparent. However, ZIP can be coerced to request the user to make entries to the ORG-END table -- SPDS users can do this by entering C800 for the "START ADDRESS?". Then, ORG-END pairs can be entered which encompass only the interrupt service routine, thus allowing the routine to be interpreted as a separate entity. This independent interpretation of the interrupt service routine is analogous to the testing of an external subroutine of a structured computer program before the main (calling) program is tested as a whole.

By interpreting the target program up the the point where the interrupt would occur the pertinent register and memory values can be obtained. These values can then be loaded into the appropriate places at the beginning of the interpretation of the interrupt service routine by using the SET command.

Refresh Register

The R (refresh) register exhibits some unique characteristics during program interpretations by ZIP. While the R register display does indeed reflect the actual value of the CPU R-register at the beginning of an instruction simulation, the R-register display may not reflect the actual value of the CPU R-register after any occurrence of the following simulation events:

1) Calls to the operating system.
2) Interrupts which are handled during instruction simulation.

These discrepancies are possible because the CPU R-register display value is computed by ZIP, and not merely taken from the CPU R-register. This is done in an effort to closely approximate the decrementing of the R-register during the execution of the target program alone, without reflecting the refresh cycles which occur during the execution of ZIP program code. However, the number of memory refresh cycles cannot be computed for operating system subroutines, interrupt service routines, or mode 0 interrupt instructions because their execution is not dynamically traced.

It is important to remember that the CPU R-register, which is initially set to 00_{16}, is coerced by ZIP and does contain the value indicated in the display during the execution of the instruction which is shown in reverse video on the screen.

References

[1] T. N. Mudge. "Teaching Assembly Language Using an Assembly Language Interpreter." Proc. 1981 ASEE Annual Conference, Univ. So. Cal., June 1981.

[2] Microcomputer Data Book, Mostek, 1979.

Illustrations

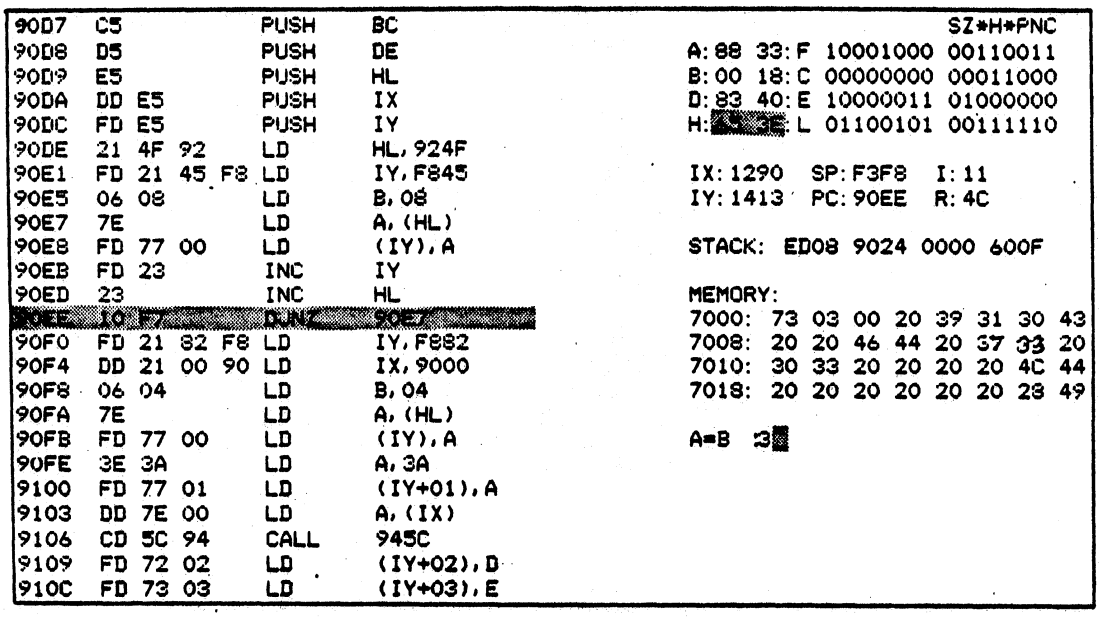

Figure 1. Screen layout

<command>::= GO<hex><hex><hex><hex>|<set>|<display>|
<trigger_condition>|ON|OF|AU|UA|OL|NE|QU

<set>::= SE<reg>=<hex><hex>|SE<double_reg>=<hex><hex><hex><hex>|
SE<memory>=<hex><hex>

<display>::= DI<memory>
<trigger_condition>::= <condition><taill>|<tail>
<condition>::= <reg><relation><rhs_reg>|
double_reg><relation><rhs_double>|
<memory><relation><rhs_memory>|
F=<bit><bit><bit><bit><bit><bit>

<reg>::= A|B|C|D|E|H|L
<relation>::= =| =| <| >|<|>|
<rhs_reg>::= <reg>|<hex><hex>|<memory>
<double_reg>::= BC|DE|HL|SP|PC|IX|IY
<rhs_double>::= <double_reg>|<hex><hex><hex><hex>|<memory>
<rhs_memory>::= <hex><hex>|<memory>
<memory>::= @<hex><hex><hex><hex>
<hex>::= 0|1|2|3|4|...|F
<taill>::= c_r|:<number>c_r
<tail>::= <number>c_r|c_r
<number>::= <hex>|<hex><hex>|<hex><hex><hex>|<hex><hex><hex><hex>
<bit>::= 0|1|X

Figure 2. ZIP's Syntax.

Assembly Listing

The following pages contain a commented Z80 assembly listing for ZIP.

```
                00001 .COMMENT>
                00002                                   BY G. D. BUZZARD
                00003                                   APRIL 1981
                00004                                   LAST UPDATED 08-21-81
                00005                                   -
                00006       THIS IS A SET OF ROUTINES TO FACILITATE AND
                00007 DIRECT THE EXECUTION OF ZIP (Z-80 INTERPRETER
                00008 PROGRAM).
                00009 .)
0010            00010 .RADIX  16
                00011 ;
0041            00012 ASCA    EQU     41        ; ASCII A
0047            00013 ASCFP   EQU     47        ; ASCII F +1
0030            00014 ASC0    EQU     30        ; ASCII 0
003A            00015 ASC9P   EQU     3A        ; ASCII 9 +1
FD82            00016 COMLOC  EQU     0FD82     ; COMMAND LINE LOCATION
000D            00017 CR      EQU     0D        ; ASCII <CR>
0005            00018 CURSH   EQU     05        ; CURSOR HOME (HIGH)
0082            00019 CURSL   EQU     82        ; CURSOR HOME (LOW)
009B            00020 CURSYN  EQU     9B        ; CURSOR ERROR POSITION
000E            00021 KYBD    EQU     0E        ; KEYBOARD INPUT ADDRESS
0014            00022 POLL    EQU     14        ; KEYBOARD STATUS REGISTER
0000            00023 QCURSH  EQU     00        ; QUESTION POSITION
00FF            00024 QCURSL  EQU     0FF       ;     "         " (LOW)
F8FF            00025 QRESP   EQU     0F8FF     ; RESPONSE POSITION
F8F0            00026 Q1      EQU     0F8F0     ; QUESTION START ADDRESS
0123            00027 SYST    EQU     123       ; SYSTEM ADDRESS
F86C            00028 TITL    EQU     0F86C     ; ADDRESS FOR TITLE
C7FF            00029 USP     EQU     0C7FF     ; USER STACK LOCATION
00ED            00030 VIDEO   EQU     0ED       ; V-RAM IN CODE
0012            00031 VIDOFF  EQU     12        ; V-RAM OUT CODE
                00032 ;
                00033         EXTRN   ACONV
                00034         EXTRN   BANKST
                00035         EXTRN   BANKSW
                00036         EXTRN   CICO
                00037         EXTRN   CURSES
                00038         EXTRN   CURSOR
                00039         EXTRN   FINTOP
                00040         EXTRN   KEYIN
                00041         EXTRN   N31
                00042         EXTRN   MDISP
                00043         EXTRN   ORGEND
                00044         EXTRN   PRESAV
                00045         EXTRN   REGA
                00046         EXTRN   REGAF
                00047         EXTRN   REGB
                00048         EXTRN   REGC
                00049         EXTRN   REGD
                00050         EXTRN   REGE
                00051         EXTRN   REGF
                00052         EXTRN   REGH
                00053         EXTRN   REGL
                00054         EXTRN   REGIX
                00055         EXTRN   REGIY
                00056         EXTRN   REGSP
```

```
                    00057           EXTRN   REGPC
                    00058           EXTRN   RSTATE
                    00059           EXTRN   REVID
                    00060           EXTRN   REVMEM
                    00061           EXTRN   REGSAV
                    00062           EXTRN   SCREEN
                    00063           EXTRN   SAVIT
                    00064           EXTRN   SAVE
                    00065           EXTRN   SIMUL
                    00066           EXTRN   TEMP
                    00067           EXTRN   TEXTUP
                    00068           EXTRN   TRACK
                    00069           EXTRN   WIPE
                    00070           EXTRN   XRAF
                    00071           EXTRN   XRBC
                    00072 ;
                    00073           ENTRY   MAIN
                    00074           ENTRY   HEXIT
                    00075           ENTRY   SYN
                    00076 ;
0000' 31 0C4B'      00077 MAIN:     LD      SP,STKP         ; SET OUR STACK
0003' CD 0000*      00078           CALL    BANKST          ; SET BANKSWITCH REG
0006' CD 0146'      00079           CALL    INIT            ; INITIALIZE STORAGE
0009' 3E ED         00080           LD      A,VIDEO
000B' CD 0000*      00081           CALL    BANKSW          ; SWITCH IN V-RAM
000E' CD 0035'      00082           CALL    QUERY           ; GET START ADDRESS
0011' 2A 00E2'      00083           LD      HL,(STRT)       ; LOAD IT INTO HL
0014' CD 0000*      00084           CALL    TRACK           ; TRACK TARGET PROGRAM
0017' 2A 0000*      00085           LD      HL,(ORGEND)     ; ABSOLUTE START ADDR
001A' 22 0000*      00086           LD      (REGPC),HL      ; AND INTO REGPC
001D' CD 0000*      00087           CALL    SCREEN
0020' 3E 9B         00088           LD      A,CURSYN
0022' 32 0125'      00089           LD      (CURERR),A      ; ERROR CURSOR POSITION
0025' 3E ED         00090           LD      A,VIDEO
0027' CD 0000*      00091           CALL    BANKSW          ; SWITCH IT IN AGAIN
002A' 21 FD82       00092 RPT:      LD      HL,COMLOC
002D' CD 0000*      00093           CALL    CICO            ; GET INPUT
0030' CD 0189'      00094           CALL    DECIDE
0033' 18 F5         00095           JR      RPT
                    00096 ;
                    00097 ; QUERY QUERIES THE USER FOR THE STARTING ADDRESS
                    00098 ; OF THE PROGRAM CODE.  ONLY THE FIRST FOUR CHARACTERS
                    00099 ; OF THE RESPONSE ARE USED (45678=>4567).
                    00100 ,
0035' F5            00101 QUERY:    PUSH    AF
0036' C5            00102           PUSH    BC
0037' D5            00103           PUSH    DE
0038' E5            00104           PUSH    HL
                    00105 ;
0039' CD 0000*      00106           CALL    WIPE            ; CLEAR SCREEN
003C' 11 F86C       00107           LD      DE,TITL
003F' 21 00FE'      00108           LD      HL,ZIP
0042' 01 0018       00109           LD      BC,18
0045' ED B0         00110           LDIR                    ; WRITE TITLE
                    00111 ;
0047' 11 F8F0       00112           LD      DE,Q1
```

```
004A' 01 000F    00113          LD      BC,0F
004D' ED B0      00114          LDIR                    ; WRITE "START ADDR"
                 00115 ;
004F' 21 0000*   00116          LD      HL,CURSES
0052' 23         00117          INC     HL              ; CHANGE
0053' 3E 00      00118          LD      A,QCURSH        ;
0055' 77         00119          LD      (HL),A          ; CURSOR
0056' 23         00120          INC     HL              ;
0057' 23         00121          INC     HL              ; ADDRESS
0058' 3E FF      00122          LD      A,QCURSL
005A' 77         00123          LD      (HL),A
                 00124 ;
005B' CD 0000*   00125          CALL    CURSOR
005E' EB         00126          EX      DE,HL           ; PUT SCR ADDR IN HL
005F' CD 0000*   00127          CALL    CICO
                 00128 ;
                 00129 ; GET INPUT, CONVERT AND CHECK SYNTAX
0062' 21 00E3'   00130 L22:     LD      HL,STRT+1       ; LOCN FOR START ADDR
0065' 06 02      00131          LD      B,2             ; COUNTER
0067' 11 0000*   00132          LD      DE,KEYIN        ; ASCII LOCN
                 00133 ;
006A' CD 0126'   00134 L21:     CALL    HEXIT           ; CONVERT TO HEX
006D' CB 7F      00135          BIT     7,A             ; ERROR?
006F' 20 37      00136          JR      NZ,SYN          ; IF SO JUMP
                 00137 ;
0071' ED 6F      00138          RLD                     ; STORE IT
0073' 13         00139          INC     DE              ; NEXT ASCII
                 00140 ;
0074' CD 0126'   00141          CALL    HEXIT           ; CONVERT IT
0077' CB 7F      00142          BIT     7,A             ; ERROR?
0079' 20 2D      00143          JR      NZ,SYN          ; IF SO JUMP
                 00144 ;
007B' ED 6F      00145          RLD                     ; STORE IT
007D' 13         00146          INC     DE              ; NEXT ASCII
007E' 2B         00147          DEC     HL
007F' 10 E9      00148          DJNZ    L21
                 00149 ;
0081' 1A         00150          LD      A,(DE)
0082' FE 0D      00151          CP      CR              ; <CR> ?
0084' C2 00A8'   00152          JP      NZ,SYN
                 00153 ;
                 00154 ; MOVE CURSOR
0087' 21 0000*   00155          LD      HL,CURSES
008A' 23         00156          INC     HL              ; CHANGE
008B' 3E 05      00157          LD      A,CURSH ;
008D' 77         00158          LD      (HL),A          ; CURSOR
008E' 23         00159          INC     HL              ;
008F' 23         00160          INC     HL              ; ADDRESS
0090' 3E 82      00161          LD      A,CURSL
0092' 77         00162          LD      (HL),A
                 00163 ;
0093' E1         00164          POP     HL
0094' D1         00165          POP     DE
0095' C1         00166          POP     BC
0096' F1         00167          POP     AF
0097' C9         00168          RET
```

```
                    00169 ;
                    00170 ; SYNTAX ERROR HANDLER
0099' DD 21         00171 SYN1:   LD      IX,DECIDE+4     ; RETURN PAST PUSHES
009A' 018D'
009C' 21 00E4'      00172         LD      HL,MESS2
009F' 11 FD82       00173         LD      DE,COMLOC
00A2' FD 21         00174         LD      IY,COMLOC
00A4' FD82
00A6' 18 0E         00175         JR      L23
                    00176 ;
00A8' DD 21         00177 SYN:    LD      IX,L22          ; RETURN ADDRESS
00AA' 0062'
00AC' 21 00E4'      00178         LD      HL,MESS2
00AF' 11 F8FF       00179         LD      DE,QRESP
00B2' FD 21         00180         LD      IY,QRESP
00B4' F8FF
00B6' 01 0019       00181 L23:    LD      BC,19
00B9' ED B0         00182         LDIR                    ; WRITE MESSAGE
                    00183 ;
00BB' 21 0000*      00184         LD      HL,CURSES
00BE' 23            00185         INC     HL
                    00186 ;
00BF' 7E            00187         LD      A,(HL)          ; CHECK FOR SPECIAL
00C0' FE 00         00188         CP      00              ; CASE -- <CR> FOR
00C2' 20 03         00189         JR      NZ,L24          ; START ADDRESS --
00C4' 3E 01         00190         LD      A,01
00C6' 77            00191         LD      (HL),A
                    00192 ;
00C7' 23            00193 L24:    INC     HL              ; THE LINKER DOES
00C8' 23            00194         INC     HL              ; NOT ALLOW ARITH-
00C9' 3A 0125'      00195         LD      A,(CURERR)      ; METICS ON EXTERNALS
00CC' 77            00196         LD      (HL),A
00CD' CD 0000*      00197         CALL    CURSOR          ; CURSOR
                    00198 ;
00D0' DD E5         00199         PUSH    IX              ; PUT IT ON STACK
                    00200 ;
00D2' FD E5         00201         PUSH    IY
00D4' E1            00202         POP     HL
00D5' F5            00203         PUSH    AF              ; WE'RE SKIPPING
00D6' C5            00204         PUSH    BC              ; THE FORMAL
00D7' D5            00205         PUSH    DE              ; SUBROUTINE
00D8' E5            00206         PUSH    HL              ; ENTRY POINT
00D9' DD E5         00207         PUSH    IX
00DB' DD 21         00208         LD      IX,KEYIN        ; RESET POINTER
00DD' 0000*
00DF' C3 0000*      00209         JP      N31             ; JUMP TO SUBR
                    00210                                 ; IN CICO
                    00211 ;
00E2'               00212 STRT:   DEFS    2
00E4' 53 59 4E      00213 MESS2:  DEFB    'SYNTAX ERROR RE-ENTER
00E7' 54 41 58
00EA' 20 45 52
00ED' 52 4F 52
00F0' 20 52 45
00F3' 2D 45 4E
00F6' 54 45 52
```

```
00F9' 20 20 20
00FC' 20 20
00FE' 5A 2D 38  00214 ZIP:    DEFB    'Z-80 INTERPRETER PROGRAM'
0101' 30 20 49
0104' 4E 54 45
0107' 52 50 52
010A' 45 54 45
010D' 52 20 50
0110' 52 4F 47
0113' 52 41 4D
0116' 53 54 41  00215         DEFB    'START ADDRESS? '
0119' 52 54 20
011C' 41 44 44
011F' 52 45 53
0122' 53 3F 20
0125' 18        00216 CURERR: DEFB    18
                00217 ;
                00218 ; HEXIT TAKES THE ASCII CONTENTS OF DE AND CONVERTS
                00219 ; IT TO HEX.  THE HEX VALUE IS STORED IN THE A-REG.
                00220 ; IF A NON-NUMERIC ENTRY IS DETECTED, THE A-REG
                00221 ; RETURNS OFF.
                00222 ;
0126' 1A        00223 HEXIT.  LD      A,(DE)          ; ASCII
0127' FE 30     00224         CP      ASC0            ; <0?
0129' FA 0143'  00225         JP      M,ERR           ; IF SO ERROR
012C' FE 3A     00226         CP      ASC9P           ; < 9?
012E' F2 0134'  00227         JP      P,NZ1           ; IF SO JUMP
                00228 ;
0131' E6 0F     00229         AND     0F              ; CONVERT TO HEX
0133' C9        00230         RET
                00231 ;
0134' FE 41     00232 NZ1:    CP      ASCA            ; <A?
0136' FA 0143'  00233         JP      M,ERR           ; IF SO ERROR
0139' FE 47     00234         CP      ASCFP           ; <F?
013B' F2 0143'  00235         JP      P,ERR           ; IF NOT ERROR
                00236 ;
013E' E6 0F     00237         AND     0F              ; CONVERT TO HEX
0140' C6 09     00238         ADD     A,9
0142' C9        00239         RET
                00240 ;
0143' 3E FF     00241 ERR:    LD      A,0FF           ; ERROR CODE
0145' C9        00242         RET
                00243 ;
                00244 ; INIT INITIALIZES THE STORAGE AREAS PRESAV,
                00245 ; REGSAV, AND MDISP WITH ZEROES.  ALSO THE USER
                00246 ; STACK LOCATION IS SET.
                00247 ;
0146' 01 0032   00248 INIT:   LD      BC,32           ; LENGTH
0149' 11 0000*  00249         LD      DE,PRESAV       ; DESTINATION
014C' 21 0176'  00250 L1:     LD      HL,ZERO         ; SOURCE
014F' ED A0     00251         LDI
0151' 2B        00252         DEC     HL
0152' EA 014C'  00253         JP      PE,L1           ; IF NOT DONE LOOP
                00254 ;
0155' 21 C7FF   00255         LD      HL,USP          ; USER STACK LOCATION
0158' 22 0000*  00256         LD      (REGSP),HL
```

```
015B' 21 0000    00257          LD      HL,00
015E' 22 0000*   00258          LD      (MDISP),HL      ; MEMORY DISP LOC'N
                 00259 ;
0161' 01 0050    00260          LD      BC,50           ; LENGTH
0164' 11 0000*   00261          LD      DE,SAVIT        ; DESTINATION
0167' 21 0176'   00262 L2:      LD      HL,ZERO         ; SOURCE
016A' ED A0      00263          LDI
016C' 2B         00264          DEC     HL
016D' EA 0167'   00265          JP      PE,L2           ; IF NOT DONE LOOP
                 00266 ;
0170' 3E 18      00267          LD      A,18            ; INIT CURERR
0172' 32 0125'   00268          LD      (CURERR),A
0175' C9'        00269          RET
                 00270 ;
0176' 00         00271 ZERO:    DEFB    00
                 00272 ;
                 00273 ; COPY COPIES REGSAV INTO PRESAV
                 00274 ;
0177' C5         00275 COPY:    PUSH    BC
0178' D5         00276          PUSH    DE
0179' E5         00277          PUSH    HL
017A' 01 0012    00278          LD      BC,12
017D' 11 0000*   00279          LD      DE,PRESAV
0180' 21 0000*   00280          LD      HL,REGSAV
0183' ED B0      00281          LDIR
0185' E1         00282          POP     HL
0186' D1         00283          POP     DE
0187' C1         00284          POP     BC
0188' C9         00285          RET
                 00286 ;
                 00287 ; DECIDE IS THE COMMAND INTERPRETER.  IT'S INPUT
                 00288 ; IS TAKEN FROM KEYIN.  THE CHARACTERS IN KEYIN ARE
                 00289 ; PARSED TO DETERMINE THE ACTION TO BE TAKEN.
                 00290 ; THIS ROUTINE IS STRUCTURED AS A BINARY DECISION
                 00291 ; TREE.
                 00292 ;
0189' F5         00293 DECIDE:  PUSH    AF
018A' C5         00294          PUSH    BC
018B' D5         00295          PUSH    DE
018C' E5         00296          PUSH    HL
                 00297 ;
018D' 21 0000*   00298          LD      HL,KEYIN        ; START OF BUFFER
0190' 7E         00299          LD      A,(HL)
0191' FE 4F      00300          CP      4F              ; ASCII O
0193' FA 01E1'   00301          JP      M,DQ            ; < O
0196' 20 1A      00302          JR      NZ,SQ           ; > O
                 00303 ;
0198' 23         00304          INC     HL              ; NEXT CHARACTER
0199' 7E         00305          LD      A,(HL)
019A' FE 4C      00306          CP      4C              ; ASCII L
019C' FA 01AA'   00307          JP      M,OFQ           ; < OL
019F' CA 0A22'   00308          JP      Z,OL            ; = OL
                 00309 ;
01A2' FE 4E      00310          CP      4E              ; ACSII N (ON?)
01A4' CA 09B7'   00311          JP      Z,ON            ; = ON
01A7' C3 0098'   00312          JP      SYN1            ; ERROR
```

```
                        00313 ;
01AA' FE 46             00314 OFQ:      CP      46              ; ASCII F (OF?)
01AC' CA 0A04'          00315           JP      Z,OF            ; = OF
01AF' C3 0098'          00316           JP      SYN1            ; ERROR
                        00317 ;
01B2' FE 53             00318 SQ:       CP      53              ; ASCII S
01B4' FA 01D2'          00319           JP      M,QQ            ; < S
01B7' 20 0A             00320           JR      NZ,UQ           ; > S
                        00321 ;
01B9' 23                00322           INC     HL              ; NEXT CHARACTER
01BA' 7E                00323           LD      A,(HL)
01BB' FE 45             00324           CP      45              ; ASCII E (SE?)
01BD' CA 02C8'          00325           JP      Z,SE            ; = SE
01C0' C3 0435'          00326           JP      TRG             ; ERROR OR TRIGGER
                        00327 ;
01C3' FE 55             00328 UQ:       CP      55              ; ASCII U
01C5' C2 0098'          00329           JP      NZ,SYN1         ; ^= U ERROR
01C8' 23                00330           INC     HL              ; NEXT CHARACTER
01C9' 7E                00331           LD      A,(HL)
01CA' FE 41             00332           CP      41              ; ASCII A
01CC' CA 098C'          00333           JP      Z,UA            ; = UA
01CF' C3 0098'          00334           JP      SYN1            ; ERROR
                        00335 ;
01D2' FE 51             00336 QQ:       CP      51              ; ASCII Q
01D4' C2 0435'          00337           JP      NZ,TRG          ; ^= Q ERROR OR TRIGGER
01D7' 23                00338           INC     HL              ; NEXT CHARACTER
01D8' 7E                00339           LD      A,(HL)
01D9' FE 55             00340           CP      55              ; ACSII U (QU?)
01DB' CA 02C0'          00341           JP      Z,QU            ; = QU
01DE' C3 0098'          00342           JP      SYN1            ; ERROR
                        00343 ;
01E1' FE 44             00344 DQ:       CP      44              ; ASCII D
01E3' FA 0213'          00345           JP      M,AQ            ; < D
01E6' 20 0A             00346           JR      NZ,NQ           ; > D
                        00347 ;
01E8' 23                00348           INC     HL              ; NEXT CHARACTER
01E9' 7E                00349           LD      A,(HL)
01EA' FE 49             00350           CP      49              ; ASCII I (DI?)
01EC' CA 0222'          00351           JP      Z,DI            ; = DI
01EF' C3 0435'          00352           JP      TRG             ; TRIGGER OR ERROR
                        00353 ;
01F2' FE 4E             00354 NQ:       CP      4E              ; ASCII N
01F4' FA 0204'          00355           JP      M,GQ            ; < N
01F7' C2 0098'          00356           JP      NZ,SYN1         ; > N ERROR
                        00357 ;
01FA' 23                00358           INC     HL              ; NEXT CHARACTER
01FB' 7E                00359           LD      A,(HL)
01FC' FE 45             00360           CP      45              ; ASCII E (NE?)
01FE' CA 0A10'          00361           JP      Z,NEE           ; = NE
0201' C3 0098'          00362           JP      SYN1            ; ERROR
                        00363 ;
0204' FE 47             00364 GQ:       CP      47              ; ASCII G
0206' C2 0435'          00365           JP      NZ,TRG          ; TRIGGER OR ERROR
0209' 23                00366           INC     HL
020A' 7E                00367           LD      A,(HL)          ; NEXT CHARACTER
020B' FE 4F             00368           CP      4F              ; ASCII O (GO?)
```

```
020D' CA 028F'  00369          JP      Z,GO            ; = GO
0210' C3 0098'  00370          JP      SYN1            ; ERROR
                00371 ;
0213' FE 41     00372 A@:      CP      41              ; ASCII A
0215' C2 0435'  00373          JP      NZ,TRG          ; TRIGGER OR ERROR
0218' 23        00374          INC     HL              ; NEXT CHATACTER
0219' 7E        00375          LD      A,(HL)
021A' FE 55     00376          CP      55              ; ASCII U (AU?)
021C' CA 096F'  00377          JP      Z,AU            ; = AU
021F' C3 0435'  00378          JP      TRG             ; TRIGGER OR ERROR
                00379 ;
                00380 ; DI HANDLES THE MEMORY DISPLAY INSTRUCTION.
                00381 ; ONLY THE FIRST FOUR NUMERICS ARE USED FOR
                00382 ; DETERMINING THE ADDRESS OF THE MEMORY TO BE
                00383 ; DISPLAYED.
                00384 ;
0222' 01 0012   00385 DI:      LD      BC,12
0225' 3E 20     00386          LD      A,20            ; ASCII <SP>
0227' 23        00387          INC     HL              ; NEXT CHARACTER
0228' ED B1     00388          CPIR                    ; SEARCH FOR BLANK
                00389 ;
022A' E2 0098'  00390          JP      PO,SYN1         ; BLANK NOT FOUND
022D' 7E        00391          LD      A,(HL)          ; CHAR AFTER <SP>
022E' FE 40     00392          CP      40              ; ASCII @
0230' C2 0098'  00393          JP      NZ,SYN1
0233' 23        00394          INC     HL              ; SHOULD POINT TO HHHH
0234' 11 0000*  00395          LD      DE,MDISP        ; TARGET LOCATION
0237' EB        00396          EX      DE,HL
0238' 23        00397          INC     HL
                00398 ;
0239' CD 025B'  00399          CALL    HEX4
                00400 ;
023C' 1A        00401          LD      A,(DE)          ; NEXT CHARACTER
023D' FE 0D     00402          CP      CR              ; IS IT <CR> ?
023F' C2 0098'  00403          JP      NZ,SYN1
                00404 ;
0242' CD 0000*  00405          CALL    RSTATE          ; PUT UP RHS OF SCREEN
0245' CD 027F'  00406          CALL    BLNK
0248' CD 0000*  00407          CALL    CURSOR
024B' CD 0000*  00408          CALL    REVID
024E' CD 0000*  00409          CALL    REVMEM
0251' CD 0000*  00410          CALL    SAVE
0254' C3 0AB8'  00411          JP      DONE
                00412 ;
                00413 ; HEX4 TAKES THE FOUR ASCII BYTES POINTED BY DE,
                00414 ; CONVERTS THEM TO HEX, AND STORES THEM IN THE
                00415 ; TWO BYTES POINTED BY HL.
                00416 ; HEX2 FUNCTIONS SIMILARLY FOR TWO ASCII BYTES.
                00417 ;
0257' 06 01     00418 HEX2:    LD      B,1
0259' 18 02     00419          JR      AGAIN
                00420 ;
025B' 06 02     00421 HEX4:    LD      B,2
025D' CD 0126'  00422 AGAIN:   CALL    HEXIT           ; CONVERT TO HEX
0260' CB 7F     00423          BIT     7,A
0262' 28 05     00424          JR      Z,H1            ; IF NO ERROR JUMP
```

```
0264' DD E1      00425          POP     IX              ; STRIP TOP OF STACK
0266' C3 0098'   00426          JP      SYN1
                 00427 ;
0269' ED 6F      00428 H1:      RLD                     ; STORE IT
026B' 13         00429          INC     DE              ; NEXT ASCII
026C' CD 0126'   00430          CALL    HEXIT           ; CONVERT IT
026F' CB 7F      00431          BIT     7,A
0271' 28 05      00432          JR      Z,H2            ; IF NO ERROR JUMP
0273' DD E1      00433          POP     IX              ; STRIP TOP OF STACK
0275' C3 0098'   00434          JP      SYN1
                 00435 ;
0278' ED 6F      00436 H2:      RLD
027A' 13         00437          INC     DE              ; NEXT ASCII
027B' 2B         00438          DEC     HL              ; NEXT STORAGE BYTE
027C' 10 DF      00439          DJNZ    AGAIN
027E' C9         00440          RET
                 00441 ;
                 00442 ; BLNK BLANKS OUT THE COMMAND LINE.
                 00443 ;
027F' 01 0019    00444 BLNK:    LD      BC,19
0282' 11 FD82    00445          LD      DE,COMLOC
0285' 21 028E'   00446 BLNK1:   LD      HL,SPCE
0288' ED A0      00447          LDI
028A' EA 0285'   00448          JP      PE,BLNK1
028D' C9         00449          RET
                 00450 ;
028E' 20         00451 SPCE:    DEFB    20
                 00452 ;
                 00453 ; GO CHANGES THE USER'S PC (REGPC)
                 00454 ;
028F' 01 0012    00455 GO:      LD      BC,12
0292' 3E 20      00456          LD      A,20            ; ASCII <SP>
0294' 23         00457          INC     HL              ; NEXT CHARACTER
0295' ED B1      00458          CPIR                    ; SEARCH FOR BLANK
                 00459 ;
0297' E2 0098'   00460          JP      PO,SYN1         ; BLANK NOT FOUND
029A' 11 0000*   00461          LD      DE,REGPC        ; TARGET LOCATION
029D' EB         00462          EX      DE,HL
029E' 23         00463          INC     HL              ; HIGH BYTE REGPC
                 00464 ;
029F' CD 025B'   00465          CALL    HEX4
                 00466 ;
02A2' 1A         00467          LD      A,(DE)          ; NEXT CHARACTER
02A3' FE 0D      00468          CP      CR              ; IS IT <CR> ?
02A5' C2 0098'   00469          JP      NZ,SYN1
                 00470 ;
02A8' CD 0000*   00471          CALL    FINTOP
02AB' CD 0000*   00472          CALL    WIPE
02AE' CD 0000*   00473          CALL    TEXTUP
02B1' CD 0000*   00474          CALL    RSTATE
02B4' CD 0000*   00475          CALL    CURSOR
02B7' CD 0000*   00476          CALL    REVID
02BA' CD 0000*   00477          CALL    REVMEM
02BD' C3 0AB8'   00478          JP      DONE
                 00479 ;
                 00480 ;
```

```
02C0' 3E 12      00481 OU:     LD      A,VIDOFF
02C2' CD 0000*   00482         CALL    BANKSW          ; SWITCH OUT V-RAM
02C5' C3 0123    00483         JP      SYST            ; END PROGRAM
                 00484 ;
                 00485 ;
                 00486 ; SE HANDLES THE SET INSTRUCTION (WHICH CHANGES
                 00487 ; MEMORY OR REGISTER VALUES).  THE MAIN BODY OF
                 00488 ; THIS ROUTINE IS A BINARY DECISION TREE WHICH
                 00489 ; DETERMINES WHICH REGISTER TO SET AND STORES
                 00490 ; THAT INFO IN HL.
                 00491 ;
02C8' 01 0013    00492 SE:     LD      BC,12
02CB' 3E 20      00493         LD      A,20            ; ASCII <SP>
02CD' 23         00494         INC     HL              ; NEXT CHARACTER
02CE' ED B1      00495         CPIR
                 00496 ;
02D0' E2 00F8'   00497         JP      PO,SYN1         ; BLANK NOT FOUND
02D3' 7E         00498         LD      A,(HL)          ; FIRST CHARACTER
02D4' FE 45      00499         CP      45              ; ASCII E
02D6' FA 02EB    00500         JP      M,SEB           ; < E
02D9' C2 037E'   00501         JP      NZ,SEL          ; > E
                 00502 ;
02DC' 23         00503         INC     HL              ; NEXT CHARACTER
02DD' 7E         00504         LD      A,(HL)
02DE' FE 3D      00505         CP      3D              ; ASCII =
02E0' C2 00F8'   00506         JP      NZ,SYN1         ; ^= ERROR
                 00507 ;
02E3' EB         00508         EX      DE,HL
02E4' 13         00509*        INC     DE              ; NEXT CHARACTER
02E5' 21 0000*   00510         LD      HL,REGE         ; DESTINATION
02E8' C3 040A'   00511         JP      REG1
                 00512 ;
02EB' FE 42      00513 SEB:    CP      42              ; ASCII B
02ED' FA 030D'   00514         JP      M,SEAT          ; < B
02F0' 20 5B      00515         JR      NZ,SEC          ; > B
                 00516 ;
02F2' 23         00517         INC     HL              ; NEXT CHARACTER
02F3' 7E         00518         LD      A,(HL)
02F4' FE 3D      00519         CP      3D              ; ASCII =
02F6' 20 09      00520         JR      NZ,SEBC
                 00521 ;
02F8' EB         00522         EX      DE,HL
02F9' 13         00523         INC     DE              ; NEXT CHARACTER
02FA' 21 0000*   00524         LD      HL,REGB         ; DESTINATION
02FD' C3 040A'   00525         JP      REG1
                 00526 ;
0300' FE 43      00527 SEBC:   CP      43              ; ASCII C
0302' C2 00F8'   00528         JP      NZ,SYN1
                 00529 ;
0305' EB         00530         EX      DE,HL
0306' 13         00531         INC     DE
0307' 21 0000*   00532         LD      HL,REGB
030A' C3 0410'   00533         JP      REG2
                 00534 ;
030D' FE 40      00535 SEAT:   CP      40              ; ASCII @
030F' FA 00F8'   00536         JP      M,SYN1          ; < @
```

```
0312' 20 2A     00537           JR      NZ,SEA          ; > @ (A)
                00538 ;
                00539 ; SPECIAL CASE (SET MEMORY)
                00540 ;
0314' EB        00541           EX      DE,HL
0315' 13        00542           INC     DE              ; NEXT CHARACTER
0316' 21 033C'  00543           LD      HL,ATMEM+1      ; STORAGE
0319' CD 025B'  00544           CALL    HEX4
031C' EB        00545           EX      DE,HL
                00546 ;
031D' 7E        00547           LD      A,(HL)          ; NEXT CHARACTER
031E' FE 3D     00548           CP      3D              ; ASCII =
0320' C2 0098'  00549           JP      NZ,SYN1
                00550 ;
0323' EB        00551           EX      DE,HL
0324' 13        00552           INC     DE              ; NEXT CHARACTER
0325' 21 033D'  00553           LD      HL,WITH         ; STORAGE
0328' CD 0257'  00554           CALL    HEX2
                00555 ;
032B' 2A 033B'  00556           LD      HL,(ATMEM)      ; LOCATION
032E' 3A 033D'  00557           LD      A,(WITH)        ; VALUE
0331' 77        00558           LD      (HL),A
                00559 ;
0332' 1A        00560           LD      A,(DE)          ; NEXT CHARACTER
0333' FE 0D     00561           CP      CR              ; <CR> ?
0335' C2 0098'  00562           JP      NZ,SYN1
0338' C3 041A'  00563           JP      SEDONE
                00564 ;
033B'           00565 ATMEM:    DEFS    2
033D'           00566 WITH:     DEFS    1
                00567 ;
                00568 ; RESUME TREE
                00569 ;
033E' 23        00570 SEA:      INC     HL              ; NEXT CHARACTER
033F' 7E        00571           LD      A,(HL)
0340' FE 3D     00572           CP      3D              ; ASCII =
0342' C2 0098'  00573           JP      NZ,SYN1
                00574 ;
0345' EB        00575           EX      DE,HL
0346' 13        00576           INC     DE              ; NEXT CHARACTER
0347' 21 0000*  00577           LD      HL,REGA         ; DESTINATION
034A' C3 040A'  00578           JP      REG1
                00579 ;
034D' FE 43     00580 SEC:      CP      43              ; ASCII C
034F' 20 0F     00581           JR      NZ,SED          ; ^= C (D)
                00582 ;
0351' 23        00583           INC     HL              ; NEXT CHARACTER
0352' 7E        00584           LD      A,(HL)
0353' FE 3D     00585           CP      3D              ; ASCII =
0355' C2 0098'  00586           JP      NZ,SYN1
                00587 ;
0358' EB        00588           EX      DE,HL
0359' 13        00589           INC     DE              ; NEXT CHARACTER
035A' 21 0000*  00590           LD      HL,REGC         ; DESTINATION
035D' C3 040A'  00591           JP      REG1
                00592 ;
```

```
0360' 23          00593 SED:    INC     HL              ; NEXT CHARACTER
0361' 7E          00594         LD      A,(HL)
0362' FE 3D       00595         CP      3D              ; ASCII =
0364' FA 0098'    00596         JP      M,SYN1          ; < =
0367' 20 08       00597         JR      NZ,SEDE
                  00598 ;
0369' EB          00599         EX      DE,HL
036A' 13          00600         INC     DE              ; NEXT CHARACTER
036B' 21 0000*    00601         LD      HL,REGD         ; DESTINATION
036E' C3 040A'    00602         JP      REG1
                  00603 ;
0371' FE 45       00604 SEDE.   CP      45              ; ASCII E
0373' C2 0098'    00605         JP      NZ,SYN1
                  00606 ;
0376' EB          00607         EX      DE,HL
0377' 13          00608         INC     DE
0378' 21 0000*    00609         LD      HL,REGD
037B' C3 0410'    00610         JP      REG2
                  00611 ;
037E' FE 4C       00612 SEL:    CP      4C              ; ASCII L
0380' FA 03C0'    00613         JP      M,SEH           ; < L
0383' 20 0F       00614         JR      NZ,SEP          ; > P
                  00615 ;
0385' 23          00616         INC     HL              ; NEXT CHARACTER
0386' 7E          00617         LD      A,(HL)
0387' FE 3D       00618         CP      3D              ; ASCII =
0389' C2 0098'    00619         JP      NZ,SYN1
                  00620 ;
038C' EB          00621         EX      DE,HL
038D' 13          00622         INC     DE
038E' 21 0000*    00623         LD      HL,REGL         ; DESTINATION
0391' C3 040A'    00624         JP      REG1
                  00625 ;
0394' FE 50       00626 SEP:    CP      50              ; ASCII P
0396' FA 0098'    00627         JP      M,SYN1          ; < P
0399' 20 10       00628         JR      NZ,SES          ; > P
                  00629 ;
039B' 23          00630         INC     HL              ; NEXT CHARACTER
039C' 7E          00631         LD      A,(HL)
039D' FE 43       00632         CP      43
039F' C2 0098'    00633         JP      NZ,SYN1
                  00634 ;
03A2' EB          00635         EX      DE,HL
03A3' 13          00636         INC     DE              ; NEXT CHARACTER
03A4' 21 0000*    00637         LD      HL,REGPC
03A7' 23          00638         INC     HL              ; DESTINATION
03A8' C3 0410'    00639         JP      REG2
                  00640 ;
03AB' FE 53       00641 SES.    CP      53              ; ASCII S
03AD' C2 0098'    00642         JP      NZ,SYN1
                  00643 ;
03B0' 23          00644         INC     HL              ; NEXT CHARACTER
03B1' 7E          00645         LD      A,(HL)
03B2' FE 50       00646         CP      50              ; ASCII P
03B4' C2 0098'    00647         JP      NZ,SYN1
                  00648 ;
```

```
03B7' EB          00649          EX      DE,HL
03B8' 13          00650          INC     DE                ; NEXT CHARACTER
03B9' 21 0000*    00651          LD      HL,REGSP
03BC' 23          00652          INC     HL
03BD' C3 0410'    00653          JP      REG2
                  00654 ;
03C0' FE 48       00655 SEH:     CP      48                ; ASCII H
03C2' FA 0098'    00656          JP      M,SYN1
03C5' 20 1E       00657          JR      NZ,SEI
                  00658 ;
03C7' 23          00659          INC     HL                ; NEXT CHARACTER
03C8' 7E          00660          LD      A,(HL)
03C9' FE 3D       00661          CP      3D                ; ASCII =
03CB' FA 0098'    00662          JP      M,SYN1
03CE' 20 08       00663          JR      NZ,SEHL
                  00664 ;
03D0' EB          00665          EX      DE,HL
03D1' 13          00666          INC     DE                ; NEXT CHARACTER
03D2' 21 0000*    00667          LD      HL,REGH           ; DESTINATION
03D5' C3 040A'    00668          JP      REG1
                  00669 ;
03D8' FE 4C       00670 SEHL:    CP      4C                ; ASCII L
03DA' C2 0098'    00671          JP      NZ,SYN1
                  00672 ;
03DD' EB          00673          EX      DE,HL
03DE' 13          00674          INC     DE                ; NEXT CHARACTER
03DF' 21 0000*    00675          LD      HL,REGH           ; DESTINATION
03E2' C3 0410'    00676          JP      REG2
                  00677 ;
03E5' FE 49       00678 SEI:     CP      49                ; ASCII I
03E7' C2 0098'    00679          JP      NZ,SYN1
                  00680 ;
03EA' 23          00681          INC     HL
03EB' 7E          00682          LD      A,(HL)            ; NEXT CHARACTER
03EC' FE 58       00683          CP      58                ; ASCII X
03EE' FA 0098'    00684          JP      M,SYN1
03F1' 20 09       00685          JR      NZ,SEIY
                  00686 ;
03F3' EB          00687          EX      DE,HL
03F4' 13          00688          INC     DE                ; NEXT CHARACTER
03F5' 21 0000*    00689          LD      HL,REGIX
03F8' 23          00690          INC     HL                ; DESTINATION
03F9' C3 0410'    00691          JP      REG2
                  00692 ;
03FC' FE 59       00693 SEIY:    CP      59                ; ASCII Y
03FE' C2 0098'    00694          JP      NZ,SYN1
                  00695 ;
0401' EB          00696          EX      DE,HL
0402' 13          00697          INC     DE                ; NEXT CHARACTER
0403' 21 0000*    00698          LD      HL,REGIY
0406' 23          00699          INC     HL                ; DESTINATION
0407' C3 0410'    00700          JP      REG2
                  00701 ;
                  00702 ; THE REGISTERS ARE CHANGED HERE.
040A' CD 0257'    00703 REG1:    CALL    HEX2
040D' C3 041A'    00704          JP      SEDONE
```

```
                        00705 ;
0410' 1A                00706 REGI.     LD      A,(DE)          ; NEXT CHARACTER
0411' FE 3D             00707           CP      3D              ; ASCII =
0413' C2 0398'          00708           JP      NZ,SYN1
                        00709 ;
0416' 13                00710           INC     DE
0417' CD 03EB'          00711           CALL    HEX4
                        00712 ;
041A' 1A                00713 SECOND.   LD      A,(DE)          ; NEXT CHARACTER
041B' FE 0D             00714           CP      CR              ; <CR> ?
041D' C2 0098'          00715           JP      NZ,SYN1
                        00716 ;
0420' CD 0000*          00717           CALL    RSTATE          ; PUT UP RHS OF SCREEN
0423' CD 027F'          00718           CALL    BLNK
0426' CD 0000*          00719           CALL    CURSOR
0429' CD 0000*          00720           CALL    REVID
042C' CD 0000*          00721           CALL    REVMEM
042F' CD 0000*          00722           CALL    SAVE
0432' C3 0AB8'          00723           JP      DONE
                        00724 ;
                        00725 ;
                        00726 ; TRG HANDLES THE SIMULATION OF PROGRAM EXECUTION
                        00727 ; BY INTERPRETING THE INPUT STRING AND CALLING
                        00728 ; SIM THE APPROPRIATE NUMBER OF TIMES.
                        00729 ;
0435' CD 0A5D'          00730 TRG:      CALL    SAVOLD          ; SAVE RHS OF SCREEN
0438' 11 0946'          00731           LD      DE,TPAD
043B' 21 0944'          00732           LD      HL,ZEROA
043E' 01 0004           00733*          LD      BC,4
0441' ED A0             00734 T3:       LDI                     ; ASCII ZERO OUT TPAD
0443' 2B                00735           DEC     HL
0444' EA 0441'          00736           JP      PE,T3
                        00737 ;
0447' 21 0000           00738           LD      HL,00
044A' 22 0944'          00739           LD      (STEPS),HL      ; ZERO OUT STEPS
                        00740 ;
044D' 22 05F5'          00741           LD      (BIT7),HL       ; RESET CODE
0450' 22 05F7'          00742           LD      (BIT6),HL
0453' 22 05F9'          00743           LD      (BIT4),HL
0456' 22 05FB'          00744           LD      (BIT2),HL
0459' 22 05FD'          00745           LD      (BIT1),HL
045C' 22 05FF'          00746           LD      (BIT0),HL
045F' 22 0639'          00747           LD      (BIT7A),HL
0462' 22 063B'          00748           LD      (BIT6A),HL
0465' 22 063D'          00749           LD      (BIT4A),HL
0468' 22 063F'          00750           LD      (BIT2A),HL
046B' 22 0641'          00751           LD      (BIT1A),HL
046E' 22 0643'          00752           LD      (BIT0A),HL
0471' 22 0660'          00753           LD      (BIT7B),HL
0474' 22 0662'          00754           LD      (BIT6B),HL
0477' 22 0664'          00755           LD      (BIT4B),HL
047A' 22 0666'          00756           LD      (BIT2B),HL
047D' 22 0668'          00757           LD      (BIT1B),HL
0480' 22 066A'          00758           LD      (BIT0B),HL
                        00759 ;
0483' 21 0000*          00760           LD      HL,KEYIN        ; INPUT BUFFER
```

```
0486' 7E          00761          LD      A,(HL)          ; FIRST CHARACTER
0487' FE 3A       00762          CP      3A              ; ASCII 9 +1
0489' F2 04BA'    00763          JP      P,TREG
                  00764 ;
048C' FE 0D       00765          CP      CR              ; ASCII <CR>
048E' 20 08       00766          JR      NZ,T4
0490' 3E 01       00767          LD      A,1
0492' 32 0944'    00768          LD      (STEPS),A       ; ONE SIMULATION
0495' C3 04A6'    00769          JP      TNUM
                  00770 ;
0498' FE 30       00771 T4:      CP      30              ; ASCII 0
049A' FA 0098'    00772          JP      M,SYN1
049D' 11 0000*    00773          LD      DE,KEYIN
                  00774 ;
04A0' CD 094B'    00775          CALL    UNFORM
04A3' CD 015B'    00776          CALL    HEX4
                  00777 ;
04A6' ED 4B       00778 TNUM:    LD      BC,(STEPS)      ; LOAD COUNTERS
04A8' 0944'
04AA' 04          00779          INC     B
04AB' CD 0000*    00780          CALL    SIMUL           ; RUN SIMULATION
04AE' 0D          00781          DEC     C               ; INNER COUNTER
04AF' 20 FA       00782          JR      NZ,T2
04B1' 3A 0944'    00783          LD      A,(STEPS)       ; RESET INNER
04B4' 4F          00784          LD      C,A
04B5' 10 F4       00785          DJNZ    T2              ; DEC OUTER
04B7' C3 0926'    00786          JP      TDONE
                  00787 ;
04BA' 3E 00       00788 TREG:    LD      A,0
04BC' 32 091E'    00789          LD      (LOOP),A        ; ZERO LOOP
04BF' 32 0919'    00790          LD      (CINFO),A       ; ZERO CINFO
                  00791 ;
04C2' 3A 091E'    00792 TEST:    LD      A,(LOOP)
04C5' FE 01       00793          CP      1
04C7' 20 EB       00794          JR      NZ,T5           ; JUMP IF FIRST TIME
                  00795 ;
                  00796 ; THIS SECTION DETERMINE THE HEX VALUE OF AN ASCII
                  00797 ; NUMERIC INPUT STRING FOR THE RHS OF A TRIGGER
                  00798 ; CONDITION STATEMENT.  NUMERICS ARE NOT VALID FOR
                  00799 ; THE LHS OF A TRIGGER CONDITION (WITH RELATION).
                  00800 ;
04C9' 7E          00801          LD      A,(HL)          ; NEXT CHARACTER
04CA' FE 3A       00802          CP      3A              ; ASCII 9 +1
04CC' F2 0A8E'    00803          JP      P,CKIT          ; CHECK FOR A-F HEX
04CF' FE 30       00804          CP      30
04D1' FA 0098'    00805          JP      M,SYN1          ; TOO LOW FOR COMMAND
                  00806 ;
04D4' E5          00807          PUSH    HL
04D5' DD E1       00808          POP     IX              ; MOVE VALUE TO INDEXED
                  00809                                  ; REGISTER
04D7' DD 7E 02    00810 NUM:     LD      A,(IX+2)        ; 3RD CHARACTER
04DA' 11 0F30'    00811          LD      DE,VAL          ; PUT VALUE IN VAL
04DD' EB          00812          EX      DE,HL           ; (USED IN HEX2,4)
04DE' FE 30       00813          CP      30              ; ASCII 0
04E0' FA 04EC'    00814          JP      M,TNUM2         ; SHOULD BE TWO DIGITS
                  00815 ;
```

```
                    00816 ; ASSUME 4 DIGITS IN NUMBER.  WE WILL GET THE NUMBER
                    00817 ; AND STORE IT IN RHS.
                    00818 ;
04E3' 23            00819           INC     HL              ; POINT TO HIGH BYTE
04E4' CD 013E'      00820           CALL    HEX4
04E7' EB            00821           EX      DE,HL           ; POINT HL TO NEXT CHAR
04E8' 13            00822           INC     DE              ; POINT DE TO VALUE
04E9' C3 07A0'      00823           JP      TREG16
                    00824 ;
04EC' CD 0137'      00825 TNUM2:    CALL    HEX2            ; TWO DIGIT NUMBER
04EF' EB            00826           EX      DE,HL           ; POINT HL TO NEXT CHAR
04F0' 13            00827           INC     DE              ; POINT DE TO VALUE
04F1' C3 074A'      00828           JP      TREG8
                    00829 ;
                    00830 ; DECISION TREE
                    00831 ;
04F4' 7E            00832 T5:       LD      A,(HL)
04F5' FE 46         00833           CP      46              ; ASCII F
04F7' FA 0174'      00834           JP      M,T8            ; < F
04FA' C2 060C'      00835           JP      NZ,TL           ; > F
                    00836 ;
04FD' 23            00837           INC     HL
04FE' 7E            00838           LD      A,(HL)          ; NEXT CHARAACTER
04FF' FE 3D         00839           CP      3D              ; ASCII =
0501' C2 00F8'      00840           JP      NZ,SYN1
                    00841 ;
                    00842 ; THIS SECTION HANDLES THE F=BBBBBB INSTRUCTION.
                    00843 ;
0504' 3A 071E'      00844           LD      A,(LOOP)
0507' FE 00         00845           CP      0
0509' C2 00F8'      00846           JP      NZ,SYN1
                    00847 ;
050C' 06 00         00848           LD      B,0
050E' 23            00849           INC     HL              ; NEXT CHARACTER
050F' 7E            00850           LD      A,(HL)          ; BIT 7
0510' FE 58         00851           CP      58              ; ASCII X
0512' 20 11         00852           JR      NZ,T5.5
                    00853 ;
0514' DD 21         00854           LD      IX,0BFCB        ; MODIFY PROGRAM
0516' BFCB
0518' ED 22         00855           LD      (BIT7),IX
051A' 08F8'
051C' DD 22         00856           LD      (BIT7A),IX
051E' 0639'
0520' DD 22         00857           LD      (BIT7B),IX
0522' 0640'
0524' 18 0B         00858           JR      T6
0526' FE 30         00859 T5.5:     CP      30              ; ASCII 0
0528' 28 07         00860           JR      Z,T6
                    00861 ;
052A' FE 31         00862           CP      31              ; ASCII 1
052C' C2 00F8'      00863           JP      NZ,SYN1
052F' CB F8         00864           SET     7,B
                    00865 ;
0531' 23            00866 T6:       INC     HL              ; NEXT CHARACTER
0532' 7E            00867           LD      A,(HL)          ; BIT 6
```

```
0533  FE 58      00868          CP      58
0535  20 12      00869          JR      NZ,T6.5
                  00870 ;
0537  DD 21      00871          LD      IX,0B7CB
0539  B7CB
053B  DD 22      00872          LD      (BIT6),IX
053D  05F7
053F  DD 22      00873          LD      (BIT6A),IX
0541  063B
0543  DD 22      00874          LD      (BIT6B),IX
0545  0662
0547  18 0B      00875          JR      T7
0549  FE 30      00876 T6.5:    CP      30
054B  28 07      00877          JR      Z,T7
                  00878 ;
054D  FE 31      00879          CP      31
054F  C2 00F8    00880          JP      NZ,SYN1
0552  CB F0      00881          SET     6,B
                  00882 ;
0554  23         00883 T7:      INC     HL              ; NEXT CHARACTER
0555  7E         00884          LD      A,(HL)          ; BIT 4
0556  FE 58      00885          CP      58
0558  20 12      00886          JR      NZ,T7.5
                  00887 ;
055A  DD 21      00888          LD      IX,0A7CB
055C  A7CB
055E  DD 22      00889          LD      (BIT4),IX
0560  05F4
0562  DD 22      00890          LD      (BIT4A),IX
0564  063D
0566  DD 22      00891          LD      (BIT4B),IX
0568  0664
056A  18 0B      00892          JR      T8
056C  FE 30      00893 T7.5:    CP      30
056E  28 07      00894          JR      Z,T8
                  00895 ;
0570  FE 31      00896          CP      31
0572  C2 00F8    00897          JP      NZ,SYN1
0575  CB E0      00898          SET     4,B
                  00899 ;
0577  23         00900 T8:      INC     HL              ; NEXT CHARACTER
0578  7E         00901          LD      A,(HL)          ; BIT 2
0579  FE 58      00902          CP      58
057B  20 12      00903          JR      NZ,T8.5
                  00904 ;
057D  DD 21      00905          LD      IX,097CB
057F  97CB
0581  DD 22      00906          LD      (BIT2),IX
0583  05FB
0585  DD 22      00907          LD      (BIT2A),IX
0587  063F
0589  DD 22      00908          LD      (BIT2B),IX
058B  0666
058D  18 0B      00909          JR      T9
058F  FE 30      00910 T8.5:    CP      30
0591  28 07      00911          JR      Z,T9
```

```
                        00912 ;
05F3  FE 31             00913           CP      31
05F5  C2 0098           00914           JP      NZ,SYN1
05F8  CB D0             00915           SET     2,B
                        00916 ;
05FA  23                00917 T9:       INC     HL                      ; NEXT CHARACTER
05FB  7E                00918           LD      A,(HL)                  ; BIT 1
05FC  FE 58             00919           CP      58
05FE  20 12             00920           JR      NZ,T9.5
                        00921 ;
0600  DD 21             00922           LD      IX,08FCB
0602  8FCB
0604  DD 22             00923           LD      (BIT1),IX
0606  08FD
0608  DD 22             00924           LD      (BIT1A),IX
060A  0641
060C  DD 22             00925           LD      (BIT1B),IX
060E  0648
0610  18 0B             00926           JR      T10
0612  FE 30             00927 T9.5:     CP      30
0614  28 07             00928           JR      Z,T10
                        00929 ;
0616  FE 31             00930           CP      31
0618  C2 0098           00931           JP      NZ,SYN1
061B  CB C8             00932           SET     1,B
                        00933 ;
061D  23                00934 T10:      INC     HL                      ; NEXT CHARACTER
061E  7E                00935           LD      A,(HL)                  ; BIT 0
061F  FE 58             00936           CP      58
0621  20 12             00937           JR      NZ,T10.5
                        00938 ;
0623  DD 21             00939           LD      IX,087CB
0625  87CB
0627  DD 22             00940           LD      (BIT0),IX
0629  08FF
062B  DD 22             00941           LD      (BIT0A),IX
062D  0643
062F  DD 22             00942           LD      (BIT0B),IX
0631  064A
0633  18 0B             00943           JR      T11
0635  FE 30             00944 T10.5:    CP      30
0637  28 07             00945           JR      Z,T11
                        00946 ;
0639  FE 31             00947           CP      31
063B  C2 0098           00948           JP      NZ,SYN1
063E  CB C0             00949           SET     0,B
                        00950 ;
0640  78                00951 T11:      LD      A,B
0641  32 0F1F           00952           LD      (BPAT),A                ; STORE BIT PATTERN
0644  23                00953           INC     HL
0645  7E                00954           LD      A,(HL)
0646  FE 0D             00955           CP      CR                      ; IS IT <CR> ?
0648  C2 0607           00956           JP      NZ,T13
                        00957 ;
064B  CD 0000*          00958 T12:      CALL    SIMUL                   ; RUN SIMULATION
064E  3A 0000*          00959           LD      A,(REGF)
```

```
05F1' CB AF      00960          RES     5,A
05F3' CB FF      00961          RES     3,A
05F5' 00         00962 BIT7:    NOP
05F6' 00         00963          NOP
05F7' 00         00964 BIT6:    NOP
05F8' 00         00965          NOP
05F9' 00         00966 BIT4:    NOP
05FA' 00         00967          NOP
05FB' 00         00968 BIT2:    NOP
05FC' 00         00969          NOP
05FD' 00         00970 BIT1:    NOP
05FE' 00         00971          NOP
05FF' 00         00972 BIT0:    NOP
0600' 00         00973          NOP
0601' B8         00974          CP      B               ; CHECK FLAGS
0602' 20 E7      00975          JR      NZ,T12
0604' C3 0926'   00976          JP      TDONE
                 00977 ;
                 00978 ;
0607' 3E 20      00979 T13:     LD      A,20            ; ASCII <SP>
0609' 50         00980          LD      D,B             ; BIT PATTERN
060A' 01 000A    00981          LD      BC,0A
060D' BE         00982          CP      (HL)            ; CHECK FOR <SP>
060E' C2 0098'   00983          JP      NZ,SYN1
                 00984 ;
0611' ED A1      00985 T14:     CPI
0613' EA 0098'   00986          JP      PO,SYN1         ; NO NUMBERS FOUND
0616' CA 0611'   00987          JP      Z,T14
                 00988 ;
0619' 2B         00989          DEC     HL              ; FOUND SOMETHING
061A' 3E 3A      00990          LD      A,3A            ; ASCII :
061C' BE         00991          CP      (HL)
061D' C2 0098'   00992          JP      NZ,SYN1
                 00993 ;
0620' 23         00994          INC     HL              ; NEXT CHARACTER
0621' E5         00995          PUSH    HL
0622' D1         00996          POP     DE
                 00997 ;
0623' CD 094B'   00998          CALL    UNFORM
0626' CD 025B'   00999          CALL    HEX4
                 01000 ;
0629' 3A 091F'   01001          LD      A,(BPAT)        ; GET BIT PATTERN
062C' 57         01002          LD      D,A
062D' ED 4B      01003          LD      BC,(STEPS)      ; LOAD COUNTERS
062F' 0944'
0631' 04         01004          INC     B
                 01005 ;
0632' 3A 0000*   01006 T17:     LD      A,(REGF)        ; UNTIL F=BBBBBB
0635' CB AF      01007          RES     5,A             ; RUN SIMULATION
0637' CB 9F      01008          RES     3,A
0639' 00         01009 BIT7A:   NOP
063A' 00         01010          NOP
063B' 00         01011 BIT6A:   NOP
063C' 00         01012          NOP
063D' 00         01013 BIT4A:   NOP
063E' 00         01014          NOP
```

```
063F  00          01015 BIT2A:  NOP
0640  00          01016         NOP
0641  00          01017 BIT1A:  NOP
0642  00          01018         NOP
0643  00          01019 BIT0A:  NOP
0644  00          01020         NOP
0645  BA          01021         CP      D                 ; CHECK FLAGS
0646  28 05       01022         JR      Z,T17.5
                  01023 ;
0648  CD 0000*     01024         CALL    SIMUL             ; RUN SIMULATION
064B  18 E5       01025         JR      T17
                  01026 ;
064D  0D          01027 T17.5:  DEC     C                 ; INNER COUNTER
064E  20 09       01028         JR      NZ,T18
0650  3A 0944'    01029         LD      A,(STEPS)         ; RESET INNER COUNTER
0653  4F          01030         LD      C,A
0654  10 03       01031         DJNZ    T18               ; OUTER COUNTER
0656  C3 0926'    01032         JP      TDONE
                  01033 ;
0659  3A 0000*    01034 T18:    LD      A,(REGF)          ; UNTIL F^=BBBBBB
065C  CB AF       01035         RES     5,A               ; RUN SIMULATION
065E  CB 9F       01036         RES     3,A
0660  00          01037 BIT7B:  NOP
0661  00          01038         NOP
0662  00          01039 BIT6B:  NOP
0663  00          01040         NOP
0664  00          01041 BIT4B:  NOP
0665  00          01042         NOP
0666  00          01043 BIT2B:  NOP
0667  00          01044         NOP
0668  00          01045 BIT1B:  NOP
0669  00          01046         NOP
066A  00          01047 BIT0B:  NOP
066B  00          01048         NOP
066C  BA          01049         CP      D                 ; CHECK FLAGS
066D  20 C3       01050         JR      NZ,T17
                  01051 ;
066F  CD 0000*    01052         CALL    SIMUL             ; RUN SIMULATION
0672  18 E5       01053         JR      T18
                  01054 ;
                  01055 ; DECISION TREE (RESUMED).
                  01056 ;
0674  FE 41       01057 TB:     CP      42                ; ASCII B
0676  FA 068A'    01058         JP      M,TAT             ; < B
0679  20 1D       01059         JR      NZ,TD             ; > B
                  01060 ;
067B  23          01061         INC     HL                ; NEXT CHARACTER
067C  7E          01062         LD      A,(HL)
067D  FE 43       01063         CP      43                ; ASCII C
067F  11 0000*    01064         LD      DE,REGB           ; 8 BIT LOCATION
0682  C2 074A'    01065         JP      NZ,TREG8          ; 8 BIT COMP.  (B=,<..
                  01066 ;
0685  23          01067         INC     HL                ; NEXT CHARACTER
0686  1B          01068         DEC     DE                ; POINT TO BC
0687  C3 07A0'    01069         JP      TREG16            ; 16 BIT CP.  (BC=,<..
                  01070 ;
```

```
06BA' FE 40      01071 TAT:    CP      40              ; ASCII @
06BC' FA 0098'   01072         JP      M,SYN1
06BF' 28 07      01073         JR      Z,TAT1
                 01074 ;
06C1' 23         01075         INC     HL              ; NEXT CHAR (A=<...)
06C2' 11 0000*   01076         LD      DE,REGA         ; 8 BIT LOCATION
06C5' C3 074A'   01077         JP      TREG8
                 01078 ;
06C8' 23         01079 TAT1.   INC     HL              ; NEXT CHAR
06C9' 11 0918'   01080         LD      DE,REGMEM+1     ; TARGET
06CC' EB         01081         EX      DE,HL
06CD' CD 025B'   01082         CALL    HEX4            ; GET INPUT ADDRESS
06D0' EB         01083         EX      DE,HL           ; HL POINTS NEXT CHAR
06D1' ED 5B      01084         LD      DE,(REGMEM)     ; DE POINTS 8 BIT LOC'N
06D3' 0917'
06D5' C3 073A'   01085         JP      TRMEM
                 01086 ;
06D8' FE 44      01087 TD:     CP      44              ; ASCII D
06DA' FA 06BE'   01088         JP      M,TC            ; MUUST BE C
06DD' 20 16      01089         JR      NZ,TE           ; MUST BE E
                 01090 ;
06DF' 23         01091         INC     HL              ; NEXT CHAR
06E0' 7E         01092         LD      A,(HL)
06E1' FE 45      01093         CP      45              ; ASCII E
06E3' 11 0000*   01094         LD      DE,REGD         ; 8 BIT LOCATION
06E6' C2 074A'   01095         JP      NZ,TREG8        ; 8 BIT CP.  (D=<...)
                 01096 ;
06E9' 23         01097         INC     HL              ; NEXT CHAR
06EA' 1B         01098         DEC     DE              ; POINT TO DE
06EB' C3 07A0'   01099         JP      TREG16          ; 16 BIT CP.  (DE=<...)
                 01100 ;
06EE' 23         01101 TC:     INC     HL              ; NEXT CHAR (C=<...)
06EF' 11 0000*   01102         LD      DE,REGC
06F2' C3 074A'   01103         JP      TREG8           ; 8 BIT
                 01104 ;
06F5' 23         01105 TE:     INC     HL              ; NEXT CHAR (E=<...)
06F6' 11 0000*   01106         LD      DE,REGE
06F9' C3 074A'   01107         JP      TREG8
                 01108 ;
06FC' FE 4C      01109 TL:     CP      4C              ; ASCII L
06FE' FA 06DA'   01110         JP      M,TH            ; < L
0701' 20 3E      01111         JR      NZ,TS           ; > L
                 01112 ;
0703' 23         01113         INC     HL              ; NEXT CHAR (L=<...)
0704' 11 0000*   01114         LD      DE,REGL
0707' C3 074A'   01115         JP      TREG8
                 01116 ;
070A' FE 48      01117 TH:     CP      48              ; ASCII H
070C' FA 0098'   01118         JP      M,SYN1          ; < H
070F' 20 0F      01119         JR      NZ,TI           ; > H
                 01120 ;
0711' 23         01121         INC     HL              ; NEXT CHAR
0712' 7E         01122         LD      A,(HL)
0713' FE 4C      01123         CP      4C              ; ASCII L
0715' 11 0000*   01124         LD      DE,REGH         ; 8 BIT LOC'N
0718' C2 074A'   01125         JP      NZ,TREG8
```

```
                        01126 ;
06EB' 23                01127           INC     HL                      ; NEXT CHAR (HL=<C..)
06EC' 1B                01128           DEC     DE                      ; POINT TO HL
06ED' C3 07A0'          01129           JP      TREG16                  ;
                        01130 ;
06F0' FE 49             01131 TI:       CP      49                      ; ASCII I
06F2' C2 0098'          01132           JP      NZ,SYN1
06F5' 23                01133           INC     HL                      ; NEXT CHAR
06F6' 7E                01134           LD      A,(HL)
06F7' FE 58             01135           CP      58                      ; ASCII X
06F9' FA 0098'          01136           JP      M,SYN1
06FC' 20 07             01137           JR      NZ,TIY
                        01138 ;
06FE' 23                01139           INC     HL                      ; NEXT CHAR (IX=<C..)
06FF' 11 0000*          01140           LD      DE,REGIX
0702' C3 07A0'          01141           JP      TREG16
                        01142 ;
0705' FE 59             01143 TIY:      CP      59                      ; ASCII Y
0707' C2 0098'          01144           JP      NZ,SYN1
070A' 23                01145           INC     HL                      ; NEXT CHAR (IY=<C..)
070B' 11 0000*          01146           LD      DE,REGIY
070E' C3 07A0'          01147           JP      TREG16
                        01148 ;
0711' FE 53             01149 TS:       CP      53                      ; ASCII S
0713' FA 0727'          01150           JP      M,TPC                   ; < S
0716' C2 0098'          01151           JP      NZ,SYN1                 ; > S
                        01152 ;
0719' 23                01153           INC     HL                      ; NEXT CHAR
071A' 7E                01154           LD      A,(HL)
071B' FE 50             01155           CP      50                      ; ASCII P
071D' C2 0098'          01156           JP      NZ,SYN1
                        01157 ;
0720' 23                01158           INC     HL                      ; NEXT CHAR (SP=<C..)
0721' 11 0000*          01159           LD      DE,REGSP
0724' C3 07A0'          01160           JP      TREG16
                        01161 ;
0727' FE 50             01162 TPC:      CP      50                      ; ASCII P
0729' C2 0098'          01163           JP      NZ,SYN1
072C' 23                01164           INC     HL                      ; NEXT CHAR
072D' 7E                01165           LD      A,(HL)
072E' FE 43             01166           CP      43                      ; ASCII C
0730' C2 0098'          01167           JP      NZ,SYN1
                        01168 ;
0733' 23                01169           INC     HL                      ; NEXT CHAR (PC=<C..)
0734' 11 0000*          01170           LD      DE,REGPC
0737' C3 07A0'          01171           JP      TREG16
                        01172 ;
                        01173 ; FOR <MEM><REL><RHS+MEM> ASSUME 8 BIT COMPARE
                        01174 ;
073A' 3A 091E'          01175 TRMEM:    LD      A,(LOOP)                ; FIRST TIME
073D' FE 00             01176           CP      0                       ; THRU LOOP?
073F' 28 09             01177           JR      Z,TREG8                 ; IF SO JUMP
                        01178 ;
0741' 3A 0919'          01179           LD      A,(CINFO)               ; CHECK LENGTH BIT
0744' CB 7F             01180           BIT     7,A
0746' 28 02             01181           JR      Z,TREG8
```

```
0748' 18 56      01182          JR      TREG16
                 01183 ;
074A' 3A 091E'   01184 TREG8:   LD      A,(LOOP)        ; FIRST TIME
074D' FE 00      01185          CP      0               ; THRU LOOP ?
074F' 20 41      01186          JR      NZ,T8.1         ; IF NOT JUMP
                 01187 ;
0751' 3E 01      01188 TR16A:   LD      A,1
0753' 32 091E'   01189          LD      (LOOP),A        ; SET LOOP FLAG
0756' 3A 0919'   01190          LD      A,(CINFO)       ; GET INFO BYTE
0759' 47         01191          LD      B,A             ; PUT IT IN B
075A' ED 53      01192          LD      (LHS),DE        ; STORE LHS OF COMPARE
075C' 091C'
                 01193 ;
075E' 7E         01194          LD      A,(HL)          ; RELATIONAL CHAR
075F' FE 7E      01195          CP      7E              ; ASCII ~
0761' 20 04      01196          JR      NZ,T8.2
                 01197 ;
0763' CB E0      01198          SET     4,B             ; SET NOT FLAG
0765' 23         01199          INC     HL              ; NEXT CHAR
0766' 7E         01200          LD      A,(HL)
                 01201 ;
0767' FE 3D      01202 T8.2:    CP      3D              ; ASCII =
0769' 20 0A      01203          JR      NZ,T8.3
076B' CB C0      01204          SET     0,B             ; SET = FLAG
076D' 78         01205          LD      A,B             ; SAVE IT
076E' 32 0919'   01206          LD      (CINFO),A
0771' 23         01207          INC     HL              ; NEXT CHAR
0772' C3 04C2'   01208          JP      TRPT            ; CHECK RHS
                 01209 ;
0775' FE 3E      01210 T8.3:    CP      3E              ; ASCII >
0777' 20 0A      01211          JR      NZ,T8.4
0779' CB C8      01212          SET     1,B             ; SET > FLAG
077B' 78         01213          LD      A,B
077C' 32 0919'   01214          LD      (CINFO),A       ; SAVE IT
077F' 23         01215          INC     HL              ; NEXT CHAR
0780' C3 04C2'   01216          JP      TRPT            ; CHECK RHS
                 01217 ;
0783' FE 3C      01218 T8.4:    CP      3C              ; ASCII <
0785' C2 0098'   01219          JP      NZ,SYN1
0788' CB D0      01220          SET     2,B             ; SET < FLAG
078A' 78         01221          LD      A,B
078B' 32 0919'   01222          LD      (CINFO),A       ; SAVE IT
078E' 23         01223          INC     HL              ; NEXT CHAR
078F' C3 04C2'   01224          JP      TRPT
                 01225 ;
0792' 3A 0919'   01226 T8.1:    LD      A,(CINFO)       ; CHECK LENGTH FLAG
0795' CB 7F      01227          BIT     7,A
0797' C2 0098'   01228          JP      NZ,SYN1         ; JUMP IF 16 ('1')
                 01229 ;
079A' ED 53      01230          LD      (RHS),DE        ; RHS OF COMPARE
079C' 091A'
079E' 18 21      01231          JR      FINEND
                 01232 ;
07A0' 3A 091E    01233 TREG16.  LD      A,(LOOP)        ; FIRST TIME
07A3' FE 00      01234          CP      0               ; THRU LOOP ?
07A5' 20 CB      01235          JR      NZ,T16.1        ; IF NOT JUMP
```

```
                        01236 ;
07CF  3A 0919'          01237           LD      A,(CINFO)       ; GET INFO BYTE
07D2  CB FF             01238           SET     7,A             ; SET 16 BIT FLAG
07D4  32 0919'          01239           LD      (CINFO),A       ; SAVE IT
07D7  C3 07E1'          01240           JP      TR16A
                        01241 ;
07DA  3A 0919'          01242 TR16.1:   LD      A,(CINFO)       ; CHECK LENGTH FLAG
07DD  CB 7F             01243           BIT     7,A
07DF  CA 0098'          01244           JP      Z,SYN1          ; JUMP IF 8 BIT ('0')
                        01245 ;
07E2  ED 53             01246           LD      (RHS),DE        ; RHS OF COMPARE
07E4  091A'
07E6  C3 07C1'          01247           JP      FINEND
                        01248 ;
                        01249 ; FINEND FINDS THE END OF THE INPUT LINE (<CR> -OR-
                        01250 ; :###<CR>) AND LOADS THE APPROPRIATE VALUE IN
                        01251 ; THE LOOP COUNTER.
                        01252 ;
07C1  7E                01253 FINEND: LD        A,(HL)          ; BETTER BE <SP>, <CR>
07C2  FE 0D             01254           CP      CR              ; ASCII CR
07C4  20 08             01255           JR      NZ,F1
                        01256 ;
07C6  3E 01             01257           LD      A,1             ; ONCE THRU
07C8  32 0944'          01258           LD      (STEPS),A
07CB  C3 07EC'          01259           JP      DOIT
                        01260 ;
07CE  FE 20             01261 F1:       CP      20              ; ASCII <SP>
07D0  C2 0098'          01262           JP      NZ,SYN1
                        01263 ;
07D3  3E 20             01264           LD      A,20
07D5  06 08             01265           LD      B,8             ; MAX 8 SPACES
07D7  23                01266 F2:       INC     HL              ; NEXT CHAR
07D8  BE                01267           CP      (HL)            ; <SP> ?
07D9  20 02             01268           JR      NZ,F3           ; IF NOT JUMP
07DB  10 FA             01269           DJNZ    F2
                        01270 ;
07DD  7E                01271 F3:       LD      A,(HL)
07DE  FE 3A             01272           CP      3A              ; ASCII :
07E0  C2 0098'          01273           JP      NZ,SYN1
                        01274 ;
07E3  23                01275           INC     HL              ; NEXT CHAR
07E4  E5                01276           PUSH    HL
07E5  D1                01277           POP     DE              ; SAVE START LOC'N
                        01278 ;
07E6  CD 094B'          01279           CALL    UNFORM
07E9  CD 025B'          01280           CALL    HEX4            ; # IN STEPS (IN HEX)
                        01281 ;
07EC  ED 5B             01282 DOIT:     LD      DE,(LHS)        ; POINT TO LHS VALUE
07EE  091C'
07F0  2A 091A'          01283           LD      HL,(RHS)        ; POINT TO RHS VALUE
07F3  3A 0919'          01284           LD      A,(CINFO)       ; GET INFO BYTE
07F6  CB 67             01285           BIT     4,A             ; CHECK ^ FLAG
07F8  20 4D             01286           JR      NZ,DNOT         ; JUMP IF ^ ('0')
                        01287 ;
07FA  CB 47             01288           BIT     0,A             ; CHECK = FLAG
07FC  28 13             01289           JR      Z,DGL           ; JUMP IF > OR <
```

```
07FE' 3E CA     01290          LD      A,0CA           ; CC=ZERO (=)
0800' 32 089F'  01291          LD      (JP1),A         ; PUT IT IN JP1
0803' 32 08DC'  01292          LD      (JP1A),A
0806' 3E C2     01293          LD      A,0C2           ; CC=NON-ZERO (=)
0808' 32 08B5'  01294          LD      (JP2),A
080B' 32 08F8'  01295          LD      (JP2A),A
080E' C3 0891'  01296          JP      DCONT
                01297 ;
0811' 3A 0919'  01298 DGL:     LD      A,(CINFO)       ; GET INFO BYTE
0814' CB 4F     01299          BIT     1,A             ; CHECK > FLAG
0816' 28 14     01300          JR      Z,DL            ; JUMP (<)
0818' EB        01301          EX      DE,HL           ; SWITCH RHS&LHS
0819' 3E FA     01302          LD      A,0FA           ; CC=MINUS (>)
081B' 32 089F'  01303          LD      (JP1),A
081E' 32 08DC'  01304          LD      (JP1A),A
0821' 3E F2     01305          LD      A,0F2           ; CC=PLUS (>)
0823' 32 08B5'  01306          LD      (JP2),A
0826' 32 08F8'  01307          LD      (JP2A),A
0829' C3 0891'  01308          JP      DCONT
                01309 ;
082C' 3A 0919'  01310 DL:      LD      A,(CINFO)       ; GET INFO BYTE
082F' CB 57     01311          BIT     2,A             ; CHECK < FLAG
0831' CA 0098'  01312          JP      Z,SYN1          ; NOT <,>, OR =
0834' 3E FA     01313          LD      A,0FA           ; CC=MINUS (<)
0836' 32 089F'  01314          LD      (JP1),A
0839' 32 08DC'  01315          LD      (JP1A),A
083C' 3E F2     01316          LD      A,0F2           ; CC=PLUS (<)
083E' 32 08B5'  01317          LD      (JP2),A
0841' 32 08F8'  01318          LD      (JP2A),A
0844' C3 0891'  01319          JP      DCONT
                01320 ;
                01321 ;
                01322 ;
0847' CB 47     01323 DNOT:    BIT     0,A             ; CHECK = FLAG
0849' 28 13     01324          JR      Z,DNGL          ; JUMP IF > OR <
084B' 3E C2     01325          LD      A,0C2           ; CC=NON-ZERO (=)
084D' 32 089F'  01326          LD      (JP1),A         ; PUT IT IN JP1
0850' 32 08DC'  01327          LD      (JP1A),A
0853' 3E CA     01328          LD      A,0CA           ; CC=ZERO (=)
0855' 32 08B5'  01329          LD      (JP2),A
0858' 32 08F8'  01330          LD      (JP2A),A
085B' C3 0891'  01331          JP      DCONT
                01332 ;
085E' 3A 0919'  01333 DNGL:    LD      A,(CINFO)       ; GET INFO BYTE
0861' CB 4F     01334          BIT     1,A             ; CHECK > FLAG
0863' 28 14     01335          JR      Z,DNL           ; JUMP (<)
0865' EB        01336          EX      DE,HL           ; SWITCH RHS&LHS
0866' 3E F2     01337          LD      A,0F2           ; CC=PLUS (>)
0868' 32 089F'  01338          LD      (JP1),A
086B' 32 08DC'  01339          LD      (JP1A),A
086E' 3E FA     01340          LD      A,0FA           ; CC=MINUS (>)
0870' 32 08B5'  01341          LD      (JP2),A
0873' 32 08F8'  01342          LD      (JP2A),A
0876' C3 0891'  01343          JP      DCONT
                01344 ;
0879' 3A 0919'  01345 DNL:     LD      A,(CINFO)       ; GET INFO BYTE
```

```
087C' CB 57     01346          BIT     Z,A             ; CHECK < FLAG
087E' CA 0098'  01347          JP      Z,SYN1          ; NOT <,>, OR =
0881' 3E F2     01348          LD      A,0F2           ; OC=PLUS (<)
0883' 32 089F'  01349          LD      (JP1),A
0886' 32 08DC'  01350          LD      (JP1A),A
0889' 3E FA     01351          LD      A,0FA           ; OC=MINUS (>)
088B' 32 08E8'  01352          LD      (JP2),A
088E' 32 08F8'  01353          LD      (JP2A),A
                01354 ;
0891' ED 4B     01355 DCONT:   LD      BC,(STEPS)      ; LOAD LOOP COUNTERS
0893' 0944'
0895' 04        01356          INC     B
0896' 3A 0919'  01357          LD      A,(CINFO)       ; GET INFO BYTE
0899' CB 7F     01358          BIT     7,A
089B' 20 20     01359          JR      NZ,DO16         ; JUMP IF 16 BIT
                01360 ;
089D' 1A        01361 D1:      LD      A,(DE)
089E' BE        01362          CP      (HL)            ; COMPARE VALUES
089F' C2 08A7'  01363 JP1:     JP      NZ,D2           ; JUMP IF CORRECT
                01364                                  ; NZ IS MODIFIABLE
08A2' CD 0000*  01365          CALL    SIMUL           ; RUN SIMUL IF NOT
08A5' 18 F6     01366          JR      D1              ; CHECK AGAIN
                01367 ;
08A7' 0D        01368 D2:      DEC     C               ; INNER COUNTER
08A8' 20 09     01369          JR      NZ,D3
08AA' 3A 0944'  01370          LD      A,(STEPS)       ; RESET INNER COUNTER
08AD' 4F        01371          LD      C,A
08AE' 10 03     01372          DJNZ    D3              ; OUTER COUNTER
                01373 ;
08B0' C3 0926'  01374          JP      TDONE
                01375 ;
08B3' 1A        01376 D3:      LD      A,(DE)
08B4' BE        01377          CP      (HL)            ; COMPARE VALUES
08B5' CA 089D'  01378 JP2:     JP      Z,D1            ; JUMP IF CORRECT
                01379                                  ; Z IS MODIFIABLE
08B8' CD 0000*  01380          CALL    SIMUL           ; RUN SIMUL IF NOT
08BB' 18 F6     01381          JR      D3
                01382 ;
                01383 ;
08BD' ED 53     01384 DO16:    LD      (DEREG),DE      ; SAVE ADDRESS OF VALUE
08BF' 0922'
08C1' 22 0924'  01385          LD      (HLREG),HL      ; SAVE ADDR OF RHS VAL.
                01386 ;
08C4' 3A 08DC'  01387          LD      A,(JP1A)
08C7' CB AF     01388          RES     5,A             ; CHANGE MSL OR PSNC
08C9' 32 08DC'  01389          LD      (JP1A),A
                01390 ;
08CC' 3A 08F8'  01391          LD      A,(JP2A)
08CF' CB AF     01392          RES     5,A             ; CHANGE MSL OR PSNC
08D1' 32 08F8'  01393          LD      (JP2A),A
                01394 ;
08D4' CD 0900'  01395 D1A:     CALL    GETEM           ; LOAD DE, HL WITH (DE
                01396                                  ; AND (HL) RESPECTIVEL
08D7' E5        01397          PUSH    HL              ; SAVE IT
08D8' BF        01398          CP      A               ; RESET CARRY FLAG
08D9' ED 52     01399          SBC     HL,DE           ; SET FLAGS
```

```
08DB' E1          01400          POP     HL              ; GET IT BACK
08DC' C2 08E4'    01401 JP1A:    JP      NZ,D2A          ; JUMP IF CORRECT
                  01402                                  ; NZ IS SELF-MODIFIABLE
                  01403 ;
08DF' CD 0000*    01404          CALL    SIMUL           ; RUN SIMUL IF NOT
08E2' 18 F0       01405          JR      D1A             ; CHECK AGAIN
                  01406 ;
08E4' 0D          01407 D2A:     DEC     C               ; INNER COUNTER
08E5' 20 09       01408          JR      NZ,D3A
08E7' 3A 0944'    01409          LD      A,(STEPS)       ; RESET INNER COUNTER
08EA' 4F          01410          LD      C,A
08EB' 10 03       01411          DJNZ    D3A             ; OUTER COUNTER
08ED' C3 0926'    01412          JP      TDONE
                  01413 ;
08F0' CD 0900'    01414 D3A:     CALL    GETEM           ; LOAD DE, HL WITH (DE)
                  01415                                  ; AND (HL) RESPECTIVELY
08F3' E3          01416          PUSH    HL              ; SAVE IT
08F4' BF          01417          CP      A               ; RESET CARRY FLAG
08F5' ED 52       01418          SBC     HL,DE           ; SET FLAGS
08F7' E1          01419          POP     HL
08F8' CA 08D4'    01420 JP2A:    JP      Z,D1A           ; JUMP IF CORRECT
                  01421                                  ; Z IS SELF MODIFIABLE
                  01422 ;
08FB' CD 0000*    01423          CALL    SIMUL           ; RUN SIMUL IF NOT
08FE' 18 F0       01424          JR      D3A             ; CHECK AGAIN
                  01425 ;
                  01426 ;
                  01427 ; GETEM LOADS DE WITH (DE), AND HL WITH (HL).
                  01428 ;
0900' ED 5B       01429 GETEM:   LD      DE,(DEREG)      ; POINT TO LHS VALUE
0902' 0922'
0904' 2A 0924'    01430          LD      HL,(HLREG)      ; POINT TO RHS VALUE
                  01431 ;
0907' D5          01432          PUSH    DE              ; SAVE IT
0908' 5E          01433          LD      E,(HL)          ; GET LOW HALF OF (HL)
0909' 23          01434          INC     HL              ; POINT TO HIGH 1/2
090A' 56          01435          LD      D,(HL)          ; GET HIGH HALF OF (HL)
                  01436 ;
090B' DD E1       01437          POP     IX              ; GET OLD DE
090D' DD 6E 00    01438          LD      L,(IX)          ; GET LOW HALF
0910' DD 23       01439          INC     IX              ; POINT TO HIGH 1/2
0912' DD 66 00    01440          LD      H,(IX)          ; GET HIGH HALF
                  01441 ;
0915' C9          01442          RET
                  01443 ;
0916'             01444 FLG:     DEFS    1
0917'             01445 REGMEM:  DEFS    2
0919'             01446 CINFO:   DEFS    1
091A'             01447 RHS:     DEFS    2
091C'             01448 LHS:     DEFS    2
091E'             01449 LOOP:    DEFS    1
091F'             01450 BPAT:    DEFS    1
0920'             01451 VAL:     DEFS    2
0922'             01452 DEREG:   DEFS    2
0924'             01453 HLREG:   DEFS    2
                  01454 ;
```

```
0926' CD 0000*  01455 TDONE:  CALL    FINTOP
0929' CD 0000*  01456         CALL    WIPE
092C' CD 0000*  01457         CALL    TEXTUP
092F' CD 0000*  01458         CALL    RSTATE
0932' CD 0000*  01459         CALL    CURSOR
0935' CD 0000*  01460         CALL    REVID
0938' CD 0000*  01461         CALL    REVMEM
093B' CD 0000*  01462         CALL    SAVE
093E' CD 0177'  01463         CALL    COPY
0941' C3 0AB8'  01464         JP      DONE
                01465 ;
0944'           01466 STEPS:  DEFS    2
0946'           01467 TPAD:   DEFS    4
094A' 30        01468 ZEROA:  DEFB    30
                01469 ;
                01470 ; UNFORM
                01471 ; DETERMINE HEX VALUE FROM UNFORMATTED ASCII.
                01472 ;
094B' 06 03     01473 UNFORM: LD      B,3
094D' 3E 0D     01474         LD      A,CR            ; ASCII <CR>
094F' 23        01475 T15:    INC     HL
0950' 10 03     01476         DJNZ    T16             ; SYNTAX CHECK
0952' DD E1     01477         POP     IX              ; PEEL TOP OF STACK
0954' C3 0098'  01478         JP      SYN1
0957' BE        01479 T16:    CP      (HL)            ; SEARCH FOR <CR>
0958' 20 F3     01480         JR      NZ,T15
                01481 ;
095A' E5        01482         PUSH    HL
095B' BF        01483         CP      A               ; CLEAR CARRY
095C' ED 52     01484         SBC     HL,DE           ; L = # OF #'S
095E' 06 00     01485         LD      B,0
0960' 4D        01486         LD      C,L             ; BC IS LOOP COUNTER
0961' E1        01487         POP     HL
0962' 2B        01488         DEC     HL              ; POINTS TO LAST #
0963' 11 0949'  01489         LD      DE,TPAD+3       ; DESTINATION
0966' ED B8     01490         LDDR                    ; PUT # IN TPAD
                01491 ;
0968' 21 0945'  01492         LD      HL,STEPS+1
096B' 11 0946'  01493         LD      DE,TPAD
096E' C9        01494         RET
                01495 ;
                01496 ;
                01497 ; AU SWAPS  THE AF REGISTER  WITH AF'.  NO RECORD IS
                01498 ; KEPT TO INDICATE WHICH OF THE VALUES WAS THE SUCESSOR
                01499 ; OF THE ORIGINAL PRIMARY AF REGISTER.
096F' 2A 0000*  01500 AU:     LD      HL,(XRAF)       ; AF'
0972' ED 5B     01501         LD      DE,(REGAF)      ; AF
0974' 0000*
0976' ED 53     01502         LD      (XRAF),DE       ; EXCHANGE THEM
0978' 0000*
097A' 22 0000*  01503         LD      (REGAF),HL
                01504 ;
097D' CD 0000*  01505         CALL    RSTATE
0980' CD 0000*  01506         CALL    REVID
0983' CD 027F'  01507         CALL    BLNK
0986' CD 0000*  01508         CALL    CURSOR
```

```
09B9' C3 0AB8'   01509          JP       DONE
                 01510 ;
                 01511 ;
                 01512 ; UA SWAPS THE REMAINING GENERAL REGISTERS (BC, DE, HL)
                 01513 ; WITH THEIR ALTERNATES.  AGAIN NO RECORD IS KEPT OF THE
                 01514 ; ORIGINAL PRIMARY REGISTER SET.
09BC' 11 0000*   01515 UA:       LD       DE,TEMP          ; TEMPORARY STORAGE
09BF' 21 0000*   01516          LD       HL,XRBC          ; SOURCE
09C2' 01 0006    01517          LD       BC,6
09C5' ED B0      01518          LDIR
                 01519 ;
09C7' 13         01520          INC      DE               ; SKIP OVER XRAF
09C8' 13         01521          INC      DE
09C9' 23         01522          INC      HL               ; SKIP OVER REGAF
09CA' 23         01523          INC      HL
                 01524 ;
09CB' 0E 06      01525          LD       C,6
09CD' ED B0      01526          LDIR
                 01527 ;
09CF' 21 0000*   01528          LD       HL,TEMP          ; SOURCE
09D2' 13         01529          INC      DE               ; SKIP OVER REGAF
09D3' 13         01530          INC      DE
09D4' 0E 06      01531          LD       C,6
09D6' ED B0      01532          LDIR
                 01533 ;
09D8' CD 0000*   01534          CALL     RSTATE
09DB' CD 0000*   01535          CALL     REVID
09DE' CD 027F'   01536          CALL     BLNK
09E1' CD 0000*   01537          CALL     CURSOR
09E4' C3 0AB8'   01538          JP       DONE
                 01539 ;
                 01540 ;
                 01541 ; ON FILLS THE SCREEN WITH AN  EXPANDED VERSION OF
                 01542 ; THT MEMORY DISPLAY AREA.  TWENTY FOUR ROWS OF
                 01543 ; SIXTEEN BYTES EACH ARE DISPLAYED, BEGINNING WITH
                 01544 ; THE ADDRESS STORED IN MDISP.
09E7' CD 0000*   01545 ON:       CALL     WIPE
09EA' FD 21      01546          LD       IY,0F800         ; TOP LEFT OF V-RAM
09EC' F800
09EE' 2A 0000*   01547          LD       HL,(MDISP)       ; GET MEMORY ADDRESS
09F1' 0E 10      01548          LD       C,10             ; OUTER LOOP COUNTER
                 01549 ;
09F3' 7C         01550 ON3:      LD       A,H              ; WRITE MEM-ADDR TO VRAM
09F4' CD 0000*   01551          CALL     ACONV
09F7' FD 72 00   01552          LD       (IY),D
09FA' FD 73 01   01553          LD       (IY+1),E
09FD' 7D         01554          LD       A,L
09FE' CD 0000*   01555          CALL     ACONV
0A01' FD 72 02   01556          LD       (IY+2),D
0A04' FD 73 03   01557          LD       (IY+3),E
0A07' 3E 3A      01558          LD       A,':'
0A09' FD 77 04   01559          LD       (IY+4),A         ; END WRITE MEM-ADDR
                 01560 ;
0A0C' 06 10      01561          LD       B,10             ; LOAD INNER LOOP COUNT
0A0E' 11 0005    01562          LD       DE,5
```

```
09E1' FD 19      01563          ADD     IY,DE           ; UPDATE IY LOC'N
                 01564 ;
09E3' FD 23      01565 ON4:     INC     IY              ; SKIP A SPACE
09E5' 7E         01566          LD      A,(HL)          ; WRITE MEMORY CONTENTS
09E6' 23         01567          INC     HL              ; TO V-RAM
09E7' CD 0000*   01568          CALL    ACONV
09EA' FD 72 00   01569          LD      (IY),D
09ED' FD 23      01570          INC     IY
09EF' FD 73 00   01571          LD      (IY),E
09F2' FD 23      01572          INC     IY              ; END WRITE MEM-CONTENT
09F4' 10 ED      01573          DJNZ    ON4             ; RETURN FOR NEXT MEMOR
                 01574 ;
09F6' 0D         01575          DEC     C               ; DEC OUTER LOOP COUNTE
09F7' 11 001B    01576          LD      DE,1B
09FA' FD 19      01577          ADD     IY,DE           ; SKIP TO NEXT LINE
09FC' 20 C8      01578          JR      NZ,ON3          ; RETURN TO WRITE NEXT
                 01579                                  ; LINE
09FE' CD 0000*   01580          CALL    CURSOR
0A01' C3 0AB8'   01581          JP      DONE
                 01582 ;
                 01583 ;
                 01584 ; OF RESTORES THE SCREEN TO ITS NORMAL CONFIGURATION
                 01585 ; WHICH REFLECTS THE CURRENT STATE OF  THE PROGRAM.
0A04' CD 0000*   01586 OF:      CALL    WIPE
0A07' CD 0000*   01587          CALL    FINTOP
0A0A' CD 0000*   01588          CALL    TEXTUP
0A0D' C3 0A10'   01589          JP      NEE             ; CONT AT NEE
                 01590 ;
                 01591 ; NEE RESTORES THE RIGHT HAND SIDE OF THE SCREEN TO
                 01592 ; ITS NORMAL CONFIGURATION AND CURRENT VALUES
0A10' CD 0000*   01593 NEE:     CALL    RSTATE
0A13' CD 0000*   01594          CALL    CURSOR
0A16' CD 0000*   01595          CALL    REVID
0A19' CD 0000*   01596          CALL    REVMEM
0A1C' CD 027F'   01597          CALL    BLNK
0A1F' C3 0AB8'   01598          JP      DONE
                 01599 ;
                 01600 ;
                 01601 ; OL DISPLAYS THE RIGHT HAND SIDE OF THE SCREEN AS
                 01602 ; IT APPEARED IMMEDIATELY AFTER THE PRECEEDING TRIGGER
                 01603 ; CONDITION WAS MET.
0A22' 21 0ABD'   01604 OL:      LD      HL,OLDSCR       ; SAVE AREA
0A25' 11 F880    01605          LD      DE,0F880        ; START OF 8-BIT REGS
0A28' 3E 04      01606          LD      A,4             ; WRITE 4 LINES
0A2A' CD 0A4E'   01607          CALL    WRTE
                 01608 ;
0A2D' 11 FA10    01609          LD      DE,0FA10        ; START OF 16-BIT REGS
0A30' 3E 02      01610          LD      A,2             ; WRITE 2 LINES
0A32' CD 0A4E'   01611          CALL    WRTE
                 01612 ;
0A35' 11 FB00    01613          LD      DE,0FB00        ; START OF STACK AREA
0A38' 3E 01      01614          LD      A,1             ; WRITE 1 LINE
0A3A' CD 0A4E'   01615          CALL    WRTE
                 01616 ;
0A3D' 11 FBF0    01617          LD      DE,0FBF0        ; START OF MEMORY AREA
0A40' 3E 04      01618          LD      A,4             ; WRITE 4 LINES
```

```
0A42' CD 0A4E'  01619          CALL    WRTE
0A45' CD 027F'  01620          CALL    BLNK
0A48' CD 0000*  01621          CALL    CURSOR
0A4B' C3 0AB8'  01622          JP      DONE
                01623 ;
                01624 ; WRTE WRITES THE DATA STORED IN OLDSCR ON THE SCREEN
0A4E' 01 001D   01625 WRTE:    LD      BC,1D           ; INNER LOOP
0A51' ED B0     01626          LDIR                    ; WRITE TO V-RAM
0A53' 3D        01627          DEC     A               ; DEC OUTER COUNTER
0A54' 01 0033   01628          LD      BC,33
0A57' EB        01629          EX      DE,HL
0A58' 09        01630          ADD     HL,BC           ; SKIP TO NEXT LINE
0A59' EB        01631          EX      DE,HL
0A5A' 20 F2     01632          JR      NZ,WRTE         ; WRITE NEXT LINE
0A5C' C9        01633          RET
                01634 ;
                01635 ;
                01636 ; SAVOLD SAVES THE CURRENT RHS OF THE SCREEN
0A5D' 11 0ABD'  01637 SAVOLD:  LD      DE,OLDSCR       ; SAVE AREA
0A60' 21 F380   01638          LD      HL,0F380        ; START OF 8-BIT REGS
0A63' 3E 04     01639          LD      A,4             ; WRITE 4 LINES
0A65' CD 0A81'  01640          CALL    WRTE1
                01641 ;
0A68' 21 FA10   01642          LD      HL,0FA10        ; START OF 16-BIT REGS
0A6B' 3E 02     01643          LD      A,2             ; WRITE 2 LINES
0A6D' CD 0A81'  01644          CALL    WRTE1
                01645 ;
0A70' 21 FB00   01646          LD      HL,0FB00        ; START OF STACK AREA
0A73' 3E 01     01647          LD      A,1             ; WRITE 1 LINE
0A75' CD 0A81'  01648          CALL    WRTE1
                01649 ;
0A78' 21 FBF0   01650          LD      HL,0FBF0        ; START OF MEMORY AREA
0A7B' 3E 04     01651          LD      A,4             ; WRITE 4 LINES
0A7D' CD 0A81'  01652          CALL    WRTE1
0A80' C9        01653          RET
                01654 ;
                01655 ; WRTE1 WRITES THE DATA STORED ON THE SCREEN TO OLDSCR
0A81' 01 001D   01656 WRTE1:   LD      BC,1D           ; INNER LOOP
0A84' ED B0     01657          LDIR                    ; WRITE TO OLDSCR
0A86' 3D        01658          DEC     A               ; DEC OUTERR COUNTER
0A87' 01 0033   01659          LD      BC,33
0A8A' 09        01660          ADD     HL,BC           ; SKIP TO NEXT LINE
0A8B' 20 F4     01661          JR      NZ,WRTE1        ; WRITE NEXT LINE
0A8D' C9        01662          RET
                01663 ;
                01664 ; CKIT CHECKS TO DETERMINE IF THE RHS CHARACTER STRING
                01665 ; IS A REGISTER OR A HEX NUMBER.
0A8E' FE 41     01666 CKIT:    CP      41
0A90' FA 0098'  01667          JP      M,SYN1          ; < ASCII A
0A93' FE 47     01668          CP      47
0A95' F2 04F4'  01669          JP      P,T5            ; > ASCII F
                01670 ;
0A98' E5        01671          PUSH    HL              ; MOVE POINTER TO
0A99' DD E1     01672          POP     IX              ; INDEX REGISTER
0A9B' 3A 0919'  01673          LD      A,(CINFO)       ; LENGTH OF COMPARE
0A9E' CB 7F     01674          BIT     7,A
```

```
0AA0' 28 0B      01675          JR       Z,C8              ; 8 BIT LENGTH
                 01676 ;
0AA2' DD 7E 02   01677          LD       A,(IX+2)          ; GET THIRD CHARACTER
0AA5' FE 30      01678          CP       30
0AA7' FA 04F4'   01679          JP       M,T5              ; TWO CHARACTERS LONG
0AAA' C3 0407'   01680          JP       NUM               ; RETURN
                 01681 ;
0AAD' DD 7E 01   01682 C8:      LD       A,(IX+1)          ; GET 2ND CHARACTER
0AB0' FE 30      01683          CP       30
0AB2' FA 04F4'   01684          JP       M,T5              ; ONE CHARACTERR LONG
0AB5' C3 0407'   01685          JP       NUM               ; RETURN
                 01686 ;
0AB8' E1         01687 DONE.    POP      HL
0AB9' D1         01688          POP      DE
0ABA' C1         01689          POP      BC
0ABB' F1         01690          POP      AF
0ABC' C9         01691          RET
                 01692 ;
0ABD'            01693 OLDSCR:  DEFS     13F               ; HOLDS OLD SCREEN (RHS)
                 01694 ;
0BFC'            01695          DEFS     4F                ; OUR STACK AREA
0C4B'            01696 STKP:    DEFS     1
                 01697 ;
                 01698          END
```

MACROS:

SYMBOLS:

ACONV	09E8*	AGAIN	0258'	AQ	0213'	ASCO	0030	ASCPP	003A
ASCA	0041	ASCPP	0047	ATMEM	033B'	AU	096F'	BANKST	0004*
BANKEW	02C3*	BIT0	05FF'	BIT0A	0643'	BIT0B	066A'	BIT1	05FD'
BIT1A	0641'	BIT1B	0668'	BIT2	05FB'	BIT2A	063F'	BIT2B	0666'
BIT4	05F7'	BIT4A	063D'	BIT4B	0664'	BIT8	05F7'	BIT8A	063B'
BIT8B	0662'	BIT7	05F5'	BIT7A	0639'	BIT7B	0660'	BLNK	027F'
BLNK1	0285'	BPAT	091F'	C8	0AAD'	CICO	0060*	CINFO	0919'
CKIT'	0A8E'	COMLOC	FD82	COPY	0177'	CR	000D	CURERR	0125'
CURSES	00BC*	CURSH	0005	CURSL	0082	CURSOR	0A49*	CURSYN	009B
D1	089D'	D1A	08D4'	D2	08A7'	D2A	08E4'	D3	08B3'
D3A	08F0'	DCONT	0891'	DECIDE	0189'	DEREG	0922'	DGL	0811'
DI	0222'	DL	082C'	DNGL	085E'	DNL	0879'	DNOT	0847'
DO16	03BD'	DOIT	07EC'	DONE	0AB8'	DQ	01E1'	ERR	0143'
F1	07CE'	F2	07D7'	F3	07DD'	FINEND	07C1'	FINTOP	0A08*
FLG	0916'	GETEM	0900'	GO	023F'	GQ	0204'	H1	0269'
H2	0278'	HEX2	0257'	HEX4	025B'	HEXIT	0126I'	HLREG	0924'
INIT	0146'	JP1	089F'	JP1A	08DC'	JP2	08B5'	JP2A	08F8'
KEYIN	049E*	KYBD	000E	L1	014C'	L2	0167'	L21	006A'
L22	0062'	L23	00B6'	L24	00C7'	LHS	091C'	LOOP	091E'
MAIN	0000I'	MDISP	09BF*	ME382	00E4'	N21	0134'	N31	00E0*
NEE	0A10'	NQ	01F2'	NUM	04D7'	OF	0A04'	OFQ	01AA'
OL	0A22'	OLDSCR	0ABD'	ON	09B7'	ON3	09C3'	ON4	09E3'
ORGEND	0018*	POLL	0014	PRESAV	017E*	Q1	F8F0	QCURSH	0000
QCURSL	00FF	QQ	01D2'	QRESP	F8FF	QU	02C0'	QUERY	0035'
REG1	040A'	REG2	0410'	REGA	0693*	REGAF	097B*	REGB	0680*
REGC	06C0*	REGD	06B4*	REGE	06C7*	REGF	065A*	REGH	06E6*
REGIX	0700*	REGIY	070C*	REGL	06D5*	REGMEM	0917'	REGPC	0735*
REGSAV	0181*	REGSP	0722*	REVID	0A17*	REVMEM	0A1A*	RHS	091A'
RFT	002A'	RSTATE	0A11*	SAVE	093C*	SAVIT	0165*	SAVOLD	0A3D'
SCREEN	001E*	SE	02C8'	SEA	033E'	SEAT	030D'	SEB	02EB'
SEBC	0300'	SEC	034D'	SED	0360'	SEDE	0371'	SEDONE	041A'
SEH	03C0'	SEHL	03D8'	SEI	03E5'	SEIY	03FC'	SEL	037E'
SEP	0394'	SES	03AB'	SIMUL	08FC*	SPCE	023E'	SQ	01B2'
STEPS	0944'	STKP	0C4B'	STRT	00E2'	SYN	00A8I'	SYN1	0098'
SYST	0123	T10	05BD'	T10.5	05D5'	T11	05E0'	T12	05EB'
T13	0607'	T14	0611'	T15	094F'	T16	0957'	T16.1	07B2'
T17	0632'	T17.5	064D'	T18	0659'	T2	04AB'	T3	0441'
T4	0498'	T5	04F4'	T5.5	0526'	T6	0531'	T6.5	0549'
T7	0554'	T7.5	056C'	T8	0577'	T8.1	0792'	T8.2	0767'
T8.3	0775'	T8.4	0783'	T8.5	058F'	T9	059A'	T9.5	05B2'
TAT	068A'	TAT1	0698'	TB	0674'	TC	06BE'	TD	06A8'
TDONE	0926'	TE	06C5'	TEMP	09A0*	TEXTUP	0A0B*	TH	06DA'
TI	06F0'	TITL	F86C	TIY	0705'	TL	06CC'	TNUM	04A6'
TNUM2	04EC'	TPAD	0946'	TPC	0727'	TR16A	0751'	TRACK	0015*
TREG	04BA'	TREG16	07A0'	TREG8	074A'	TRG	0435'	TRMEM	073A'
TRFT	04C2'	TS	0711'	UA	098C'	UNFORM	094B'	UQ	01C3'
USP	C7FF	VAL	0920'	VIDEO	00ED	VIDOFF	0012	WIPE	0A05*
WITH	033D'	WRTE	0A4E'	WRTE1	0A81'	XRAF	0978*	XRBC	0990*
ZERO	0176'	ZEROA	094A'	ZIP	00FE'				

NO FATAL ERROR(S)

```
00001 .COMMENT>
00002                                   BY      G. BUZZARD
00003                                   AND     W. MACLEOD
00004                                   -
00005                                   -               FEBRUARY, 1981
00006                                   -       LAST UPDATED 07-SEP-81
00007          THIS IS A SET OF SUBROUTINES TO FILL THE SCREEN
00008 WITH A REPRESENTATION OF THE STATE OF THE I80 SYSTEM.
00009 THE LEFT SIDE OF THE SCREEN WILL DISPLAY A DISASSEMBLED
00010 CORE LISTING, WITH THE PROGRAM COUNTER POINTING TO
00011 THE MIDDLE LINE.  ON THE RIGHT SIDE ARE DISPLAYED THE
00012 CONTENTS OF THE UNPRIMED REGISTER SET, AS WELL AS THE
00013 TOP OF THE STACK AND SELECTED STORAGE LOCATIONS.
00014          THE CALLING PROGRAM MUST INITIALIZE A BANKSWITCH
00015 REGISTER BY CALLING SUBROUTINE BANKST.  THIS REFORMATS
00016 THE SCREEN AND SETS UP A "BANK SWITCH REGISTER" IN
00017 STORAGE.  (FOR A.K.A. XYCOM D.B.A HARDHAT COMPUTER
00018 CLAIMS THAT THE BANK SWITCH REGISTER IS READ/WRITE.
00019 NOT SO;  IT IS WRITE-ONLY.)
00020          THE USER'S REGISTERS ARE SAVED IN THE AREA
00021 BEGINNING AT REGSAV.  EACH USER REGISTER MAY BE
00022 ADDRESSED SEPARATELY AS REGR WHERE R IS ITS NAME
00023 (E.G.  THE ACCUMULATOR IS SAVED IN REGA.)
00024 .>
0010   00025 .RADIX  16
00026          EXTRN   DISASS          ;DISASSEMBLER
00027          EXTRN   HEX     ;WRITES LOCN AND HEX FIELDS
00028          EXTRN   LENGTH
00029          EXTRN   LINE            ;DISASSEMBLED TEXT
00030          EXTRN   OPCODE          ;LINE+12 OR 30
00031          EXTRN   ORGEND          ;TABLE OF ORGEND PAIRS
00032          EXTRN   INFO    ;CHARACTERIZES INSTRUCTIONS
00033          EXTRN   REGPC
00034          EXTRN   REGSAV          ;REGISTER SAVE AREA
00035          EXTRN   REGSP
00036          EXTRN   PRESAV          ;PREVIOUS REG SAVE AREA
00037 ;
00038          ENTRY   ACONV
00039          ENTRY   BANKSW
00040          ENTRY   BANKST
00041          ENTRY   CICO
00042          ENTRY   CURSES
00043          ENTRY   CURSOR
00044          ENTRY   FINTOP
00045          ENTRY   KEYIN
00046          ENTRY   MDISP
00047          ENTRY   N31
00048          ENTRY   REVID
00049          ENTRY   REVMEM
00050          ENTRY   SAVE
00051          ENTRY   SAVIT
00052          ENTRY   RSTATE
00053          ENTRY   SCREEN
00054          ENTRY   TEXTUP
00055          ENTRY   WIPE
00056 ;
```

```
0000'                   00057 MDISP:  DEFS    2         ; HOLDS ADDRESS OF MEMORY
                        00058                           ; TO BE DISPLAYED
                        00059 ;
0020                    00060 BLANK:  EQU     20        ;ASCII BLANK
F7B0                    00061 BEFOR   EQU     0F7B0     ;JUST BEFORE TOP OF SCREEN
0008                    00062 BS      EQU     08        ;ASCII BACKSPACE
000D                    00063 CR      EQU     0D        ;ASCII 'CR'
0081                    00064 CURSL   EQU     82        ;CURSOR HOME RELATIVE (LOW)
0000                    00065 NULL    EQU     00        ;ASCII NULL
000E                    00066 KYBD    EQU     0E        ;KEYBOARD DATA ADDRESS
FBA2                    00067 MEMLOC  EQU     0FBA2           ;V-RAM ADDRESS
FFFF                    00068 NIL     EQU     0FFFF     ;ORGEND SENTINEL MARKER
0014                    00069 POLL    EQU     14        ;KEYBOARD STATUS ADDRESS
FA12                    00070 R16BIT  EQU     0FA12
F800                    00071 SCRTOP  EQU     0F800     ;TOP OF SCREEN IN RAM
F845                    00072 SFLAGS  EQU     0F845     ;FLAG NAME FIELD
FB02                    00073 STKLOC  EQU     0FB02     ;STACK FIELD IN VIDEO RAM
F882                    00074 SREGS   EQU     0F882     ;REGISTER FIELD IN VIDEO RAM
00ED                    00075 VIDEO   EQU     0ED             ;FOR USE WITH BANKSW
0012                    00076 VIDOFF  EQU     12
                        00077 ;
                        00078 ; THE FOLLOWING TABLE HOLDS THE ADDRESSES OF THE SCREEN
                        00079 ; LOCATIONS OF THE REGISTER CONTENTS DISPLAY.  THIS
                        00080 ; THIS TABLE IS INDEXED INTO FROM THE REVID (REVERSE
                        00081 ; VIDEO) ROUTINE.
                        00082 ;
0001' 87 F8 88          00083 SCRLOC: DEFB    87,0F8,88,0F8,87,0F8,88,0F8     ; F REG
0005' F8 87 F8
0008' 88 F8
000A' 84 F8 85          00084         DEFB    84,0F8,85,0F8,84,0F8,85,0F8     ; A REG
000D' F8 84 F8
0010' 85 F8
0012' D7 F8 D8          00085         DEFB    0D7,0F8,0D8,0F8,0D7,0F8,0D8,0F8 ; C REG
0015' F8 D7 F8
0018' D8 F8
001A' D4 F8 D5          00086         DEFB    0D4,0F8,0D5,0F8,0D4,0F8,0D5,0F8 ; B REG
001D' F8 D4 F8
0020' D5 F8
0022' 27 F9 28          00087         DEFB    27,0F9,28,0F9,27,0F9,28,0F9     ; E REG
0025' F9 27 F9
0028' 28 F9
002A' 24 F9 25          00088         DEFB    24,0F9,25,0F9,24,0F9,25,0F9     ; D REG
002D' F9 24 F9
0030' 25 F9
0032' 77 F9 78          00089         DEFB    77,0F9,78,0F9,77,0F9,78,0F9     ; L REG
0035' F9 77 F9
0038' 78 F9
003A' 74 F9 75          00090         DEFB    74,0F9,75,0F9,74,0F9,75,0F9     ; H REG
003D' F9 74 F9
0040' 75 F9
                        00091 ;
0042' 15 FA 16          00092         DEFB    15,0FA,16,0FA,17,0FA,18,0FA     ; IX REG
0045' FA 17 FA
0048' 18 FA
004A' 15 FA 16          00093         DEFB    15,0FA,16,0FA,17,0FA,18,0FA     ; IX REG
004D' FA 17 FA
```

```
00E0' 1E FA
00E2' 1E FA 1F   00094          DEFB    1E,0FA,1F,0FA,20,0FA,21,0FA     ; SP RE
00E5' FA 20 FA
00E8' 21 FA
00EA' 1E FA 1F   00095          DEFB    1E,0FA,1F,0FA,20,0FA,21,0FA     ; SP RE
00ED' FA 20 FA
00F0' 21 FA
00F2' 26 FA 27   00096          DEFB    26,0FA,27,0FA,26,0FA,27,0FA     ; I REG
00F5' FA 26 FA
00F8' 27 FA
00FA' 63 FA 66   00097          DEFB    63,0FA,66,0FA,67,0FA,68,0FA     ; IY RE
00FD' FA 67 FA
0100' 68 FA
0102' 63 FA 66   00098          DEFB    63,0FA,66,0FA,67,0FA,68,0FA     ; IY RE
0105' FA 67 FA
0108' 68 FA
010A' 6E FA 6F   00099          DEFB    6E,0FA,6F,0FA,70,0FA,71,0FA     ; PC RE
010D' FA 70 FA
0110' 71 FA
0112' 6E FA 6F   00100          DEFB    6E,0FA,6F,0FA,70,0FA,71,0FA     ; PC RE
0115' FA 70 FA
0118' 71 FA
011A' 76 FA 77   00101          DEFB    76,0FA,77,0FA,76,0FA,77,0FA     ; R REG
011D' FA 76 FA
0120' 77 FA
                 00102 ;
0122' F5         00103 SCREEN.  PUSH    AF
0123' CD 0333'   00104          CALL    FINTOP          ;FIND TOP OF SCREEN
0126' 3E ED      00105          LD      A,VIDEO
0128' CD 0529'   00106          CALL    BANKSW          ;SWITCH IN VIDEO RAM
012B' CD 00B7'   00107          CALL    WIPE    ;WIPE IT CLEAN
012E' CD 00E0'   00108          CALL    TEXTUP  ;PUT UP DISASSEMBLED TEXT
0131' CD 0178'   00109          CALL    RSTATE  ;PUT UP REGISTER STATE
0134' CD 00CB'   00110          CALL    CURSOR  ;SEND IT TO USER'S AREA
0137' CD 0530'   00111          CALL    REVID   ;PUT REV. VIDEO IN REGS V-RAM
013A' CD 060C'   00112          CALL    REVMEM  ;REVERSE VIDEO OF MEMORY CHNG
013D' CD 057F'   00113          CALL    SAVE    ;SAVE MEMORY DISPLAYED
0140' 3E 12      00114          LD      A,VIDOFF
0142' CD 0529'   00115          CALL    BANKSW          ;SWITCH OUT VIDEO RAM
0145' F1         00116          POP     AF
0146' C9         00117          RET
                 00118 ;
                 00119 ; WIPE BLANKS THE ENTIRE SCREEN.
0147' E5         00120 WIPE:    PUSH    HL
0148' D5         00121          PUSH    DE
0149' C5         00122          PUSH    BC
014A' 21 F800    00123          LD      HL,SCRTOP
014D' 36 20      00124          LD      (HL),BLANK
014F' 01 07FF    00125          LD      BC,7FF
0152' E5         00126          PUSH    HL
0153' D1         00127          POP     DE
0154' 13         00128          INC     DE
0155' ED B0      00129          LDIR
0157' C1         00130          POP     BC
0158' D1         00131          POP     DE
0159' E1         00132          POP     HL
```

```
00CA' C9          00133          RET
                  00134 ;
                  00135 ; CURSOR REPOSITIONS (AND CAN ALTER) THE CURSOR.
00CB' 06 04       00136 CURSOR:  LD       B,4
00CD' 21 00DA'    00137          LD       HL,CURSES
00D0' 0E 00       00138 CRSOUT.  LD       C,00
00D2' ED A3       00139          OUTI
00D4' 0C          00140          INC      C
00D5' ED A3       00141          OUTI
00D7' 20 F7       00142          JR       NZ,CRSOUT
00D9' C9          00143          RET
00DA' 0E 05 0F    00144 CURSES:  DEFB     0E,05,0F,82,0A,40
00DD' 82 0A 40
                  00145 ;   THE LAST TWO BYTES WILL MAKE THE CURSOR BLINK.
                  00146 ;        MAKE LOOP COUNT = 6 TO INCLUDE THEM.
                  00147 ;
                  00148 .COMMENT]        TEXTUP PUTS THE DISASSEMBLED TEXT
                  00149 UP ON THE (MEMORY MAPPED AT F800-FFFF) SCREEN.
                  00150 IT EXPECTS HL TO POINT TO THE LOCATION TO BE DISPLAYED
                  00151 AT TOP OF SCREEN.   THE LINE POINTED TO BY USER PC WILL
                  00152 BE IN REVERSE VIDEO [BLACK ON WHITE.]
00E0' D5          00153 TEXTUP:  PUSH     DE
00E1' C5          00154          PUSH     BC
00E2' E5          00155          PUSH     HL
00E3' E5          00156          PUSH     HL
00E4' 21 F7B0     00157          LD       HL,BEFOR          ;JUST BEFORE SCREEN TOP
00E7' 22 0176'    00158          LD       (HERE),HL         ;SAVE SCREEN LOCATION
00EA' 06 18       00159          LD       B,18              ;LOOP COUNTER
00EC' E1          00160          POP      HL        ;BRING BACK REF POINTER
00ED' C5          00161 TXTP:    PUSH     BC
00EE' E5          00162          PUSH     HL        ;SAVE OLD REFERENCE POINTER
00EF' CD 03FA'    00163          CALL     WITHIN
00F2' 30 3B       00164          JR       NC,TIN
00F4' DD 21       00165          LD       IX,ORGEND
00F6' 0000*
00F8' DD 66 01    00166 TNEXT:   LD       H,(IX+1)
00FB' DD 6E 00    00167          LD       L,(IX+0)
00FE' C1          00168          POP      BC                ;GET REF POINTER
00FF' C5          00169          PUSH     BC                ;BALANCE STACK
0100' BF          00170          CP       A                 ;CLEAR CARRY FLAG
0101' ED 42       00171          SBC      HL,BC             ;FIND NEXT ORG
0103' 01 0004     00172          LD       BC,0004
0106' 30 04       00173          JR       NC,TFOUND
0108' DD 09       00174          ADD      IX,BC
010A' 18 EC       00175          JR       TNEXT             ;AFTER THE LAST ORG-END
                  00176                                    ;PAIR,THERE HAD BETTER
                  00177                                    ;BE THE SENTINEL:
                  00178                                    ;        FFFF,0000
                  00179 ;
010C' ED 42       00180 TFOUND:  SBC      HL,BC             ;HOW MANY BYTES?
010E' 30 03       00181          JR       NC,FOROFM         ;FOUR OF THEM
0110' 09          00182          ADD      HL,BC             ;OR FEWER
0111' E5          00183          PUSH     HL
0112' C1          00184          POP      BC
0113' E1          00185 FOROFM:  POP      HL
0114' 09          00186          ADD      HL,BC
```

```
011E' E5          00187          PUSH    HL       ;RESTORE REF POINTER
011F' ED 43       00188          LD      (LENGTH),BC
0113' 0000*
011A' ED 42       00189          SBC     HL,BC
011C' CD 0000*    00190          CALL    HEX
011F' 41          00191          LD      B,C              ;LOOP COUNTER
0120' 11 0000*    00192          LD      DE,OPCODE        ;POINT AT OPCODE FIELD
0123' CD 042A'    00193 FLOOP:   CALL    ASCII
0126' 10 FB       00194          DJNZ    FLOOP
0128' 18 20       00195          JR      NORMAL
012A' 3E 12       00196          LD      A,VIDOFF
012C' CD 0529'    00197          CALL    BANKSW           ; SWITCH OUT V-RAM
012F' CD 0000*    00198 TIN:     CALL    DISASS   ;DISASSEMBLE THE INSTR AT (HL)
0132' 3E ED'      00199          LD      A,VIDEO
0134' CD 0529'    00200          CALL    BANKSW           ; SWITCH IN V-RAM
0137' D1          00201          POP     DE       ;GET BACK OLD REF PTR
0138' E5          00202          PUSH    HL       ;SAVE NEW REFERENCE POINTER
013P' EB          00203          EX      DE,HL ;POINT AT INSTR JUST DISASSEMBLED
013A' ED 4B       00204          LD      BC,(REGPC)
013C' 0000*
013E' BF          00205          CP      A        ;CLEAR CARRY FLAG
013F' ED 42       00206          SBC     HL,BC    ;AT USER PC?
0141' 20 0A       00207          JR      NZ,NORMAL
0143' 06 28       00208          LD      B,28
0145' 21 0000*    00209          LD      HL,LINE
0148' CB FE       00210 REVERS:  SET     7,(HL)   ;REVERSE VIDEO IF SO
014A' 23          00211          INC     HL
014B' 10 FB       00212          DJNZ    REVERS
014D' 2A 0176'    00213 NORMAL:  LD      HL,(HERE)        ;GET SCREEN POINTER
0150' 11 0050     00214          LD      DE,50            ;LINE INCREMENT
0153' 19          00215          ADD     HL,DE            ;POINT TO NEXT LINE
0154' 22 0176'    00216          LD      (HERE),HL        ;SAVE SCREEN POINTER
0157' E5          00217          PUSH    HL               ;PUT SCREEN POINTER
0158' D1          00218          POP     DE               ;IN DESTINATION REG
015P' 21 0000*    00219          LD      HL,LINE          ;POINT AT TEXT
015D' 01 0032     00220          LD      BC,32            ;CHARACTER COUNT
015P' ED B0       00221          LDIR                     ;PUT IT ON SCREEN
                  00222 ;
0161' 21 0000*    00223          LD      HL,LINE          ; BLANK OUT LINE
0164' 0E 20       00224          LD      C,20             ; ASCII BLANK
0166' 06 32       00225          LD      B,32
0168' 71          00226 LOOP1:   LD      (HL),C           ; C ALREADY = 0
0169' 23          00227          INC     HL
016A' 10 FC       00228          DJNZ    LOOP1
                  00229 ;
016C' E1          00230          POP     HL               ;RESTORE SCRE POINTER
016D' C1          00231          POP     BC               ;RESTORE LOOP COUNTER
016E' 05          00232          DEC     B                ; DJNZ  TXTP
016F' C2 00ED'    00233          JP      NZ,TXTP          ; LOOP FOR 24 LINES
                  00234 ;
0172' E1          00235 TXRET:   POP     HL       ;RESTORE CALLER'S REGISTERS
0173' C1          00236          POP     BC
0174' D1          00237          POP     DE
0175' C9          00238          RET
0176'             00239 HERE:    DEFS    2
                  00240 ;
```

```
                    00241 ; RSTATE PUTS UP THE RIGHT HAND SIDE OFTHE SCREEN
                    00242 ; (THE REGISTER STATES AND MEMORY LOCATIONS).
0178' F5            00243 RSTATE: PUSH    AF          ;SAVE CALLER'S REGISTERS
0179' C5            00244         PUSH    BC
017A' D5            00245         PUSH    DE
017B' E5            00246         PUSH    HL
017C' DD E5         00247         PUSH    IX
017E' FD E5         00248         PUSH    IY
0180' 21 02F7'      00249         LD      HL,NAMES          ;REGISTER NAME LIST
0183' FD 21         00250         LD      IY,SFLAGS         ;REGISTER FIELD ADDR
0185' F843
0187' 06 08         00251         LD      B,8
0189' 7E            00252 NFLAG:  LD      A,(HL)            ;PUT UP FLAG LABELS
018A' FD 77 00      00253         LD      (IY),A
018D' FD 23         00254         INC     IY
018F' 23            00255         INC     HL
0190' 10 F7         00256         DJNZ    NFLAG
0192' FD 21         00257         LD      IY,SREGS          ;POINT TO REG FIELD
0194' F882
                    00258 ;
0196' DD 21         00259         LD      IX,REGSAV         ;USER'S REGISTER SET
0198' 0000*
019A' 06 04         00260         LD      B,4               ;LOOP COUNTER
019C' 7E            00261 REGS:   LD      A,(HL)            ;REGISTER NAME
019D' FD 77 00      00262         LD      (IY),A            ;TO VIDEO RAM
01A0' 3E 3A         00263         LD      A,':'
01A2' FD 77 01      00264         LD      (IY+1),A
01A5' DD 7E 00      00265         LD      A,(IX)            ;REGISTER CONTENTS
01A8' CD 04E2'      00266         CALL    ACONV
01AB' FD 72 05      00267         LD      (IY+5),D          ;ONTO SCREEN
01AE' FD 73 06      00268         LD      (IY+6),E
01B1' E5            00269         PUSH    HL                ;SAVE NAME POINTER
01B2' FD E5         00270         PUSH    IY
01B4' E1            00271         POP     HL                ;16-BIT LOAD
01B5' 11 0013       00272         LD      DE,13
01B8' 19            00273         ADD     HL,DE     ;POINT TO BINARY FIELD
01B9' CD 031E'      00274         CALL    PBITS
01BC' DD 7E 01      00275         LD      A,(IX+1)          ;NEXT REG. CONTENTS
01BF' CD 04E2'      00276         CALL    ACONV
01C2' FD 72 02      00277         LD      (IY+2),D          ;ONTO SCREEN
01C5' FD 73 03      00278         LD      (IY+3),E
01C8' 37            00279         SCF
01C9' 3F            00280         CCF                       ;CLEAR CARRY
01CA' 11 0011       00281         LD      DE,11
01CD' ED 52         00282         SBC     HL,DE     ;POINT TO NEXT BINARY FIELD
01CF' CD 031E'      00283         CALL    PBITS
01D2' E1            00284         POP     HL                ;RESTORE NAME POINTER
01D3' 23            00285         INC     HL
01D4' 3E 3A         00286         LD      A,':'
01D6' FD 77 07      00287         LD      (IY+7),A
01D9' 7E            00288         LD      A,(HL)            ;REGISTER NAME
01DA' FD 77 08      00289         LD      (IY+8),A
01DD' 23            00290         INC     HL
01DE' 11 0002       00291         LD      DE,02
01E1' DD 19         00292         ADD     IX,DE             ;NEXT REGISTER PAIR
01E3' 11 0050       00293         LD      DE,50
```

```
01E6' FD 19      00294          ADD     IY,DE           ;NEXT SCREEN LINE
01E8' 10 B2      00295          DJNZ    REG8
                 00296 ;
                 00297 .COMMENT>
                 00298
                 00299          THIS SECTION WRITES THE TITLES AND  CONTENTS OF
                 00300 THE 16-BIT REGISTERS, THE I REGISTER, AND THE R REGISTE
                 00301 TO THE VIDEO RAM.
                 00302 .>
01EA' FD 21      00303          LD      IY,R16BIT
01EC' FA12
01EE' 06 02      00304          LD      B,2
01F0' 7E         00305 SREG:    LD      A,(HL)  ; WRITE TITLE (FOR IX)
01F1' FD 77 00   00306          LD      (IY),A  ; (2ND TIME FOR IY)
01F4' 23         00307          INC     HL
01F5' 7E         00308          LD      A,(HL)
01F6' FD 77 01   00309          LD      (IY+1),A
01F9' 23         00310          INC     HL
01FA' 3E 3A      00311          LD      A,':'
01FC' FD 77 02   00312          LD      (IY+2),A        ; END WRITE TITLE
01FF' DD 7E 00   00313          LD      A,(IX)  ; WRITE CONTENTS OF IX
0202' DD 23      00314          INC     IX
0204' CD 04E2'   00315          CALL    ACONV   ; CONVERT TO ASCII
0207' FD 72 05   00316          LD      (IY+5),D
020A' FD 73 06   00317          LD      (IY+6),E
020D' DD 7E 00   00318          LD      A,(IX)
0210' DD 23      00319          INC     IX
0212' CD 04E2'   00320          CALL    ACONV
0215' FD 72 03   00321          LD      (IY+3),D
0218' FD 73 04   00322          LD      (IY+4),E ; END WRITE IX
021B' 7E         00323          LD      A,(HL)  ; WRITE TITLE (FOR SP)
021C' 23         00324          INC     HL      ; (2ND TIME FOR PC)
021D' FD 77 09   00325          LD      (IY+9),A
0220' 7E         00326          LD      A,(HL)
0221' 23         00327          INC     HL
0222' FD 77 0A   00328          LD      (IY+0A),A
0225' 3E 3A      00329          LD      A,':'
0227' FD 77 0B   00330          LD      (IY+0B),A
022A' DD 7E 00   00331          LD      A,(IX)  ; WRITE CONTENTS OF SP
022D' DD 23      00332          INC     IX
022F' CD 04E2'   00333          CALL    ACONV
0232' FD 72 0E   00334          LD      (IY+0E),D
0235' FD 73 0F   00335          LD      (IY+0F),E
0238' DD 7E 00   00336          LD      A,(IX)
023B' DD 23      00337          INC     IX
023D' CD 04E2'   00338          CALL    ACONV
0240' FD 72 0C   00339          LD      (IY+0C),D
0243' FD 73 0D   00340          LD      (IY+0D),E ; END WRITE SP
0246' 7E         00341          LD      A,(HL)  ; WRITE TITLE (FOR I)
0247' 23         00342          INC     HL      ; (2ND TIME FOR R)
0248' FD 77 12   00343          LD      (IY+12),A
024B' 3E 3A      00344          LD      A,':'
024D' FD 77 13   00345          LD      (IY+13),A
0250' DD 7E 00   00346          LD      A,(IX)  ; WRITE CONTENTS OF I
0253' DD 23      00347          INC     IX
0255' CD 04E2'   00348          CALL    ACONV
```

```
0258' FD 72 14   00349          LD      (IY+14),D
025B' FD 73 15   00350          LD      (IY+15),E
025E' 11 0050    00351          LD      DE,80
0261' FD 19      00352          ADD     IY,DE
0263' 10 BB      00353          DJNZ    SREG
                 00354 ;
                 00355 ; THIS SECTION WRITES THE TITLE AND TOP FOUR
                 00356 ; STACK LOCATION CONTENTS TO THE VIDEO RAM
0265' DD 2A      00357           LD      IX,(REGSP) ; USER STACK POINTER
0267' 0000*
0269' 11 FBC2    00358          LD      DE,STKLOC ; ADDRESS IN VIDEO RAM
026C' 01 0006    00359          LD      BC,6
026F' ED B0      00360          LDIR                ; WRITE "STACK:" TO V-RAM
0271' D5         00361          PUSH DE
0272' FD E1      00362          POP     IY          ; USE IY FOR ADDRESSES
0274' 06 04      00363          LD      B,4         ;  LOOP COUNTER
0276' FD 23      00364 N1:      INC     IY          ; SKIP SPACE
0278' DD 7E 01   00365          LD      A,(IX+1)            ; GET ONE NIBLE
027B' DD 23      00366          INC     IX          ;THIS IS AN UNROLLED
027D' CD 04E2'   00367          CALL    ACONV       ;LOOP
0280' FD 72 00   00368          LD      (IY),D
0283' FD 23      00369          INC     IY
0285' FD 73 00   00370          LD      (IY),E  ; WRITE NIBBLE
0288' FD 23      00371          INC     IY
028A' DD 7E FF   00372          LD      A,(IX-1)            ; GET ONE NIBLE
028D' DD 23      00373          INC     IX          ;THIS IS THE OTHER
028F' CD 04E2'   00374          CALL    ACONV       ;HALF OF IT.
0292' FD 72 00   00375          LD      (IY),D
0295' FD 23      00376          INC     IY
0297' FD 73 00   00377          LD      (IY),E  ; WRITE NIBBLE
029A' FD 23      00378          INC     IY
029C' 10 D8      00379          DJNZ    N1          , RETURN FOR NEXT NIBBLE
                 00380 ;
                 00381 ; THIS SECTION WRITES THE TITLE, ADDRESS, AND
                 00382 ; CONTENTS OF 32 CONTIGUOUS MEMORY LOCATIONS TO
                 00383 ; THE VIDEO RAM
029E' 11 FBA2    00384          LD      DE,MEMLOC ; LOAD V-RAM ADDRESS
02A1' 01 0007    00385          LD      BC,7
02A4' ED B0      00386          LDIR                ; WRTIE "MEMORY:" TO V-RAM
02A6' D5         00387          PUSH    DE
02A7' FD E1      00388          POP     IY          ; USE IY FOR V-RAM ADDRESS
02A9' 11 0049    00389          LD      DE,49   ; SKIP TO NEXT LINE
02AC' FD 19      00390          ADD     IY,DE   ;   "       "
02AE' 2A 0000'   00391          LD      HL,(MDISP) ; GET MEMORY DISPLAY
                 00392 ;                             LOCATION
02B1' 0E 04      00393          LD      C,4         , OUTER LOOP COUNTER
02B3' 7C         00394 N3:      LD      A,H         ; WRITE MEM-ADDR TO V-RAM
02B4' CD 04E2'   00395          CALL    ACONV
02B7' FD 72 00   00396          LD      (IY),D
02BA' FD 73 01   00397          LD      (IY+1),E
02BD' 7D         00398          LD      A,L
02BE' CD 04E2'   00399          CALL    ACONV
02C1' FD 72 02   00400          LD      (IY+2),D
02C4' FD 73 03   00401          LD      (IY+3),E
02C7' 3E 3A      00402          LD      A,':'
02C9' FD 77 04   00403          LD      (IY+4),A ; END WRITE MEM-ADDR
```

```
02CC  06 08      00404          LD      B,8       ; INNER LOOP COUNTER
02CE  11 0020    00405          LD      DE,32
02D1  FD 19      00406          ADD     IY,DE     ; UPDATE IY LOCATION
02D3  FD 23      00407 N4:      INC     IY        ; SKIP SPACE
02D5  7E         00408          LD      A,(HL)    ; WRITE MEMORY CONTENTS
02D6  23         00409          INC     HL        ; TO VIDEO RAM
02D7  CD 1DE1    00410          CALL    ADDNY
02DA  FD 71 00   00411          LD      (IY),C
02DD  FD 23      00412          INC     IY
02DF  FD 73 00   00413          LD      (IY),E
02E2  FD 23      00414          INC     IY        ; END WRITE MEMORY CONTENTS
02E4  10 EC      00415          DJNZ    N4        ; RETURN FOR NEXT MEMORY
02E6  0D         00416          DEC     C         ; DEC OUTEER LOOP
02E7  11 0033    00417          LD      DE,33
02EA  FD 19      00418          ADD     IY,DE     ; SKIP TO NEXT LINE
02EC  20 CE      00419          JR      NZ,N3     ; RETURN TO WRITE NEXT LINE
02EE  FD E1      00420          POP     IY
02F0  DD E1      00421          POP     IX
02F2  E1         00422          POP     HL
02F3  D1         00423          POP     DE
02F4  C1         00424          POP     BC
02F5  F1         00425          POP     AF
02F6  C9         00426          RET
02F7  53 5A 2A   00427 NAMES:   DEFM    'SZ*H*PNC'
02FA  48 2A 50
02FD  4E 43
0300  41 46 42   00428          DEFM    'AFBCDEHL'
0303  43 44 45
0306  48 4C
0308  49 58 58   00429          DEFM    'IXSPIIYPCR'
030B  50 49 49
030E  59 50 43
0311  52
0312  53 54 41   00430          DEFM    'STACK:MEMORY:'
0315  43 4B 3A
0318  4D 45 4D
031B  4F 52 59
031E  3A
                 00431 ;
                 00432 .COMMENT>
                 00433          PBITS PUTS THE CONTENTS OF THE ACCUMULATOR
                 00434 ON THE SCREEN IN BINARY FORMAT, AT (HL) IN VIDEO RAM.
                 00435  >
031F  C5         00436 PBITS:   PUSH    BC
0320  D5         00437          PUSH    DE
0321  06 08      00438          LD      B,8               ;LOOP 8 TIMES
0323  CB 07      00439 BITS:    RLC     A                 ;LOOK AT MSB
0325  38 04      00440          JR      C,ONE             ;SELECT CHARACTER
0327  16 30      00441          LD      D,'0'
0329  18 02      00442          JR      PUT
032B  16 31      00443 ONE:     LD      D,'1'
032D  72         00444 PUT:     LD      (HL),D            ;PUT CHAR. ON SCREEN
032E  23         00445          INC     HL
032F  10 F2      00446          DJNZ    BITS
0331  D1         00447          POP     DE
0332  C1         00448          POP     BC
```

```
0332' C9          00449          RET
                  00450 ;
                  00451 ;        FINTOP FINDS THE PLACE TO START DISASSEMBLY
                  00452 ;        SO THAT THE USER'S PROGRAM COUNTER WILL POINT
                  00453 ;        TO THE LINE IN THE MIDDLE OF THE SCREEN.
                  00454 ;        ON RETURN, HL POINTS TO THE LOCATION TO BE
                  00455 ;        DISPLAYED AT TOP OF SCREEN.
                  00456 ;
                  00457 ;        IN THE CIRCULAR QUEUE THAT "REMEMBERS" THE
                  00458 ;        LAST 12 LINES, WE WILL USE FFFF(HEX) TO MEAN
                  00459 ;        NIL.  SINCE THIS PROGRAM IS TO BE LOADED AT
                  00460 ;        THE TOP OF MEMORY, THE USER'S PROGRAM WILL
                  00461 ;        NEVER REACH FFXX.
                  00462 ;
0333' FD 21       00463 FINTOP: LD      IY,SPOT          ; FIRST SPOT IN QUEUE
0335' 3304'
0337' 2A 0000+    00464         LD      HL,(REGPC)       ; USER PC
033A' B7          00465         CP      A                ; CLEAR CARRY FLAG
033B' 01 0034     00466         LD      BC,34
033E' ED 42       00467         SBC     HL,BC            ; GO BACK 52 BYTES
                  00468 ;
0340' CD 03F4'    00469         CALL    WITHIN
0343' 30 20       00470         JR      NC,F1            ; JUMP IF WITHIN ORGEND
                  00471 ; FILL QUEUE WITH ADDRESSES (4 AT A TIME) IF NOT WITHIN
                  00472 ; ORG-END PAIR
0345' FD 75 00    00473 F2:     LD      (IY),L           ; PUT HL IN QUEUE
0348' FD 74 01    00474         LD      (IY+1),H
034B' 23          00475         INC     HL               ; NEXT ADDRESS (HL+4)
034C' 23          00476         INC     HL
034D' 23          00477         INC     HL
034E' 23          00478         INC     HL
034F' CD 0399'    00479         CALL    GNEXT            ; POINT TO NEXT SPOT
                  00480 ;
0351' CD 03F4'    00481         CALL    WITHIN
0354' 30 0E       00482         JR      NC,F1            ; REPEAT IF STILL OUT
                  00483 ; CHECK IF REF. POINTER (HL) IS >= REGPC -- IF SO JUMP
0356' EB          00484         EX      DE,HL            ; SAVE HL
0357' 2A 0000+    00485         LD      HL,(REGPC)
035A' B7          00486         CP      A                ; ZERO CARRY FLAG
035B' ED 52       00487         SBC     HL,DE
035D' EB          00488         EX      DE,HL
035E' 28 41       00489         JR      Z,FDONE          ; JUMP IF =
0360' 38 3F       00490         JR      C,FDONE          ; JUMP IF HL>REGPC
0362' 18 E0       00491         JR      F2
                  00492 ; IF WITHIN ORG-END START AT ORG, FILL QUEUE UNTIL
                  00493 ; END OR UNTIL HL=REGPC
0364' DD 66 01    00494 F1:     LD      H,(IX+1)         ; START AT ORG
0367' DD 6E 00    00495         LD      L,(IX)
                  00496 ; CHECK IF IX (HL) = (REGPC)
036A' E5          00497         PUSH    HL               ; SAVE HL
036B' B7          00498         CP      A                ; CLEAR CARRY FLAG
036C' ED 4B       00499         LD      BC,(REGPC)
036E' 0000+
0370' ED 42       00500         SBC     HL,BC
0372' 20 03       00501         JR      NZ,F1.5          ; JUMP IF NOT EQUAL
                  00502 ;
```

```
0373' E1          00503          POP     HL              ; BALANCE STACK
0374' 18 17       00504          JR      FDON
                  00505 ;
0376' E1          00506 F1.B.    POP     HL              ; BALANCE STACK
0377' FD 75 00    00507 F1.      LD      (IY),L          ; PUT HL IN QUEUE
037A' FD 74 01    00508          LD      (IY+1),H
037D' CD 0000*    00509          CALL    INFO            ; FIND INSTR LENGTH
0380' E6 03       00510          AND     03
0382' 3C          00511          INC     A
0383' 06 00       00512          LD      B,0
0385' 4F          00513          LD      C,A
                  00514
0386' 09          00515          ADD     HL,BC           ; INC REF. POINTER
0387' CD 03A9'    00516          CALL    CNEXT           ; POINT TO NEXT SPOT
                  00517 ; CHECK IF REF. POINTER (HL) IS >= REGPC -- IF SO JUMP
038A' EB          00518          EX      DE,HL           ; SAVE HL
038B' 2A 0000*    00519          LD      HL,(REGPC)
038E' BF          00520          CP      A               ; ZERO CARRY FLAG
038F' ED 52       00521          SBC     HL,DE
0391' EB          00522          EX      DE,HL
0392' 38 09       00523          JR      C,FDON          ; JUMP IF HL>REGPC
0394' 28 07       00524          JR      Z,FDON          ; JUMP IF =
                  00525
0396' CD 03DC     00526          CALL    INSIDE
0399' 38 A8       00527          JR      C,F2            ; IF OUTSIDE, JUMP
039B' 18 DA       00528          JR      F3              ; ELSE, REPEAT
                  00529 ;
039D' CD 03A9     00530 FDON:    CALL    CNEXT
03A0' FD 66 01    00531 FDONE:   LD      H,(IY+1)        ; PUT STARTING LOC'N
03A3' FD 6E 00    00532          LD      L,(IY)          ; IN HL
03A6' C9          00533          RET
                  00534 ;
                  00535 ;                         -000-
                  00536 ;
                  00537 ;CNEXT TAKES THE IY REGISTER (ASSUMED TO POINT TO AN
                  00538 ;ADDRESS WITHIN SPOT) AND RETURNS THE NEXT SPOT.
                  00539 ;ADDRESSES WITHIN SPOT WRAP AROUND; IT IS A FIFO QUEUE
                  00540 ;WITH ONLY TWELVE MEMBERS.
03A9' E5          00541 CNEXT:   PUSH    HL
03AA' C5          00542          PUSH    BC
03AB' 01 03DC     00543          LD      BC,SPOT-18
03AE' FD 23       00544          INC     IY
03B0' FD 23       00545          INC     IY
03B2' FD E5       00546          PUSH    IY
03B4' FD E5       00547          PUSH    IY
03B6' B7          00548          CP      A       ;RESET CARRY FLAG
03B7' E1          00549          POP     HL
03B8' ED 42       00550          SBC     HL,BC
03BA' FD E1       00551          POP     IY
03BC' C1          00552          POP     BC
03BD' E1          00553          POP     HL
03BE' D8          00554          RET     C
03BF' FD 21       00555          LD      IY,SPOT
03C1' 03C4'
03C3' C9          00556          RET
03C4'             00557 SPOT:    DEFS    18      ;CIRCULAR FIFO ADDRESS QUEUE
```

```
                        00558
                        00559 ;INSIDE RESETS THE CARRY FLAG IF THE HL ADDRESS LIES
                        00560 ;WITHIN THE ORGEND PAIR POINTED TO BY IX.
03DC' C5                00561 INSIDE: PUSH    BC          ;SAVE CALLER'S REGS
03DD' E5                00562         PUSH    HL          ;HL HOLDS ADDR
03DE' BF                00563         CP      A           ;CLEAR CARRY FLAG
03DF' DD 46 01          00564         LD      B,(IX+1)
03E2' DD 4E 00          00565         LD      C,(IX)      ;BC GETS ORG
03E5' ED 42             00566         SBC     HL,BC       ;SET CY IF ORG>ADDR
03E7' E1                00567         POP     HL
03E8' C1                00568         POP     BC
03E9' D8                00569         RET     C
03EA' C5                00570         PUSH    BC
03EB' E5                00571         PUSH    HL
03EC' C1                00572         POP     BC                  ;BC GETS ADDR
03ED' DD 66 03          00573         LD      H,(IX+3)
03F0' DD 6E 02          00574         LD      L,(IX+2)            ;HL GETS END
03F3' 2B                00575         DEC     HL
03F4' ED 42             00576         SBC     HL,BC       ;SET CY IF ADDR>(END-1)
03F6' C5                00577         PUSH    BC
03F7' E1                00578         POP     HL
03F8' C1                00579         POP     BC
03F9' C9                00580         RET
                        00581 ;
                        00582 ;WITHIN CHECKS THE HL REG AGAINST ALL ORGEND PAIRS.
                        00583 ;IT RESETS CARRY FLAG IF HL LIES INSIDE A PAIR.
                        00584 ;ON RETURN, IX POINTS TO THE ORG-END PAIR CONTAINING
                        00585 ;THE INSTRUCTION (IF INSIDE AN ORGEND.)
03FA' DD 21             00586 WITHIN. LD      IX,ORGEND           ;START AT BEGINNING.
03FC' 0000*
03FE' C5                00587         PUSH    BC
03FF' E5                00588         PUSH    HL                  ;SAVE CALLERS REGS
0400' DD 46 01          00589 WNEXT.  LD      B,(IX+1)
0403' DD 4E 00          00590         LD      C,(IX)              ;BC GETS ORG
0406' BF                00591         CP      A                   ;CLEAR CARRY FLAG
0407' ED 42             00592         SBC     HL,BC               ;SET CY IF ORG>ADDR
0409' 38 1C             00593         JR      C,WRET
040B' E1                00594         POP     HL                  ;RESTORE ADDR
040C' E5                00595         PUSH    HL
040D' CD 03DC           00596         CALL    INSIDE
0410' 30 18             00597         JR      NC,WRET
0412' E5                00598         PUSH    HL
0413' 01 0004           00599         LD      BC,0004
0416' DD 09             00600         ADD     IX,BC
0418' DD 66 01          00601         LD      H,(IX+1)
041B' DD 6E 00          00602         LD      L,(IX)              ;HL GETS ORG
041E' 01 0001           00603         LD      BC,0001
0421' 09                00604         ADD     HL,BC               ;ORG = FFFF?
0422' E1                00605         POP     HL
0423' 28 02             00606         JR      Z,WRET
0425' 18 D9             00607         JR      WNEXT       ;ORGEND HAD BETTER HAVE FFFF
                        00608                             ;AFTER IT!
0427' E1                00609 WRET:   POP     HL
0428' C1                00610         POP     BC
0429' C9                00611         RET
                        00612
```

```
042A' C5          00613 ASCII:  PUSH   BC
042B' E5          00614         PUSH   HL
042C' 7E          00615         LD     A,(HL)
042D' CB BF       00616         RES    7,A      ;TREAT THEM ALL AS ASCII
042F' FE 7F       00617         CP     7F
0431' 20 05       00618         JR     NZ,ASK
0433' 21 04DE'    00619         LD     HL,DLET
0436' 18 1D       00620         JR     MOV4
0438' FE 20       00621 ASK:    CP     20
043A' 38 0E       00622         JR     C,NONPR
043C' 12          00623         LD     (DE),A
043D' 13          00624         INC    DE
043E' 3E 20       00625         LD     A,BLANK
0440' 06 03       00626         LD     B,3
0442' 12          00627 BLOOP:  LD     (DE),A
0443' 13          00628         INC    DE
0444' 10 FC       00629         DJNZ   BLOOP
0446' E1          00630         POP    HL
0447' 23          00631         INC    HL
0448' C1          00632         POP    BC
0449' C9          00633         RET
044A' 06 00       00634 NONPR:  LD     B,0
044C' CB 27       00635         SLA    A
044E' CB 27       00636         SLA    A
0450' 4F          00637         LD     C,A
0451' 21 045E'    00638         LD     HL,CTRL
0454' 09          00639         ADD    HL,BC
0455' 01 0004     00640 MOV4:   LD     BC,4
0458' ED B0       00641         LDIR
045A' E1          00642         POP    HL
045B' 23          00643         INC    HL
045C' C1          00644         POP    BC
045D' C9          00645         RET
045E' 4E 55 4C    00646 CTRL:   DEFM   'NUL SOH STX ETX EOT ENQ ACK BEL
0461' 20 53 4F
0464' 48 20 53
0467' 54 58 20
046A' 45 54 58
046D' 20 45 4F
0470' 54 20 45
0473' 4E 51 20
0476' 41 43 4B
0479' 20 42 45
047C' 4C 20
047E' 42 53 20    00647         DEFM    BS  HT  LF  VT  FF  CR  SO  SI
0481' 20 48 54
0484' 20 20 4C
0487' 46 20 20
048A' 56 54 20
048D' 20 46 46
0490' 20 20 43
0493' 52 20 20
0496' 53 4F 20
0499' 20 53 49
049C' 20 20
049E' 44 4C 45    00648         DEFM    DLE DC1 DC2 DC3 DC4 NAK SYN ETB
```

```
04A1' 20 44 43
04A4' 31 20 44
04A7' 43 32 20
04AA' 44 43 33
04AD' 20 44 43
04B0' 34 20 4E
04B3' 41 4E 20
04B6' 53 59 4E
04B9' 20 45 54
04BC' 42 20
04BE' 43 41 4E   00649          DEFM     'CAN EM  SUB ESC FS  GS  RS  VS
04C1' 20 45 4D
04C4' 20 20 53
04C7' 55 42 20
04CA' 45 53 43
04CD' 20 46 53
04D0' 20 20 47
04D3' 53 20 20
04D6' 52 53 20
04D9' 20 56 53
04DC' 20 20
04DE' 44 45 4C   00650 DLET:    DEFM     'DEL '
04E1' 20
                 00651 ;
04E2' E5         00652 ACONV:   PUSH     HL
04E3' F5         00653          PUSH     AF
04E4' 21 0500'   00654          LD       HL,ALOC
04E7' 77         00655          LD       (HL),A
04E8' 3E 33      00656          LD       A,33H
04EA' ED 6F      00657          RLD
04EC' ED 6F      00658          RLD
04EE' FE 3A      00659          CP       3AH
04F0' 38 02      00660          JR       C,ANAJ1
04F2' C6 07      00661          ADD      A,7
04F4' 5F         00662 ANAJ1:   LD       E,A
04F5' 7E         00663          LD       A,(HL)
04F6' FE 3A      00664          CP       3AH
04F8' 38 02      00665          JR       C,ANAJ2
04FA' C6 07      00666          ADD      A,7
04FC' 57         00667 ANAJ2:   LD       D,A
04FD' F1         00668          POP      AF
04FE' E1         00669          POP      HL
04FF' C9         00670          RET
0500'            00671 ALOC:    DEFS     1
                 00672 ;
                 00673 ;                              -=<(*)>=-
                 00674 ;
                 00675 ;BANKST INITIALIZES THE BANK SWITCHING REGISTER, AND
                 00676 ;SETS UP A SAVE FIELD (XYCOM'S OUTSTANDING DOCUMENTATION
                 00677 ;NOTWITHSTANDING, THE BANK SWITCHING REGISTER IS WRITE
                 00678 ;ONLY.)
                 00679 ;        IT ALSO SETS UP THE CRT CONTROLLER FOR 24
                 00680 ;80-CHARACTER LINES.
0501' F5         00681 BANKST:  PUSH     AF               ;SAVE ACCUMULATOR
0502' C5         00682          PUSH     BC
0503' E5         00683          PUSH     HL
```

```
0504' 3E 03      00684          LD      A,03              ;DEFAULT SETTING
0506' 32 051E'   00685          LD      (BANKS),A         ;SAVE BANK STATUS
0509' D3 2F      00686          OUT     (2F),A            ;WRITE TO BANK REGISTER
                 00687 ;
050B' 21 051F'   00688          LD      HL,VIDSET         ;CRT CONTROLLER SETTING
050E' 06 05      00689          LD      B,5               ;# OF CRTC REGS TO FIX
0510' 7E         00690 SETVID:  LD      A,(HL)            ;GET REGISTER NUMBER
0511' D3 00      00691          OUT     (VREG),A          ;SEND TO CRTC REG SELECT
0513' 23         00692          INC     HL                ;POINT TO NEXT ENTRY
0514' 7E         00693          LD      A,(HL)            ;GET REGISTER CONTENTS
0515' D3 01      00694          OUT     (VWORD),A         ;SEND TO CRTC REGISTER
0517' 23         00695          INC     HL                ;POINT TO NEXT ENTRY
0518' 10 F6      00696          DJNZ    SETVID
051A' E1         00697          POP     HL
051B' C1         00698          POP     BC
051C' F1         00699          POP     AF
051D' C9         00700          RET
                 00701 ,
051E'            00702 BANKS:   DEFS    1
051F' 04 18 05   00703 VIDSET:  DEFB    04,18,05,1A,06,18,07,18,09,0A
0522' 1A 06 18
0525' 07 18 09
0528' 0A
0000             00704 VREG     EQU     00
0001             00705 VWORD    EQU     01
                 00706 ;
                 00707 .COMMENT#
                 00708                                  -(*)-
                 00709 BANKSW ALTERS THE STORED (SAVED) BANK SWITCH REGISTER
                 00710 AND SENDS AN ALTERED WORD TO THE HARDWARE BANK REG.
                 00711 IT EXPECTS THE ACCUMULATOR TO CONTAIN INFORMATION
                 00712 SPECIFYING THE CHANGE TO BE MADE.
                 00713 TO TURN ON A BIT OR BITS, SET BIT 4 OF THE ACC. AND
                 00714 SET THE DESIRED BIT/S (ALL OTHERS SHOULD BE ZERO.)
                 00715 TO TURN OFF A BIT OR BITS, RESET BIT 4 AND THE DESIRED
                 00716 BIT/S, WITH ALL OTHER BITS SET.
                 00717                                  -(*)-
                 00718 #
0529' E5         00719 BANKSW:  PUSH    HL
052A' 21 051E'   00720          LD      HL,BANKS          ;POINT TO STATUS BYTE
052D' CB 67      00721          BIT     4,A               ;TURN ON OR OFF?
052F' 28 03      00722          JR      Z,OFF             ;
0531' B6         00723          OR      (HL)              ;TURN BIT(S) ON
0532' 18 01      00724          JR      GOBAK             ;
0534' A6         00725 OFF:     AND     (HL)              ;TURN BIT(S) OFF
0535' 32 051E'   00726 GOBAK:   LD      (BANKS),A         ;SAVE STATUS BYTE
0538' E1         00727          POP     HL
0539' D3 2F      00728          OUT     (2F),A            ;WRITE TO BANK REGISTER
053B' C9         00729          RET
                 00730 ;
                 00731 ; THIS ROUTINE COMPARES THE VALUES OF THE USER'S
                 00732 ; REGISTER SET BEFORE AND AFTER THE SIMULATION OF
                 00733 ; THE PREVIOUS (USER'S) INSTRUCTION.  IF A DIFFERENCE
                 00734 ; IS FOUND THE CORRESPONDING REGISTER VALUE IS
                 00735 ; DISPLAYED IN REVERSE VIDEO IN THE VIDEO RAM.
                 00736 ;
```

```
053C' F5          00737 REVID:  PUSH    AF
053D' C5          00738         PUSH    BC
053E' D5          00739         PUSH    DE
053F' E5          00740         PUSH    HL
                  00741 ;
0540' 01 0012     00742         LD      BC,12           ; LENGTH OF COMPARE
0543' 11 0000*    00743         LD      DE,PRESAV       ; OLD ONES
0546' 21 0000*    00744         LD      HL,REGSAV       ; NEW ONES
                  00745 ;
0549' 1A          00746 L1:     LD      A,(DE)          ; OLD VALUE
054A' ED A1       00747         CPI                     ; COMPARE 'EM
054C' 13          00748         INC     DE
054D' C4 0558'    00749         CALL    NZ,REV          ; IF DIFF REVERSE IT
0550' EA 0549'    00750         JP      PE,L1           ; LOOP IF NOT DONE
                  00751 ;
0553' E1          00752         POP     HL
0554' D1          00753         POP     DE
0555' C1          00754         POP     BC
0556' F1          00755         POP     AF
0557' C9          00756         RET
                  00757 ;
0558' F5          00758 REV:    PUSH    AF
0559' C5          00759         PUSH    BC
055A' D5          00760         PUSH    DE
055B' E5          00761         PUSH    HL
                  00762 ;
055C' 11 0000*    00763         LD      DE,REGSAV       ; START ADDRESS
055F' 2B          00764         DEC     HL
0560' 37          00765         SCF
0561' 3F          00766         CCF                     ; ZERO CARRY FLAG
0562' ED 52       00767         SBC     HL,DE           ; DISPLACEMENT OF DIFF
                  00768 ;
0564' CB 25       00769         SLA     L
0566' CB 25       00770         SLA     L
0568' CB 25       00771         SLA     L               ; MULTIPLY BY 8
056A' 11 0002'    00772         LD      DE,SCRLOC       ; ADDRESS OF TABLE
056D' 19          00773         ADD     HL,DE           ; FIND ADDR IN TABLE
                  00774 ;
056E' 06 04       00775         LD      B,4             ; LOOP COUNT
0570' 5E          00776 L2:     LD      E,(HL)
0571' 23          00777         INC     HL
0572' 56          00778         LD      D,(HL)
0573' 23          00779         INC     HL
0574' EB          00780         EX      DE,HL           ; HL=(TABLE ENTRY)
0575' CB FE       00781         SET     7,(HL)          ; SET REV. VIDEO BIT
0577' EB          00782         EX      DE,HL
0578' 10 F6       00783         DJNZ    L2
                  00784 ;
057A' E1          00785         POP     HL
057B' D1          00786         POP     DE
057C' C1          00787         POP     BC
057D' F1          00788         POP     AF
057E' C9          00789         RET
                  00790 ;
                  00791 ; SAVE SAVES THE TOP FOUR STACK WORDS (2 BYTES EACH),
                  00792 ; AND THE 32 DISPLAYED MEMORY VALUES.  THESE VALUES
```

```
                        00793 ; ARE STORED IN SAVIT.
                        00794 ;
057F' F5                00795 SAVE.   PUSH   AF
0580' C5                00796         PUSH   BC
0581' D5                00797         PUSH   DE
0582' E5                00798         PUSH   HL
                        00799 ;
0583' 3E 04             00800         LD     A,4              ; OUTER COUNTER
0585' 21 FB09           00801         LD     HL,STKLOC+7      ; FROM V-RAM .
0588' 11 05BC           00802         LD     DE,SAVIT         ; TO SAVIT
058B' 01 0004           00803 L3:     LD     BC,4             ; MOVE TOP 4 OF STACK
058E' ED B0             00804         LDIR
0590' 23                00805         INC    HL               ; SKIP SPACE
0591' 3D                00806         DEC    A                ; DEC COUNTER
0592' 20 F7             00807         JR     NZ,L3
                        00808 ;
0594' 21 FBF8           00809         LD     HL,MEMLOC+56     ; START ADDRESS
0597' 3E 04             00810         LD     A,4              ; OUTER COUNTER
0599' 01 0010           00811 N7:     LD     BC,10
059C' ED A0             00812 N7A:    LDI
059E' ED A0             00813         LDI
05A0' 23                00814         INC    HL               ; NEXT BYTE
05A1' EA 059C'          00815         JP     PE,N7A
                        00816 ;
05A4' D5                00817         PUSH   DE
05A5' 11 0038           00818         LD     DE,38
05A8' 19                00819         ADD    HL,DE            ; POINT TO NEXT LINE
05A9' D1                00820         POP    DE
                        00821 ;
05AA' 3D                00822         DEC    A
05AB' 20 EC             00823         JR     NZ,N7            ; NEXT LINE
                        00824 ;
05AD' 21 05BC'          00825         LD     HL,SAVIT
05B0' 06 50             00826         LD     B,50
05B2' CB BE             00827 FIXIT:  RES    7,(HL)           ; RESET VIDEO BIT
05B4' 23                00828         INC    HL
05B5' 10 FB             00829         DJNZ   FIXIT
                        00830 ;
05B7' E1                00831         POP    HL
05B8' D1                00832         POP    DE
05B9' C1                00833         POP    BC
05BA' F1                00834         POP    AF
05BB' C9                00835         RET
                        00836 ;
05BC'                   00837 SAVIT:  DEFS   50
                        00838 ;
                        00839 ; REVMEM COMPARES THE NEW MEMORY DISPLAY WITH THE OLD
                        00840 ; ONE.  THE BYTES WHICH DIFFER ARE DISPLAYED IN REVERS
                        00841 ; VIDEO.  PLEASE NOTE THAT WHEN THE MEMORY DISPLAY
                        00842 ; ADDRESS IS CHANGED THE BYTES IN THE SAME RELATIVE
                        00843 ; SCREEN POSITION ARE COMPARED FOR DIFFERENCES.  THIS
                        00844 ; ALLOWS YOU TO COMPARE BLOCKS OF MEMORY FROM THE
                        00845 ; KEYBOARD WITH THE DIFFERENCES BEING HIGHLIGHTED.
                        00846 ;
060C' F5                00847 REVMEM: PUSH   AF
060D' C5                00848         PUSH   BC
```

```
060E' D5          00849          PUSH    DE
060F' E5          00850          PUSH    HL
                  00851
0610' FD 21       00852          LD      IY,00           ; DISPLACEMENT COUNT
0612' 0000
0614' 3E 04       00853          LD      A,4
0616' 32 0675'    00854          LD      (CNT),A         ; COUNTER
0619' 21 FB0F     00855          LD      HL,STKLOC+7     ; START ADDRESS
061C' DD 21       00856          LD      IX,STKLOC+7     ; SAVE IT
061E' FB0F
0620' 11 05BC'    00857          LD      DE,SAVIT        ; OTHER ONE
                  00858 ;
0623' 01 0004     00859 L5:      LD      BC,4            ; COMPARE FIRST WORD
0626' 1A          00860 L4:      LD      A,(DE)
0627' ED A1       00861          CPI                     ; COMPARE 'EM
0629' 13          00862          INC     DE
062A' 20 EE       00863          JR      NZ,CHANG4       ; JUMP IF DIFF
062C' FD 23       00864 R1:      INC     IY              ; INC DISPLACEMENT
062E' EA 0626'    00865          JP      PE,L4
                  00866 ;
0631' 23          00867          INC     HL
0632' 3A 0675'    00868          LD      A,(CNT)
0635' 3D          00869          DEC     A               ; DEC COUNTER
0636' 32 0675'    00870          LD      (CNT),A
0639' 20 E8       00871          JR      NZ,L5           ; CHECK NEXT WORD
                  00872 ;
063B' 3E 04       00873          LD      A,4
063D' 32 0675'    00874          LD      (CNT),A
                  00875 ;
0640' 21 FBF3     00876          LD      HL,MEMLOC+56    ; MEM DISPLAY ADDR
0643' 01 0010     00877 L8:      LD      BC,10
0646' E5          00878          PUSH    HL
0647' DD E1       00879          POP     IX              ; START OF LINE
0649' FD 21       00880          LD      IY,00           ; DISPLACEMENT COUNT
064B' 0000
064D' 1A          00881 L7:      LD      A,(DE)
064E' ED A1       00882          CPI                     ; CHECK FIRST BYTE
0650' 20 24       00883          JR      NZ,CHANGA       ; JUMP IF DIFF
                  00884 ;
0652' FD 23       00885          INC     IY              ; INC DISPLACEMENT
0654' 13          00886          INC     DE
0655' 1A          00887          LD      A,(DE)
0656' ED A1       00888          CPI                     ; CHECK 2ND BYTE
0658' 20 21       00889          JR      NZ,CHANG2       ; JUMP IF DIFF
                  00890 ;
065A' 13          00891 R2:      INC     DE
065B' 23          00892          INC     HL              ; SKIP SPACE
065C' FD 23       00893          INC     IY              ; INC COUNTER
065E' EA 064D'    00894          JP      PE,L7           ; NEXT BYTE
                  00895 ;
0661' D5          00896          PUSH    DE
0662' 11 0038     00897          LD      DE,38
0665' 19          00898          ADD     HL,DE           ; POINT TO NEXT LINE
0666' D1          00899          POP     DE
                  00900 ;
0667' 3A 0675'    00901          LD      A,(CNT)
```

```
066A' 3D          00902          DEC     A                ; DEC OUTER COUNTER
066B' 32 0675'    00903          LD      (CNT),A
066E' 20 D3       00904          JR      NZ,L8
                  00905 ;
0670' E1          00906          POP     HL
0671' D1          00907          POP     DE
0672' C1          00908          POP     BC
0673' F1          00909          POP     AF
0674' C9          00910          RET
                  00911 ;
0675'             00912 CNT:     DEFS    1
                  00913 ;
                  00914 ; THIS ROUTINE CHANGES THE V-RAM BYTES TO REVERSE
                  00915 ; VIDEO.
                  00916 ;
0676' 13          00917 CHANZA:  INC     DE
0677' FD 23       00918          INC     IY
0679' ED A1       00919          CPI             , DUMMY STATEMENTS
                  00920 ;
067B' F5          00921 CHANG2:  PUSH    AF
067C' C5          00922          PUSH    BC
067D' CB F8       00923          SET     7,B              ; FLAG
067F' 18 02       00924          JR      N12
                  00925 ;
0681' F5          00926 CHANG4:  PUSH    AF
0682' C5          00927          PUSH    BC
0683' D5          00928 N12:     PUSH    DE
0684' E5          00929          PUSH    HL
                  00930 ;
0685' FD E5       00931          PUSH    IY
0687' E1          00932          POP     HL               ; HL IS DISPLACEMENT
                  00933 ;
0688' CB 3D       00934          SRL     L                ; DIVIDE BY 2
068A' CB 78       00935          BIT     7,B              ; CHECK FLAG
068C' 20 02       00936          JR      NZ,N8
068E' CB 3D       00937          SRL     L                ; DIVIDE BY 2
0690' 7D          00938 N8:      LD      A,L              ; STORE # SPACES
0691' CB 25       00939          SLA     L                ; MULTIPLY BY 2
0693' CB 78       00940          BIT     7,B              ; CHECK FLAG
0695' 20 02       00941          JR      NZ,N9
0697' CB 25       00942          SLA     L                ; MULT BY 2
0699' 85          00943 N9:      ADD     A,L              ; ADD IN SPACES
069A' 6F          00944          LD      L,A
069B' DD E5       00945          PUSH    IX               ; GET START ADDRESS
069D' D1          00946          POP     DE
069E' 19          00947          ADD     HL,DE            ; ADDRESS IN V-RAM
069F' CB 78       00948          BIT     7,B
06A1' 20 0E       00949          JR      NZ,N10
                  00950 ;
06A3' 06 04       00951          LD      B,4
06A5' CB FE       00952 L6:      SET     7,(HL)           ; REVERSE IT
06A7' 23          00953          INC     HL
06A8' 10 FB       00954          DJNZ    L6
                  00955 ;
06AA' E1          00956          POP     HL
06AB' D1          00957          POP     DE
```

```
06AC' C1          00958          POP     BC
06AD' F1          00959          POP     AF
06AE' C3 062C'    00960          JP      R1
                  00961 ;
06B1' 06 02       00962 N10:     LD      B,2                  REVERSE IT
06B3' CB FE       00963 N11:     SET     7,(HL)
06B5' 23          00964          INC     HL
06B6' 10 FB       00965          DJNZ    N11
                  00966 ;
06B8' E1          00967          POP     HL
06B9' D1          00968          POP     DE
06BA' C1          00969          POP     BC
06BB' F1          00970          POP     AF
06BC' 18 9C       00971          JR      R2
                  00972 ;
                  00973 ; CICO READS CHARACTERS FROM THE KEYBOARD (IN POLLED
                  00974 ; FASHION) AND STORES THEM IN KEYIN.  THIS ACTION IS
                  00975 ; TERMINATED UPON RECEIVING A <CR> OF THE 25TH
                  00976 ; CHARACTER (WHICH CAUSES AN INPUT BUFFER OVERFLOW).
                  00977 ; THE CHARACTERS INPUTTED ARE ECHOED ON THE SCREEN
                  00978 ; AT THE LOCATION POINTED TO BY HL.
                  00979 ;
06BE' F5          00980 CICO:    PUSH    AF
06BF' C5          00981          PUSH    BC
06C0' D5          00982          PUSH    DE
06C1' E5          00983          PUSH    HL
06C2' DD E5       00984          PUSH    IX
                  00985 ;
                  00986 ;
06C4' DD 21       00987          LD      IX,KEYIN
06C6' 078F'
06C8' 1E 1A       00988          LD      E,1A                 ; OVERFLOW COUNTER
                  00989 ;
06CA' DB 14       00990 CICO1:   IN      A,(POLL)             ; READ STATUS
06CC' CB 6F       00991          BIT     5,A
06CE' 20 FA       00992          JR      NZ,CICO1             ; IF SO CHECK AGAIN
06D0' DB 0E       00993          IN      A,(KYBD)             ; GET INPUT
06D2' FE 08       00994          CP      BS                   ; BACKSPACE?
06D4' 28 29       00995          JR      Z,BCKUP              ; IF SO JUMP
                  00996 ;
06D6' FE 0D       00997 CICO2:   CP      CR                   ; <CR>?
06D8' CA 077E'    00998          JP      Z,DONE               ; IF SO DONE
06DB' 47          00999          LD      B,A                  ; SAVE INPUT
06DC' 1D          01000          DEC     E                    ; DEC OVERFLOW COUNT
06DD' 28 53       01001          JR      Z,OVER               ; JUMP IF OVERFLOW
                  01002 ;
06DF' C5          01003          PUSH    BC
06E0' E5          01004          PUSH    HL
06E1' 3A 00DD'    01005          LD      A,(CURSES+3)
06E4' 3C          01006          INC     A                    ; INC CURSOR POSITION
06E5' 32 00DD'    01007          LD      (CURSES+3),A
06E8' 20 07       01008          JR      NZ,N34
                  01009 ;
06EA' 3A 00DB'    01010          LD      A,(CURSES+1)
06ED' 3C          01011          INC     A                    ; INC CURSOR POSITION
06EE' 32 00DB'    01012          LD      (CURSES+1),A         ; (HIGH BYTE)
```

```
                        01013 ;
06F1' CD 00CD'          01014 N34:      CALL    CURSOR
06F4' E1                01015           POP     HL
06F5' C1                01016           POP     BC
                        01017 ;
06F6' DD 70 00          01018           LD      (IX),B          ; PUT IN KEYIN
06F9' DD 23             01019           INC     IX
06FB' 70                01020           LD      (HL),B          ; PUT ON SCREEN
06FC' 23                01021           INC     HL
06FD' 18 CB             01022           JR      CIC01
                        01023 ;
06FF' E5                01024 BCKUP:    PUSH    HL
0700' C5                01025           PUSH    BC
0701' AF                01026           XOR     A               ; ZERO CARRY FLAG
0702' DD E5             01027           PUSH    IX              ; COPY IX TO BC
0704' C1                01028           POP     BC
0705' 21 079F'          01029           LD      HL,KEYIN        ; START ADDRESS
0708' ED 42             01030           SBC     HL,BC           ; CHECK FOR UNDERFLOW
070A' 20 04             01031           JR      NZ,BCK1         ; CONT.  IF NZ
                        01032 ;
070C' C1                01033           POP     BC
070D' E1                01034           POP     HL
070E' 18 BA             01035           JR      CIC01           ; RETURN
                        01036 ;
0710' 3A 00DD'          01037 BCK1:     LD      A,(CURSES+3)
0713' FE 00             01038           CP      0               ; SET FLAGS
0715' 20 09             01039           JR      NZ,N36
                        01040 ;
0717' 47                01041           LD      B,A
0718' 3A 00DB'          01042           LD      A,(CURSES+1)
071B' 3D                01043           DEC     A               ; DEC HIGH BYTE
071C' 32 00DB'          01044           LD      (CURSES+1),A
071F' 78                01045           LD      A,B
                        01046 ;
0720' 3D                01047 N36:      DEC     A               ; BACK-UP CURSOR
0721' 32 00DD'          01048           LD      (CURSES+3),A
0724' CD 00CB'          01049           CALL    CURSOR
0727' C1                01050           POP     BC
0728' E1                01051           POP     HL
0729' 3E 20             01052           LD      A,BLANK
072B' 77                01053           LD      (HL),A          ; BLANK-OUT CHAR
072C' DD 2B             01054           DEC     IX              ; BACK-UP
072E' 2B                01055           DEC     HL              ; POINTERS
072F' 1C                01056           INC     E               ; INC COUNTER
0730' 18 98             01057           JR      CIC01
                        01058 ;
0732' DD E1             01059 OVER:     POP     IX
0734' E1                01060           POP     HL
0735' E5                01061           PUSH    HL              ; START OF TEXT
0736' DD E5             01062           PUSH    IX
0738' 11 07AF'          01063           LD      DE,MESS1        ; MESSAGE ADDRESS
073B' EB                01064           EX      DE,HL
073C' 01 0019           01065           LD      BC,19           ; LENGTH
073F' ED B0             01066           LDIR                    ; WRITE IT
0741' EB                01067           EX      DE,HL
                        01068 ;
```

```
0742' A7          01069          AND     A                ; ZERO CARRY FLAG
0743' 11 0019     01070          LD      DE,19
0746' ED 52       01071          SBC     HL,DE            ; RESET SCREEN ADDRESS
0748' DD 21       01072          LD      IX,KEYIN         ; RESET BUFFER
074A' 078F'
                  01073 ;
074C' DB 14       01074 N31.     IN      A,(POLL)         ; READ STATUS
074E' CB 6F       01075          BIT     5,A
0750' 20 FA       01076          JR      NZ,N31           ; IF SO CHECK AGAIN
                  01077 ;
0752' 01 0019     01078          LD      BC,19            ; LENGTH
0755' 11 07C8'    01079          LD      DE,BLNK          ; BLANKS
0758' EB          01080          EX      DE,HL
0759' ED A0       01081 N32:     LDI                      ; PUT IT ON SCREEN
075B' 2B          01082          DEC     HL               ; NEXT BLANK
075C' EA 0759'    01083          JP      PE,N32           ; IF NOT DONE JUMP
075F' EB          01084          EX      DE,HL
                  01085 ;
0760' A7          01086          AND     A                ; ZERO CARRY FLAG
0761' 11 0019     01087          LD      DE,19
0764' ED 52       01088          SBC     HL,DE            ; RESET SCREEN ADDRESS
0766' 1E 1A       01089          LD      E,1A             ; RESET COUNTER
                  01090 ;
0768' 3A 00DD'    01091          LD      A,(CURSES+3)
076B' D6 19       01092          SUB     19
076D' 32 00DD'    01093          LD      (CURSES+3),A     ; RESET CURSOR
0770' 30 07       01094          JR      NC,N35
                  01095 ;
0772' 3A 00DB'    01096          LD      A,(CURSES+1)
0775' 3D          01097          DEC     A                ; RESET CURSOR (HIGH)
0776' 32 00DB'    01098          LD      (CURSES+1),A
                  01099 ;
0779' DB 0E       01100 N35.     IN      A,(KYBD)         ; GET INPUT
077B' C3 06D6'    01101          JP      CICO2
                  01102 ;
077E' 3E 32       01103 DONE.    LD      A,CURSL          ; PUT CURSOR
0780' 32 00DD'    01104          LD      (CURSES+3),A     ; ADDRESS BACK
                  01105 ;
0783' 3E 0D       01106          LD      A,CR
0785' DD 77 00    01107          LD      (IX),A           ; INSERT <CR>
                  01108 ;
0788' DD E1       01109          POP     IX
078A' E1          01110          POP     HL
078B' D1          01111          POP     DE
078C' C1          01112          POP     BC
078D' F1          01113          POP     AF
078E' C9          01114          RET
                  01115 ;
078F'             01116 KEYIN:   DEFS    20
07AF' 49 4E 50    01117 MES81:   DEFB    'INPUT OVERFLOW RE-ENTER  '
07B2' 55 54 20
07B5' 4F 56 45
07B8' 52 46 4C
07BB' 4F 57 20
07BE' 52 45 2D
07C1' 45 4E 54
```

```
07C4  45 52 20
07C7  20
07C8  20          01118 BLNK.    DEFB
                  01119 ;
                  01120 ;                                  \
                  01121 ;                              -=<. *:>=-
                  01122 ;                                  \
                  01123          END
```

MACROS:

SYMBOLS:

ACONV	04E2I'	ALOC	0500'	ANAJ1	04F4'	ANAJ2	04FC'	ASCII	042A'
ASK	0438'	BANKS	051E'	BANKST	0501I'	BANKSW	0529I'	BCK1	0710'
BCKUP	06FF'	BEFOR	F7B0	BITS	0322'	BLANK	0020	BLNK	07C8'
BLOOP	0442'	BS	0008	CHANZA	0676'	CHANG2	067B'	CHANG4	0681'
CICO	06BEI'	CICO1	06CA'	CICO2	06D6'	CNEXT	03A9'	CNT	0675'
CR	000D	CRSOUT	00D0'	CTRL	045E'	CURSES	00DAI'	CURSL	0082
CURSOR	00CBI'	DISABS	0130*	DLET	04DE'	DONE	077E'	F1	0365'
F1.5	0378'	F2	0345'	F3	0379'	FDON	039F'	FDONE	03A2'
FINTOP	0333I'	FIXIT	05B2'	FLOOP	0123'	FOROFM	0113'	GOBAK	0535'
HERE	0176'	HEX	011D*	INFO	0380*	INSIDE	03DC'	KEYIN	078FI'
KYBD	000E	L1	0549'	L2	0570'	L3	058B'	L4	0626'
L5	0623'	L6	06A5'	L7	064D'	L8	0643'	LENGTH	0118*
LINE	0162*	LOOP1	0168'	MDISP	0000I'	MEMLOC	FBA2	MESS1	07AF'
MOV4	0455'	N1	0276'	N10	06B1'	N11	06B3'	N12	0683'
N3	02E3'	N31	074CI'	N32	0759'	N34	06F1'	N35	0779'
N36	0720'	N4	02D3'	N7	0599'	N7A	059C'	N8	0690'
N9	0699'	NAMES	02F7'	NFLAG	0189'	NIL	FFFF	NONFR	044A'
NORMAL	014D'	NULL	0000	OFF	0534'	ONE	032A'	OPCODE	0121*
ORGEND	03FC*	OVER	0732'	PBITS	031E'	POLL	0014	PRESAV	0544*
PUT	032C'	R1	062C'	R16BIT	FA12	R2	065A'	REGPC	038E*
REGS	019C'	REGSAV	055D*	REGSP	0267*	REV	0558'	REVERS	0148'
REVID	053CI'	REVMEM	060CI'	RSTATE	0178I'	SAVE	057FI'	SAVIT	05BCI'
SCREEN	0092I'	SCRLOC	0002'	SCRTOP	F800	SETVID	0510'	SFLAGS	F845
SPOT	03C4'	SREG	01F0'	SREGS	F882	STKLOC	FB02	TEXTUP	00E0I'
TFOUND	010C'	TIN	012F'	TNEXT	00F8'	TXRET	0172'	TXTP	00ED'
VIDEO	00ED	VIDOFF	0012	VIDSET	051F'	VREG	0000	VWORD	0001
WIPE	00B7I'	WITHIN	03FA'	WNEXT	0400'	WRET	0427'		

NO FATAL ERROR(S)

```
                00001 ;
                00002 ;                                 -=(O)=-
                00003 ;-----------------------------------------------------
                00004 ;  DISASS IS A DISASSEMBLER SUBROUTINE.  IT WORKS ON
                00005 ;  ONE INSTRUCTION PER CALL.  IT EXPECTS THE ABSOLUTE
                00006 ;  ADDRESS OF THE INSTRUCTION TO BE PASSED IN THE HL
                00007 ;  REGISTER PAIR.
                00008 ;  THE DISASSEMBLED TEXT IS WRITTEN TO AN EXTERNALLY
                00009 ;  ACCESSIBLE BUFFER CALLED LINE.
                00010 ;  ON RETURN, HL WILL POINT TO THE NEXT INSTRUCTION.
                00011 ;       BY BILL MACLEOD          1980:X:30
                00012 ;       LAST UPDATED 08-08-81
                00013 ;
                00014 ;       THIS FILE HAS THE UTILITY SUBROUTINES
                00015 ;       ORIGINALLY IN DAS, AS WELL AS THE INFO
                00016 ;       ROUTINE.
                00017 ;-----------------------------------------------------
0010            00018 .RADIX  16
                00019         ENTRY   DISASS
                00020 ;
                00021         ENTRY   BYTE
                00022         ENTRY   HEX
                00023         ENTRY   INSRET
                00024         ENTRY   INSTR
                00025         ENTRY   LENGTH
                00026         ENTRY   LINE
                00027         ENTRY   OPCODE
                00028         ENTRY   OPRNDS
                00029 ;
                00030         EXTRN   BANKSW
                00031 ;
00ED            00032 VIDEO   EQU     0ED     ; SWITCH IN V-RAM CODE
001Z            00033 VIDOFF  EQU     1Z      ; SWITCH V-RAM OUT CODE
                00034 ,
                00035 ,
                00036 NEXPT   MACRO           ;NEXT POINT OF REF COUNTER
                00037         POP     HL
                00038         PUSH    HL
                00039         INC     HL
                00040         ENDM
                00041 PUT     MACRO   CHAR
                00042         LD      A,CHAR
                00043         LD      (DE),A
                00044         INC     DE
                00045         ENDM
                00046 ;                         -=(O)=-
                00047         PAGE
```

```
0000'                   00048 ;-------------------------------------------------
                        00049 ;                                                 -
                        00050 ;       DEFINITIONS AND DECLARATIONS              +
                        00051 ;                                                 -
                        00052 ;-------------------------------------------------
0000'                   00053 BYTE:   DEFS    1         ;SAVE THE INSTRUCTION HERE
0001'                   00054 LENGTH: DEFS    2         ;USABLE AS 1 OR 2 BYTES
                        00055 ;
0003'                   00056 LINE:   DEFS    1         ;OUTPUT LINE 50 CHARACTERS LONG
0004'                   00057 LOCN:   DEFS    4         ;FIRST FIELD IN "LINE"
0008'                   00058 SP1:    DEFS    2         ;
000A'                   00059 CODE:   DEFS    0C        ;
0016'                   00060 OPCODE: DEFS    8         ;
001E'                   00061 OPRNDS: DEFS    17        ;LAST FIELD IN "LINE"
                        00062 ;                         -=(0)=-
0035' 48 4C 2C          00063 HLMES:  DEFM    'HL,'
0038' 44 45 43          00064 DECM:   DEFM    'DEC '
003B' 20
003C' 49 4E 43          00065 INCM:   DEFM    'INC '
003F' 20
0040' 4E 4F 50          00066 NOPM:   DEFM    'NOP'
0043' 45 58 41          00067 EXM:    DEFM    'EXAF,AF'
0046' 46 2C 41
0049' 46
004A' 27                00068         DEFB    27        ;APOSTROPHE
004B' 44 4A 4E          00069 DJNZM:  DEFM    'DJNZ '
004E' 5A 20
0050' 4A 52 20          00070 JRSM:   DEFM    'JR '
0053' 52 52 43          00071 LTAB1:  DEFM    'RRCARRA CPL CCF '
0056' 41 52 52
0059' 41 20 43
005C' 50 4C 20
005F' 43 43 46
0062' 20
0063' 52 4C 43          00072 LTAB2:  DEFM    'RLCARLA DAA SCF '
0066' 41 52 4C
0069' 41 20 44
006C' 41 41 20
006F' 53 43 46
0072' 20
0073' 43 41 4C          00073 CALLM:  DEFM    "CALL"
0076' 4C
0077' 50 55 53          00074 PUSHM:  DEFM    "PUSH"
007A' 48
007B' 50 4F 50          00075 POPM:   DEFM    "POP "
007E' 20
007F' 52 45 54          00076 PLM:    DEFM    "RET EXX "
0082' 20 45 58
0085' 58 20
0087' 4A 50 20          00077 JPM:    DEFM    "JP ."
008A' 2E
008B' 4C 44 20          00078 LDM:    DEFM    "LD ."
008E' 2E
008F' 28 48 4C          00079 HLIND:  DEFM    "(HL)SP,HL."
```

```
0091' 2F 53 50
0095' 2C 48 4C
0098' 2E
0099' 52 53 54  00080 RSTM.    DEFM    "RST"
009C' 20 49     00081 INTERM:  DEFM    " I"
009E' 49 4E 20  00082 INM.     DEFM    "IN "
00A1' 4F 55 54  00083 OUTM.    DEFM    "OUT"
00A4' 53 50 2C  00084 SPHLM.   DEFM    "SP,(HL)"
00A7' 28 48 4C
00AA' 29
00AB' 44 45 2C  00085 DEHLM.   DEFM    "DE,HL"
00AE' 48 4C
00B0' 00        00086 DISPX:   DEFB    00
00B1' 49 2D     00087 XBUF:    DEFM    'I-'
00B3'           00088 DBUF:    DEFS    3
00B6' CB 00     00089 CBBUF:   DEFB    0CB,00
00B8' 42 49 54  00090 TWITAB:  DEFM    'BIT RES SET'
00BB' 20 52 45
00BE' 53 20 53
00C1' 45 54
00C3' 52 4C 43  00091 SHTAB.   DEFM    'RLC RRC RL  RR  SLA SRA ****SRL'
00C6' 20 52 52
00C9' 43 20 52
00CC' 4C 20 20
00CF' 52 52 20
00D2' 20 53 4C
00D5' 41 20 53
00D8' 52 41 20
00DB' 2A 2A 2A
00DE' 2A 53 52
00E1' 4C
00E2' 48 41 4C  00092 HLTM:    DEFM    'HALT'
00E5' 54
00E6'           00093 NBUF:    DEFS    4
00EA' 41 2C     00094 AMES:    DEFM    'A,'
00EC' 41 44 44  00095 LOGTAB:  DEFM    'ADD ADC SUB SBC AND XOR OR  CP
00EF' 20 41 44
00F2' 43 20 53
00F5' 55 42 20
00F8' 53 42 43
00FB' 20 41 4E
00FE' 44 20 58
0101' 4F 52 20
0104' 4F 52 20
0107' 20 43 50
010A' 20 20
010C' 42 43 44  00096 RNAMES:  DEFM    'BCDEHL'
010F' 45 48 4C
0112'           00097 RNAM:    DEFS    2       ;CAN BE "SP", "AF", OR "*A"
0114' 4E 5A 5A  00098 CONDXM.  DEFM    "NZZ.NCC.POPEP.M."
0117' 2E 4E 43
011A' 43 2E 50
011D' 4F 50 45
0120' 50 2E 4D
0123' 2E
                00099          PAGE
```

```
0124'
0124' F5          00100 DISASS:  PUSH    AF
0125' C5          00101          PUSH    BC
0126' 7E          00102          LD      A,(HL)
0127' 32 0000'    00103          LD      (BYTE),A
012A' CD 01CB'    00104          CALL    INFO
012D' E6 03       00105          AND     03
012F' 3C          00106          INC     A                ;LENGTH CAN BE USED AS
0130' 32 0001'    00107          LD      (LENGTH),A       ;EITHER EIGHT BITS
0133' AF          00108          XOR     A
0134' 32 0002'    00109          LD      (LENGTH+1),A     ;OR SIXTEEN BITS
0137' CD 0187'    00110          CALL    INSTR
013A' CD 0145'    00111          CALL    HEX
013D' ED 4B       00112          LD      BC,(LENGTH)
013F' 0001'
0141' 09          00113          ADD     HL,BC
0142' C1          00114          POP     BC
0143' F1          00115          POP     AF
0144' C9          00116          RET
                  00117 ;
                  00118 ;                        -=(D)=-
                  00119          PAGE
```

```
0145'
                    00120                               ;-------------------------------
0145' D5            00121 HEX:      PUSH    DE          ;THIS ROUTINE PUTS CHARACTERS
0146' C5            00122           PUSH    BC          ;INTO "LOCATION" AND "OBJECT"
0147' F5            00123           PUSH    AF          ;FIELDS OF THE OUTPUT LINE.
0148' DD E5         00124           PUSH    IX          ;-------------------------------
014A' E5            00125           PUSH    HL          ;SAVE CALLERS REGISTERS
                    00126 ;
014B' 3E 12         00127           LD      A,VIDOFF            ; SWITCH OUT V-RAM
014D' CD 0000*      00128           CALL    BANKSW
                    00129 ;
0150' 7C            00130           LD      A,H
0151' CD 072D'      00131           CALL    ACONVI
0154' ED 53         00132           LD      (LOCN),DE           ;PUT ADDR IN LOCN FIELD
0156' 0004'
0158' 7D            00133           LD      A,L
0159' CD 072D'      00134           CALL    ACONVI
015C' ED 53         00135           LD      (LOCN+2),DE
015E' 0006'
0160' DD 21         00136           LD      IX,CODE             ;OBJECT FIELD
0162' 000A'
0164' 3A 0001'      00137           LD      A,(LENGTH)
0167' 47            00138           LD      B,A
0168' 7E            00139 CORE:     LD      A,(HL)              ;WRITE OBJECT CODE
016F' CD 072D'      00140           CALL    ACONVI
016C' DD 73 00      00141           LD      (IX),E
016F' DD 72 01      00142           LD      (IX+1),D
0172' 23            00143           INC     HL
0173' DD 23         00144           INC     IX
0175' DD 23         00145           INC     IX
0177' DD 23         00146           INC     IX
0179' 10 EE         00147           DJNZ    CORE        ;COUNTING INSTRUCTION BYTES
                    00148 ;
017B' 3E E1         00149           LD      A,VIDEO             ; SWITCH V-RAM IN
017D' CD 0000*      00150           CALL    BANKSW
                    00151 ;
0180' E1            00152           POP     HL
0181' DD E1         00153           POP     IX
0183' F1            00154           POP     AF          ;RESTORE CALLERS REGISTERS
0184' C1            00155           POP     BC
0185' D1            00156           POP     DE
0186' C9            00157           RET                 ;AND RETURN
                    00158 ;
                    00159 ;                             -=(O)=-
                    00160           PAGE
```

```
0187'
                    00161                           ;--------------------------------
0187' DE            00162 INSTR:    PUSH    DE      ;EXPECTS HL TO POINT AT
0188' C5            00163           PUSH    BC      ;INSTRUCTION IN CORE.
0189' F5            00164           PUSH    AF      ;WRITES TO "SOURCE" FIELD
018A' E5            00165           PUSH    HL      ;OF OUTPUT LINE.
                    00166                           ;--------------------------------
018B' 21 0003'      00167           LD      HL,LINE
018E' 36 20         00168           LD      (HL),20         ;FIRST BLANK OUT
0190' 11 0004'      00169           LD      DE,LINE+1       ;OUTPUT LINE
0193' 01 0031       00170           LD      BC,31
0196' ED B0         00171           LDIR
0198' 11 0016'      00172           LD      DE,OPCODE       ;POINT AT SOURCE FIELD
                    00173 ;
019B' 3A 0000'      00174           LD      A,(BYTE)
019E' FE ED         00175           CP      0ED
01A0' CA 0910'      00176           JP      Z,SPECL
01A3' FE CB         00177           CP      0CB
01A5' CA 05E2'      00178           JP      Z,TWIDL
01A8' FE DD         00179           CP      0DD
01AA' CA 0544'      00180           JP      Z,INDEXD
01AD' FE FD         00181           CP      0FD
01AF' CA 0544'      00182           JP      Z,INDEXD
                    00183 ;
                    00184 ;         FALL THROUGH TO THE GARDEN VARIETY INSTR-
                    00185 ;         UCTIONS.  THESE ARE THE ONES WE CAN IDENTIFY
                    00186 ;         BY LOOKING AT ONLY THE FIRST BYTE.
                    00187           PAGE
```

```
01B2'
                    00188 ;         HERE WE DEAL WITH GARDEN VARIETY INSTRUCTIONS
                    00189 ;
01B2' E6 C0         00190           AND     0C0         ;CLASSIFY BY 1ST 2 BITS
01B4' FE 00         00191           CP      00
01B6' CA 039E'      00192           JP      Z,ARITH
01B9' FE 40         00193           CP      40
01BB' CA 0633'      00194           JP      Z,LOADS
01BE' FE 80         00195           CP      80
01C0' CA 0660'      00196           JP      Z,LOGIC
01C3' C3 01F2'      00197           JP      HIQTR       ;GO DO HIGH QUARTER OF MAP
                    00198 ;
                    00199 ;                             -=(O)=-
01C6' E1            00200 INSRET:   POP     HL          ;JUMP HERE FOR UNIFORM RETURN
01C7' F1            00201           POP     AF
01C8' C1            00202           POP     BC
01C9' D1            00203           POP     DE
01CA' C9            00204           RET
                    00205 ;
                    00206 ;                                 \
                    00207 ;                             -=(<*>)=-
                    00208 ;                                 \
                    00209           PAGE
```

```
01CB'
                    00210 ;         INFO ROUTINE  80:VII:14
                    00211 ;         RETURNS A BYTE OF INFORMATION
                    00212 ;         ABOUT THE INSTRUCTION
                    00213 ;         POINTED TO BY THE HL REGISTER PAIR.
                    00214 ;         THE INFORMATION BYTE
                    00215 ;         IS RETURNED IN THE A REGISTER.
                    00216 ;
                    00217 ;         ****      BITS RETURNED:          ****
                    00218 ;
                    00219 ;                 BIT 7   :   I/O FLAG
                    00220 ;                 BIT 6   :  AFFECTS F REGISTER
                    00221 ;                 BIT 5   :  CONDITIONAL INSTRUCTION
                    00222 ;                 BIT 4   :  CHANGES PROGRAM COUNTER
                    00223 ;                 BIT 3   :  CHANGES SOME REGISTER
                    00224 ;                 BIT 2   :  CHANGES MEMORY
                    00225 ;                 BIT 1   :  HIGH BIT OF DIMINISHED LENGTH
                    00226 ;                 BIT 0   :  LOW BIT OF DIMINISHED LENGTH
                    00227 ;                                    (LENGTH-1)
                    00228 ;         ****                       ****
                    00229 ;
                    00230           ENTRY   INFO
0010                00231 .RADIX    16
01CB' 3E 12         00232 INFO:     LD      A,VIDOFF        ; SWITCH OUT V-RAM
01CD' CD 0000*      00233           CALL    BANKSW
                    00234 ;
01D0' 7E            00235           LD      A,(HL)
01D1' E3            00236           PUSH    HL
01D2' FE CB         00237           CP      0CB     ;SEE IF IT'S A BIT TWIDDLE
01D4' CA 078C'      00238           JP      Z,TWIDDL
01D7' FE DD         00239           CP      0DD             ;INDEXED?
01D9' CA 07C0'      00240           JP      Z,NDEXD
01DC' FE FD         00241           CP      0FD
01DE' CA 07C0'      00242           JP      Z,NDEXD
01E1' FE ED         00243           CP      0ED
01E3' CA 07EE'      00244           JP      Z,SPCIAL
                    00245 ;         ***  TO GET HERE WE ARE POINTING AT     ***
                    00246 ;         ***  A GARDEN VARIETY INSTRUCTION       ***
01E6' CD 0821'      00247           CALL    LOOK
01E9' E1            00248 BACK:     POP     HL
                    00249 ;
01EA' F5            00250           PUSH    AF              ; SAVE A
01EB' 3E ED         00251           LD      A,VIDEO         ; SWITCH V-RAM IN
01ED' CD 0000*      00252           CALL    BANKSW
01F0' F1            00253           POP     AF
                    00254 ;
01F1' C9            00255           RET
                    00256 $EJECT
```

```
01F2'
01F2' 3A 0000'  00257 HIGTR:  LD      A,(BYTE)        ;HERE DO HIGH QUARTER
01F5' FE CD     00258         CP      0CD             ;OF INSTRUCTION SET.
01F7' 28 7B     00259         JR      Z,CALLO         ; CO <= 1ST BYTE <= FF
01F9' E6 03     00260         AND     03      ;CLASSIFY BY LOW 2 BITS
01FB' FE 00     00261         CP      00
01FD' 28 4F     00262         JR      Z,CALRET
01FF' FE 01     00263         CP      01
0201' CA 0282'  00264         JP      Z,PDPUSH
0204' FE 03     00265         CP      03
0206' CA 02DC'  00266         JP      Z,RSTJNK
                00267 ;       FALL THROUGH TO IMMEDIATE ARITHMETIC & JPS
0209' 3A 0000'  00268         LD      A,(BYTE)
020C' CB 57     00269         BIT     2,A
020E' 28 26     00270         JR      Z,JPS
                00271 ;       FELL THRU TO IMMEDIATE ARITHMETIC
0210' CB B7     00272         RES     6,A
0212' 32 0000'  00273         LD      (BYTE),A
0215' CD 0187'  00274         CALL    INSTR
0218' 3E 28     00275         LD      A,'('
021A' 21 001E'  00276         LD      HL,OPRNDS
021D' 01 0017   00277         LD      BC,17
0220' ED B1     00278         CPIR
0222' C2 01C6'  00279         JP      NZ,INSRET       ;ERROR RETURN, NO MATCH:
0225' 2B        00280         DEC     HL
0226' E5        00281         PUSH    HL
0227' D1        00282         POP     DE
                00283         NEXPT
0228' E1                      POP     HL
0229' E5                      PUSH    HL
022A' 23                      INC     HL
022B' CD 06EA'  00284         CALL    NUMB8
022E' 3E 20     00285         LD      A,' '
0230' 12        00286         LD      (DE),A
0231' 13        00287         INC     DE
0232' 12        00288         LD      (DE),A
0233' C3 01C6'  00289         JP      INSRET
                00290 ;
                00291 ;                         -=(O)=-
0236' 21 0087'  00292 JPS:    LD      HL,JPM
0239' 01 0002   00293         LD      BC,2
023C' ED B0     00294         LDIR
023E' CD 0766'  00295         CALL    CONDX
                00296         PUT     ','
0241' 3E 2C                   LD      A,','
0243' 12                      LD      (DE),A
0244' 13                      INC     DE
                00297 TARG:   NEXPT           ;FIND TARGET OF CALL OR JP
0245' E1                      POP     HL
0246' E5                      PUSH    HL
0247' 23                      INC     HL
0248' CD 06C8'  00298         CALL    NUMB16
024B' C3 01C6'  00299         JP      INSRET
                00300 ;
                00301 ;                         -=(O)=-
```

```
024E' 3A 0000'  00302 CALRET: LD      A,(BYTE)          ;DO CALLS & RETS.
0251' CB 57     00303         BIT     2,A
0253' 28 11     00304         JR      Z,RETS
0255' 21 0073'  00305         LD      HL,CALLM
0258' 01 0004   00306         LD      BC,4
025B' ED B0     00307         LDIR
025D' CD 0766'  00308         CALL    CONDX
                00309         PUT     ','
0260' 3E 2C                   LD      A,','
0262' 12                      LD      (DE),A
0263' 13                      INC     DE
0264' 18 DF     00310         JR      TARG
                00311 ;
0266' 21 007F'  00312 RETS:   LD      HL,PLM
0269' 01 0003   00313         LD      BC,3
026C' ED B0     00314         LDIR
026E' CD 0766'  00315         CALL    CONDX
0271' C3 01C6'  00316         JP      INSRET
                00317 ;
0274' 21 0073'  00318 CALLO:  LD      HL,CALLM
0277' 01 0004   00319         LD      BC,4
027A' ED B0     00320         LDIR
027C' 11 001E'  00321         LD      DE,OPRNDS
027F' C3 0248'  00322         JP      TARG
                00323 ;
                00324 ;                         -=(*)=-
0282' 3A 0000'  00325 POPUSH: LD      A,(BYTE)
0285' CB 5F     00326         BIT     3,A
0287' 20 29     00327         JR      NZ,PLOOK          ;GO LOOK IN TABLE.
0289' CB 57     00328         BIT     2,A
028B' 28 05     00329         JR      Z,POPS
028D' 21 0077'  00330         LD      HL,PUSHM
0290' 18 03     00331         JR      PUTS
0292' 21 007B'  00332 POPS.   LD      HL,POPM
0295' 01 0004   00333 PUTS.   LD      BC,4
0298' ED B0     00334         LDIR
029A' 11 001E'  00335         LD      DE,OPRNDS
029D' E6 30     00336         AND     30
029F' 06 03     00337         LD      B,3               ;SHIFT COUNT
02A1' 0E 02     00338         LD      C,2               ;CHARACTER COUNT
02A3' 21 4641   00339         LD      HL,'FA'
02A6' 22 0112'  00340         LD      (RNAM),HL
02A9' 21 010C'  00341         LD      HL,RNAMES
02AC' CD 0782'  00342         CALL    LOOKUP
02AF' C3 01C6'  00343         JP      INSRET
                00344 ;
02B2' E6 30     00345 PLOOK:  AND     30
02B4' 1F        00346         RRA
02B5' 1F        00347         RRA
02B6' 06 00     00348         LD      B,0
02B8' 4F        00349         LD      C,A
02B9' 21 007F'  00350         LD      HL,PLM
02BC' 09        00351         ADD     HL,BC
02BD' 01 0003   00352         LD      BC,3
02C0' ED B0     00353         LDIR
02C2' 3E 2E     00354         LD      A,'.'
```

```
02C4' BE          00358          CP      (HL)
02C5' C2 01C6'    00359          JP      NZ,INSRET
02C9' 11 001E'    00360          LD      DE,OPRNDS
02CB' 01 0005     00361          LD      BC,5
02CE' 09          00362          ADD     HL,BC
02CF' ED B0       00363          LDIR
02D1' BE          00364          CP      (HL)
02D2' CA 01C6'    00365          JP      Z,INSRET
02D5' 3E 20       00366          LD      A,' '
02D7' 1B          00367          DEC     DE
02D8' 12          00368          LD      (DE),A
02D9' C3 01C6'    00369          JP      INSRET
02DC' 3A 0000'    00370 RSTJNK:  LD      A,(BYTE)        ;RESETS & OTHER JUNK
02DF' CB 57       00371          BIT     2,A
02E1' 28 19       00372          JR      Z,JNK           ;OTHER JUNK
02E3' 21 0099'    00373          LD      HL,RSTM
02E6' 01 0003     00374          LD      BC,3
02E9' ED B0       00375          LDIR
02EB' 11 001E'    00376          LD      DE,OPRNDS
02EE' E6 38       00377          AND     38
02F0' 32 00B3'    00378          LD      (DBUF),A
02F3' 21 00B3'    00379          LD      HL,DBUF
02F6' CD 04E9'    00380          CALL    NUMB8
02F9' C3 01C6'    00381          JP      INSRET
                  00382 ;
02FC' FE F0       00383 JNK:     CP      0F0
02FE' 38 15       00384          JR      C,RSOPND        ;SOME HAVE OPERANDS.
0300' 21 0090'    00385          LD      HL,INTERM
0303' CB 5F       00386          BIT     3,A
0305' 28 04       00387          JR      Z,DISABL
0307' 36 45       00388          LD      (HL),'E'
0309' 18 02       00389          JR      INTEWR
030B' 36 44       00390 DISABL:  LD      (HL),'D'
030D' 01 0002     00391 INTEWR:  LD      BC,2
0310' ED B0       00392          LDIR
0312' C3 01C6'    00393          JP      INSRET
0315' FE C3       00394 RSOPND:  CP      0C3     ;IS IT = JUMP?
0317' 20 14       00395          JR      NZ,NOTJP
0319' 21 0087'    00396          LD      HL,JPM
031C' 01 0002     00397          LD      BC,2
031F' ED B0       00398          LDIR
0321' 11 001E'    00399          LD      DE,OPRNDS
                  00400          NEXPT
0324' E1                         POP     HL
0325' E5                         PUSH    HL
0326' 23                         INC     HL
0327' CD 06C8'    00401          CALL    NUMB16
032A' C3 01C6'    00402          JP      INSRET
032D' CB 67       00403 NOTJP    BIT     4,A
032F' 28 49       00404          JR      Z,EXS
0331' CB 5F       00405          BIT     3,A
0333' 28 05       00406          JR      Z,OUTW
0335' 21 009E'    00407          LD      HL,INM
0338' 18 03       00408          JR      INOUT
033A' 21 00A1'    00409 OUTW:    LD      HL,OUTM
033D' 01 0003     00410 INOUT:   LD      BC,3
```

```
0340' ED B0      00408           LDIR
0342' 11 001E'   00409           LD      DE,OPRNDS
0345' CB 7F      00410           BIT     3,A
0347' 20 13      00411           JR      NZ,OUTO
                 00412           PUT     '('
0349' 3E 28                      LD      A,'('
034B' 12                         LD      (DE),A
034C' 13                         INC     DE
                 00413           NEXPT
034D' E1                         POP     HL
034E' E5                         PUSH    HL
034F' 23                         INC     HL
0350' CD 06EA'   00414           CALL    NUMBS
                 00415           PUT     ')'
0353' 3E 29                      LD      A,')'
0355' 12                         LD      (DE),A
0356' 13                         INC     DE
                 00416           PUT     ','
0357' 3E 2C                      LD      A,','
0359' 12                         LD      (DE),A
035A' 13                         INC     DE
035B' 3E 41      00417           LD      A,'A'
035D' 12         00418           LD      (DE),A
035E' C3 01C6'   00419           JP      INSRET
0361' 3E 41      00420 OUTO:     LD      A,'A'
0363' 12         00421           LD      (DE),A
0364' 13         00422           INC     DE
                 00423           PUT     ','
0365' 3E 2C                      LD      A,','
0367' 12                         LD      (DE),A
0368' 13                         INC     DE
                 00424           PUT     '('
0369' 3E 28                      LD      A,'('
036B' 12                         LD      (DE),A
036C' 13                         INC     DE
                 00425           NEXPT
036D' E1                         POP     HL
036E' E5                         PUSH    HL
036F' 23                         INC     HL
0370' CD 06EA'   00426           CALL    NUMBS
                 00427           PUT     ')'
0373' 3E 29                      LD      A,')'
0375' 12                         LD      (DE),A
0376' 13                         INC     DE
0377' C3 01C6'   00428           JP      INSRET
037A' 11 0043'   00429 EXS:      LD      HL,EXM
037D' 01 0002    00430           LD      BC,2
0380' ED B0      00431           LDIR
0382' 11 001E'   00432           LD      DE,OPRNDS
0385' CB 3F      00433           BIT     3,A
0387' 28 0A      00434           JR      Z,SPHLI
0389' 21 00AB'   00435           LD      HL,DEHLM
038C' 01 0005    00436           LD      BC,5
038F' ED B0      00437           LDIR
0391' 18 08      00438           JR      EXRET
0393' 21 00A4'   00439 SPHLI:    LD      HL,SPHLM
```

```
03F6  01 0007   00440          LD      BC,7
03F9  ED B1     00441          LDIR
03FB  C3 01C6   00442 EXRET.   JP      INSRET
                00443 ;
03FE  3A 0000   00444 ARITH:   LD      A,(BYTE)        ;----------------------
0401  E6 0F     00445          AND     0F              ;FALL BETWEEN 00 & 3F.
0403  FE 01     00446          CP      01              ;THIS GROUP IS ONLY   .
0405  CA 044C   00447          JP      Z,LOAD16        ;SLIGHTLY REGULAR.
0408  FE 03     00448          CP      03
040A  CA 0467   00449          JP      Z,INC16
040D  FE 09     00450          CP      09
040F  CA 049C   00451          JP      Z,ADDHL
0412  FE 0B     00452          CP      0B
0414  CA 046C   00453          JP      Z,DEC16
0417  E6 07     00454          AND     07
0419  FE 07     00455          CP      07
041B  CA 047D   00456          JP      Z,LOOKA
041E  FE 05     00457          CP      05
0420  CA 04E5   00458          JP      Z,DEC8
0423  FE 04     00459          CP      04
0425  CA 04DA   00460          JP      Z,INC8
0428  FE 00     00461          CP      0
042A  CA 041E   00462          JP      Z,JRS
                00463 ;
                00464 ;        FALL THROUGH TO 8-BIT LOADS
                00465 ;        FELL THROUGH TO 8-BIT LOADS
042D  21 003B   00466 LD8:     LD      HL,LDM
0430  01 0002   00467          LD      BC,2
0433  ED B0     00468          LDIR
0435  11 001E   00469          LD      DE,OPRNDS       ;DEST IS OPERAND FIELD
0438  3A 0000   00470          LD      A,(BYTE)
043B  CB 57     00471          BIT     2,A
043D  28 10     00472          JR      Z,INDI8         ;FOR REGISTER INDIRECT
                00473 ;
043F  CD 06FF   00474          CALL    REG38           ;GET REGISTER NAME
                00475          PUT     ','             ;PUT A COMMA ON "LINE"
0444  3E 2C                    LD      A,','
0446  12                       LD      (DE),A
0447  13                       INC     DE
                00476          NEXPT                   ;(REF COUNTER + 1) INTO HL PAIR
044A  E1                       POP     HL
044B  E5                       PUSH    HL
044C  23                       INC     HL
044D  CD 04EA   00477          CALL    NUMB8           ;WRITE IMMED N TO "LINE"
0450  C3 01C6   00478          JP      INSRET          ;TIDY UP AND RETURN.
                00479 ;
                00480 ;
                00481 ;                                ;HERE DO REGISTER INDIRECT
0453  E6 F0     00482 INDI8:   AND     0F0             ;ACCUMULATOR OR HL PAIR?
0455  FE 20     00483          CP      20
0457  CA 0463   00484          JR      Z,HLM           ;GO WRITE "HL"
045A  3E 41                    LD      A,'A'
045C  12                       LD      (DE),A          ;WRITE "A"
045D  13                       INC     DE
045E  18 0E     00488          JR      COMM
0460  21 0039   00489 HLM.     LD      HL,HLMES
```

```
0340' ED B0      00408           LDIR
0342' 11 001E'   00409           LD      DE,OPRNDS
0345' CB 7F      00410           BIT     3,A
0347' 20 18      00411           JR      NZ,OUTO
                 00412           PUT     '('
0349' 3E 28                      LD      A,'('
034B' 12                         LD      (DE),A
034C' 13                         INC     DE
                 00413           NEXPT
034D' E1                         POP     HL
034E' E5                         PUSH    HL
034F' 23                         INC     HL
0350' CD 06EA'   00414           CALL    NUMBS
                 00415           PUT     ')'
0353' 3E 29                      LD      A,')'
0355' 12                         LD      (DE),A
0356' 13                         INC     DE
                 00416           PUT     ','
0357' 3E 2C                      LD      A,','
0359' 12                         LD      (DE),A
035A' 13                         INC     DE
035B' 3E 41      00417           LD      A,'A'
035D' 12         00418           LD      (DE),A
035E' C3 0116'   00419           JP      INSRET
0361' 3E 41      00420 OUTO:     LD      A,'A'
0363' 12         00421           LD      (DE),A
0364' 13         00422           INC     DE
                 00423           PUT     ','
0365' 3E 2C                      LD      A,','
0367' 12                         LD      (DE),A
0368' 13                         INC     DE
                 00424           PUT     '('
0369' 3E 28                      LD      A,'('
036B' 12                         LD      (DE),A
036C' 13                         INC     DE
                 00425           NEXPT
036D' E1                         POP     HL
036E' E5                         PUSH    HL
036F' 23                         INC     HL
0370' CD 06EA'   00426           CALL    NUMBS
                 00427           PUT     ')'
0373' 3E 29                      LD      A,')'
0375' 12                         LD      (DE),A
0376' 13                         INC     DE
0377' C3 0116'   00428           JP      INSRET
037A' 21 0043'   00429 EXS:      LD      HL,EXM
037D' 01 0002    00430           LD      BC,2
0380' ED B0      00431           LDIR
0382' 11 001E'   00432           LD      DE,OPRNDS
0385' CB 5F      00433           BIT     3,A
0387' 28 0A      00434           JR      Z,SPHLI
0389' 21 00AB'   00435           LD      HL,DEHLM
038C' 01 0005    00436           LD      BC,5
038F' ED B0      00437           LDIR
0391' 18 08      00438           JR      EXRET
0393' 21 00A4'   00439 SPHLI:    LD      HL,SPHLM
```

```
03FE  01 0002   00489          LD      BC,2
0401  ED B0      00491          LDIR                    ;WRITE "HL"
                00492 COMM:    PUT     ,       ;WRITE A COMMA
0403' 3E 2C                    LD      A,','
0405' 12                       LD      (DE),A
0406  13                       INC     DE
                00493          PUT     '('     ;WRITE A LEFT PARENTHESIS
0407' 3E 28                    LD      A,'('
0409' 12                       LD      (DE),A
040A' 13                       INC     DE
040B' 3A 0000'  00494          LD      A,(BYTE)        ;GET BACK INSTR BYTE
040E' CB 6F     00495          BIT     5,A     ;IMMEDIATE OR REGISTER PAIR?
0410' 28 08     00496          JR      Z,RG16
                00497          NEXPT           ;(REF COUNTER + 1) INTO HL PAIR:
0412' E1                       POP     HL
0413' E5                       PUSH    HL
0414' 23                       INC     HL
0415' CD 0603'  00498          CALL    NUMB16
0418' 18 03     00499          JR      WCHWAY
041A' CD 0698'  00500 RG16:    CALL    SS      ;GET NAME OF REGISTER PAIR
                00501 WCHWAY:  PUT     ')'     ;PUT A RIGHT PARENTHESIS
041D' 3E 29                    LD      A,')'
041F' 12                       LD      (DE),A
0420' 13                       INC     DE
0421' 3A 0000'  00502          LD      A,(BYTE)
0424' CB 5F     00503          BIT     3,A     ;WHICH WAY DOES THE LOAD GO?
0426' C2 0106'  00504          JP      NZ,INSRET       ;TIDY UP & RETURN
0429' 3E 2C     00505          LD      A,','   ;APPEND A COMMA
042B' 12        00506          LD      (DE),A
042C' 13        00507          INC     DE
042D' 21 001E'  00508          LD      HL,OPRNDS
0430' ED A0     00509 LTRLD:   LDI     ;LOAD A LETTER TO "LINE"
0432' BE        00510          CP      (HL)            ;NEXT CHAR A COMMA?
0433' 20 FB     00511          JR      NZ,LTRLD
0435' 3E 1A     00512          LD      A,'*'
0437' 12        00513          LD      (DE),A
0438' 23        00514          INC     HL
0439' 11 001E'  00515          LD      DE,OPRNDS
043C' ED A0     00516 REVRS:   LDI
043E' BE        00517          CP      (HL)
043F' 20 FB     00518          JR      NZ,REVRS
0441' 3E 20     00519          LD      A,' '
0443' 77        00520          LD      (HL),A
0444' 01 0008   00521          LD      BC,8
0447' ED B0     00522          LDIR
0449' C3 0106'  00523          JP      INSRET          ;CLEAN UP & GO BACK
                00524 ;
                00525 ;                         -OO*OO-
                00526 ;                            "
044C' 21 003B'  00527 LOAD16:  LD      HL,LDM
044F' 01 0002   00528          LD      BC,2
0452' ED B0     00529          LDIR
0454' 11 001E'  00530          LD      DE,OPRNDS       ;POINT AT OPERAND FIELD:
0457' CD 0698'  00531          CALL    SS
                00532          PUT     ','
045A' 3E 20                    LD      A,
```

```
045C  12                    LD      (DE),A
045D  13                    INC     DE
                  00533     NEXPT           ;(REF COUNTER + 1) INTO HL PAIR
045E  E1                    POP     HL
045F  E5                    PUSH    HL
0460  23                    INC     HL
0461  CD 06C8   00534       CALL    NUMB16
0464  C3 01C6   00535       JP      INSRET
                00536 ;
                00537 ;                         -=(0)=-
0467  21 003C   00538 INC16:LD      HL,INCM         ;OPCODE IS "INC"
046A  18 03     00539       JR      GO
046C  21 0038   00540 DEC16:LD      HL,DECM         ;OPCODE IS "DEC"
046F  01 0004   00541 GO:   LD      BC,4
0472  ED B0     00542       LDIR                    ;WRITE OPCODE
0474  11 001E   00543       LD      DE,OPRNDS       ;POINT AT OPERAND FIELD
0477  CD 0698   00544       CALL    SS              ;WRITE OPERAND REG PAIR
047A  C3 01C6   00545       JP      INSRET
                00546 ;
                00547 ;                         -=(0)=-
                00548 ;
                00549 ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                00550 ;   LOOKUP TABLES:
                00551 ;
047D  3A 0000   00552 LOOKA:LD      A,(BYTE)        ;GET BACK INSTR BYTE
0480  CB 5F     00553       BIT     3,A
0482  28 05     00554       JR      Z,LOOKA2
0484  21 0053   00555       LD      HL,LTAB1        ;LOOK IN TABLE 1
0487  18 03     00556       JR      LOOKA3
0489  21 0063   00557 LOOKA2:LD      HL,LTAB2        ;LOOK IN TABLE 2
048C  E6 F0     00558 LOOKA3:AND    0F0     ;MOST SIG. HEX DIGIT MATTERS
048E  1F        00559       RRA
048F  1F        00560       RRA             ;DIVIDE BY 4
0490  4F        00561       LD      C,A
0491  06 00     00562       LD      B,0
0493  09        00563       ADD     HL,BC   ;INDEX INTO TABLE
0494  01 0004   00564       LD      BC,4    ;FOUR CHARACTERS PER ENTRY
0497  ED B0     00565       LDIR
0499  C3 01C6   00566       JP      INSRET  ;TIDY UP AND RETURN
                00567 ;
                00568 ;                         -=(0)=-
049C  21 00EC   00569 ADDHL.LD      HL,LOGTAB       ;OPCODE IS "ADD HL,"
049F  01 0004   00570       LD      BC,4
04A2  ED B0     00571       LDIR
04A4  11 001E   00572       LD      DE,OPRNDS
04A7  21 0035   00573       LD      HL,HLMES
04AA  01 0003   00574       LD      BC,3
04AD  ED B0     00575       LDIR
04AF  CD 0698   00576       CALL    SS      ;WRITE SOURCE-REGISTER PAIR
04B2  C3 01C6   00577       JP      INSRET
                00578 ;
                00579 ;                         -=(0)=-
04B5  21 0038   00580 DEC8: LD      HL,DECM
04B8  18 03     00581       JR      GO8
04BA  21 003C   00582 INC8: LD      HL,INCM
04BD  01 0004   00583 GO8:  LD      BC,4
```

```
04C0' ED B0      00584          LDIR            ;WRITE "DEC" OR "INC"
04C2' 11 001E'   00585          LD      DE,OPRNDS
04C5' 3A 0000'   00586          LD      A,(BYTE)
04C8' CD 06FF'   00587          CALL    REG38   ;GET REGISTER FROM BITS 3-5
04CB' C3 01C6'   00588          JP      INSRET
                 00589 ;
                 00590 ;                        -=(0)=-
04CE' 3A 0000'   00591 JRS:     LD      A,(BYTE)        ;----------------------
04D1' FE 00      00592          CP      00              ;HERE WE DEAL WITH
04D3' 20 0B      00593          JR      NZ,JRS1         ;RELATIVE JUMPS.
04D5' 21 0040'   00594          LD      HL,NOPM         ;----------------------
04D8' 01 0003    00595          LD      BC,3
04DB' ED B0      00596          LDIR
04DD' C3 01C6'   00597          JP      INSRET
                 00598 ;
04E0' FE 08      00599 JRS1:    CP      08
04E2' 20 13      00600          JR      NZ,JRS2
04E4' 21 0043'   00601          LD      HL,EXM
04E7' 01 0002    00602          LD      BC,2
04EA' ED B0      00603          LDIR            ;WRITE "EX"
04EC' 11 001E'   00604          LD      DE,OPRNDS
04EF' 01 0006    00605          LD      BC,6
04F2' ED B0      00606          LDIR            ;WRITE "AF,AF'"
04F4' C3 01C6'   00607          JP      INSRET
                 00608 ;
04F7' FE 10      00609 JRS2:    CP      10
04F9' 20 14      00610          JR      NZ,JRS3
04FB' 21 004B'   00611          LD      HL,DJNZM
04FE' 01 0005    00612          LD      BC,5
0501' ED B0      00613          LDIR            ;WRITE "DJNZ"
0503' 11 001E'   00614          LD      DE,OPRNDS
                 00615          NEXPT
0506' E1                        POP     HL
0507' E5                        PUSH    HL
0508' 23                        INC     HL
0509' CD 06B3'   00616          CALL    DISPLC  ;FIGURE & WRITE TARGET
050C' C3 01C6'   00617          JP      INSRET
                 00618 ;
050F' 21 0050'   00619 JRS3:    LD      HL,JRSM
0512' 01 0003    00620          LD      BC,3
0515' ED B0      00621          LDIR            ;WRITE "JR"
0517' 11 001E'   00622          LD      DE,OPRNDS
051A' CB 5F      00623          BIT     3,A     ;TEST SENSE OF CONDITION
051C' 20 07      00624          JR      NZ,POSC
                 00625          PUT     'N'
051E' 3E 4E                     LD      A,'N'
0520' 12                        LD      (DE),A
0521' 13                        INC     DE
0522' 3A 0000'   00626          LD      A,(BYTE)  ; RESTORE A
0525' E6 30      00627 POSC:    AND     30
0527' FE 10      00628          CP      10
0529' 28 10      00629          JR      Z,JRDIS
052B' FE 20      00630          CP      20
052D' 28 04      00631          JR      Z,JRZ
052F' 3E 43      00632          LD      A,'C'           ;IT'S JR C,...
0531' 18 02      00633          JR      LDDE
```

```
0533' 3E 5A      00634 JRZ:     LD      A,'Z'           ;IT'S JR Z,...
0535' 12         00635 LDDE:    LD      (DE),A
0536' 13         00636          INC     DE
                 00637          PUT     ','
0537' 3E 2C                     LD      A,','
0539' 12                        LD      (DE),A
053A' 13                        INC     DE
                 00638 JRDIS:   NEXPT
053B' E1                        POP     HL
053C' E5                        PUSH    HL
053D' 23                        INC     HL
053E' CD 06B3'   00639          CALL    DISPLC          ;FIGURE & WRITE TARGET
0541' C3 01C6'   00640          JP      INSRET
                 00641 ;
0544' CB 6F      00642 INDEXD:  BIT     5,A     ;HERE DO "INDEXED" INSTRUCTIONS
0546' 28 04      00643          JR      Z,XIND  ;FIND OUT IF IT'S IX OR IY
0548' 3E 59      00644          LD      A,'Y'
054A' 18 02      00645          JR      YIND
054C' 3E 58      00646 XIND:    LD      A,'X'
054E' 32 00B2'   00647 YIND:    LD      (XBUF+1),A      ;WRITE REGISTER NAME
                 00648          NEXPT
0551' E1                        POP     HL
0552' E5                        PUSH    HL
0553' 23                        INC     HL
0554' 7E         00649          LD      A,(HL)
0555' E6 F0      00650          AND     0F0
0557' FE 20      00651          CP      20      ;FIND OUT IF DISPLACEMENT BYTE
0559' 28 11      00652          JR      Z,NODISP        ;   IS USED.
055B' FE E0      00653          CP      0E0
055D' 28 0D      00654          JR      Z,NODISP
055F' 7E         00655          LD      A,(HL)
0560' E6 0F      00656          AND     0F
0562' FE 09      00657          CP      09
0564' 28 06      00658          JR      Z,NODISP
0566' 23         00659          INC     HL      ;IF IT IS, IT'S THE 3D BYTE.
0567' 7E         00660          LD      A,(HL)
0568' 32 00B0'   00661          LD      (DISPX),A       ;REMEMBER DISPLACEMENT.
056B' 2B         00662          DEC     HL              ;POINT AT 2ND BYTE.
                 00663 ;
056C' 7E         00664 NODISP:  LD      A,(HL)
056D' FE CB      00665          CP      0CB     ;SEE IF IT'S A BIT TWIDDLE.
056F' 20 09      00666          JR      NZ,NOTWID       ;NOT A TWIDDLE
0571' 23         00667          INC     HL              ;IF IT IS, LOOK
0572' 23         00668          INC     HL              ;AT 4TH BYTE
0573' 7E         00669          LD      A,(HL)  ;THAT'S THE INSTRUCTION.
0574' 32 00B7'   00670          LD      (CBBUF+1),A
0577' 21 00B6'   00671          LD      HL,CBBUF
                 00672 ;
057A' 32 0000'   00673 NOTWID:  LD      (BYTE),A
057D' CD 0187'   00674          CALL    INSTR           ;RECURSIVELY...      <-*
0580' 3E 48      00675 AGAIN:   LD      A,'H'
0582' 21 001E'   00676          LD      HL,OPRNDS
0585' 01 0017    00677          LD      BC,17
0588' ED B1      00678 SEE:     CPIR                    ;LOOK FOR "HL,"
058A' C2 01C6'   00679          JP      NZ,INSRET       ;IF NO MATCH FOUND
058D' 3E 4C      00680          LD      A,'L'
```

```
058F' BE          00681           CP      (HL)              ;MAKE SURE IT'S "HL"

0590' 28 04       00682           JR      Z,WHOLE
0592' 3E 48       00683           LD      A,'H'
0594' 18 F2       00684           JR      SEE
0596' 2B          00685 WHOLE:    DEC     HL
0597' E5          00686           PUSH    HL
0598' D1          00687           POP     DE                ;POINT TO IT IN "LINE."
0599' 21 00B1'    00688           LD      HL,XBUF
059C' 01 0002     00689           LD      BC,2
059F' ED B0       00690           LDIR            ;REPLACE IT WITH "IX" OR "IY"
05A1' 1B          00691           DEC     DE
05A2' D5          00692           PUSH    DE
05A3' 3A 00B0'    00693           LD      A,(DISPX)
05A6' FE 00       00694           CP      0         ;CHECK OUT INDEX DISPLACEMENT
05A8' C2 05AF'    00695           JP      NZ,ONDEX
05AB' E1          00696           POP     HL
05AC' C3 0580'    00697           JP      AGAIN     ;JUST IN CASE IT'S ADD IX,IX
                  00698 ;
05AF' 21 00B3'    00699 ONDEX:    LD      HL,DBUF
05B2' F2 05BB'    00700           JP      P,POS     ;IF DISPLACEMENT IS POSITIVE
05B5' ED 44       00701           NEG               ;ELSE GET ABSOLUTE VALUE
05B7' 36 2D       00702           LD      (HL),MINUS        ;AND PREFIX A "-"
05B9' 18 02       00703           JR      BUMP
05BB' 36 2B       00704 POS:      LD      (HL),PLUS         ;PREFIX A "+"
05BD' CD 072D'    00705 BUMP:     CALL    ACONVI
05C0' ED 53       00706           LD      (DBUF+1),DE
05C2' 00B4'
05C4' 11 0034'    00707           LD      DE,OPRNDS+16
05C7' 21 0031'    00708           LD      HL,OPRNDS+13
05CA' C1          00709           POP     BC
05CB' E5          00710           PUSH    HL
05CC' 97          00711           SUB     A
05CD' 32 00B0'    00712           LD      (DISPX),A         ;ZERO FOR NEXT TIME
05D0' ED 42       00713           SBC     HL,BC             ;BYTE COUNT FOR LDDR
05D2' E5          00714           PUSH    HL
05D3' C1          00715           POP     BC
05D4' E1          00716           POP     HL
05D5' ED B8       00717           LDDR                      ;THE VERY LDDR ITSELF
05D7' 21 00B5'    00718           LD      HL,DBUF+2
05DA' 01 0003     00719           LD      BC,3
05DD' ED B8       00720           LDDR
05DF' C3 01C6'    00721           JP      INSRET
                  00722 ;
002B              00723 PLUS      EQU     '+'
002D              00724 MINUS     EQU     '-'
                  00725 ;                           \
                  00726 ;                    -=<O(*)O>=-
                  00727 ;                           \
                  00728 TWIDL:    NEXPT   ;HERE DO THE BIT TWIDDLES
05E2' E1                          POP     HL
05E3' E5                          PUSH    HL
05E4' 23                          INC     HL
05E5' 7E          00729           LD      A,(HL)  ;   (A VERY REGULAR GROUP)
05E6' 32 0000'    00730           LD      (BYTE),A
05E9' FE 40       00731           CP      40
```

```
05EB' 38 32      00732          JR      C,SHROTA         ;GO DO SHIFTS & ROTATES:
05ED' E6 C0      00733          AND     0C0
05EF' 06 06      00734          LD      B,6
05F1' 1F         00735 SH6T:    RRA
05F2' 10 FD      00736          DJNZ    SH6T
05F4' 3D         00737          DEC     A
05F5' 17         00738          RLA
05F6' 17         00739          RLA
05F7' 4F         00740          LD      C,A
05F8' 21 00B8'   00741          LD      HL,TWITAB
05FB' 09         00742          ADD     HL,BC
05FC' 01 0003    00743          LD      BC,3
05FF' ED B0      00744          LDIR
                 00745 ;
0601' 3A 0000'   00746          LD      A,(BYTE)
0604' E6 38      00747          AND     38
0606' 1F         00748          RRA
0607' 1F         00749          RRA
0608' 1F         00750          RRA
0609' CD 072D'   00751          CALL    ACONVI           ;WHAT BIT? ASCII,PLEASE.
060C' 7A         00752          LD      A,D
060D' 11 001E'   00753          LD      DE,OPRNDS
0610' 12         00754          LD      (DE),A
0611' 13         00755          INC     DE
                 00756          PUT     ',',
0612' 3E 2C                     LD      A,','
0614' 12                        LD      (DE),A
0615' 13                        INC     DE
0616' 3A 0000'   00757 RSEND:   LD      A,(BYTE)
0619' CD 0707'   00758          CALL    REG8     ;WRITE REGISTER NAME
061C' C3 01C6'   00759          JP      INSRET
061F' E6 38      00760 SHROTA.  AND     38
0621' 1F         00761          RRA
0622' 06 00      00762          LD      B,0
0624' 4F         00763          LD      C,A
0625' 21 0003'   00764          LD      HL,SHTAB
0628' 09         00765          ADD     HL,BC
0629' 01 0003    00766          LD      BC,3
062C' ED B0      00767          LDIR
062E' 11 001E    00768          LD      DE,OPRNDS
0631' 18 E3      00769          JR      RSEND
                 00770 ;
0633' 3A 0000'   00771 LOADS:   LD      A,(BYTE)
0636' FE 76      00772          CP      76       ;IS IT A HALT?
0638' 20 0B      00773          JR      NZ,LDS
063A' 21 00E2'   00774          LD      HL,HLTM
063D' 01 0004    00775          LD      BC,4
0640' ED B0      00776          LDIR
0642' C3 01C6'   00777          JP      INSRET
                 00778 ;
0645' 21 008B'   00779 LDS:     LD      HL,LDM   ;LOADS ARE EASY,
0648' 01 0002    00780          LD      BC,2     ;LOOK AT THE KARNAUGH MAP.
064B' ED B0      00781          LDIR
064D' 11 001E'   00782          LD      DE,OPRNDS
0650' CD 06FF'   00783          CALL    REG38    ;DESTINATION REGISTER NAME
0653' 3E 2C      00784          LD      A,','
```

```
0655' 12          00785          LD      (DE),A
0656' 13          00786          INC     DE
0657' 3A 0000'    00787          LD      A,(BYTE)
065A' CD 0707'    00788          CALL    REG8    ;SOURCE REGISTER NAME
065D' C3 01C6'    00789          JP      INSRET
                  00790 ;
                  00791 ;                                \
                  00792 ;                          -=<O(*)O>=-
                  00793 ;                                /
0660' 3A 0000'    00794 LOGIC:   LD      A,(BYTE)
0663' E6 38       00795          AND     38              ;LOOK AT BITS 3-5
0665' 06 00       00796          LD      B,0
0667' 1F          00797          RRA
0668' 4F          00798          LD      C,A
0669' 21 00EC'    00799          LD      HL,LOGTAB
066C' 09          00800          ADD     HL,BC
066D' 01 0004     00801          LD      BC,4
0670' ED B0       00802          LDIR                    ;WRITE THE OPCODE
0672' 11 001E'    00803          LD      DE,OPRNDS
0675' 3A 0000'    00804          LD      A,(BYTE)
0678' E6 BF       00805          AND     0BF
067A' FE 90       00806          CP      90
067C' DC 068F'    00807          CALL    C,EXPLCA        ;EXPLICIT ACCUMULATOR
067F' E6 F8       00808          AND     0F8
0681' FE 98       00809          CP      98
0683' CC 068F'    00810          CALL    Z,EXPLCA
0686' 3A 0000'    00811          LD      A,(BYTE)
0689' CD 0707'    00812          CALL    REG8            ;WRITE OPERAND REG
068C' C3 01C6'    00813          JP      INSRET
                  00814 ;
068F' 21 00EA'    00815 EXPLCA:  LD      HL,AMES
0692' 01 0002     00816          LD      BC,2
0695' ED B0       00817          LDIR                    ;WRITE EXPLICIT OPERAND
0697' C9          00818          RET
                  00819 ;
                  00820 ;
                  00821 ;                                \
                  00822 ;                          -=<O(*)O>=-
                  00823 ;                                /
                  00824          PAGE
```

```
0698'
                    00825 ;#########################################################
                    00826 ;
                    00827 ;   THIS MODULE COMPRISES SUBROUTINES CALLED BY THE
                    00828 ;   DISASSEMBLER DISASS.
                    00829 ;
                    00830 ;#########################################################
                    00831 ;
0698' E5            00832 SS:       PUSH    HL          ;==============================
0699' C5            00833           PUSH    BC          ;WRITES NAME OF REGISTER PAIR
069A' F5            00834           PUSH    AF          ;TO LINE BUFFER
069B' 3A 0000'      00835           LD      A,(BYTE);==============================
069E' E6 30         00836           AND     30
06A0' 01 0302       00837           LD      BC,0302
06A3' 21 5053       00838           LD      HL,'PS'
06A6' 22 0112'      00839           LD      (RNAM),HL
06A9' 21 010C'      00840           LD      HL,RNAMES
06AC' CD 0782'      00841           CALL    LOOKUP
06AF' F1            00842           POP     AF
06B0' C1            00843           POP     BC
06B1' E1            00844           POP     HL
06B2' C9            00845           RET
                    00846           PAGE
```

```
06B3'
06B3' 7E          00847 DISPLC: LD      A,(HL)   ;==============================
06B4' 23          00848         INC     HL       ;FIGURES DISPLACEMENT FOR
06B5' 4F          00849         LD      C,A      ;RELATIVE JUMPS, WRITES TARGET
06B6' 06 08       00850         LD      B,8      ;TO (DE).   (TO "LINE")
06B8' CB 2F       00851 SHPROP: SRA     A        ;==============================
06BA' 10 FC       00852         DJNZ    SHPROP           ;PROPAGATE SIGN BIT
06BC' 47          00853         LD      B,A
06BD' 09          00854         ADD     HL,BC
06BE' 22 00E6'    00855         LD      (NBUF),HL
06C1' 21 00E6'    00856         LD      HL,NBUF
06C4' CD 06C8'    00857         CALL    NUMB16
06C7' C9          00858         RET
                  00859 ;
                  00860                          ;==============================;
06C8' E5          00861 NUMB16: PUSH    HL
06C9' C5          00862         PUSH    BC       ;WRITES NUMBER TO LINE BUFFER  :
06CA' F5          00863         PUSH    AF
06CB' D5          00864         PUSH    DE       ;==============================;
06CC' 7E          00865         LD      A,(HL)
06CD' CD 072D'    00866         CALL    ACONVI
06D0' ED 53       00867         LD      (NBUF+2),DE
06D2' 00E8'
06D4' 23          00868         INC     HL
06D5' 7E          00869         LD      A,(HL)
06D6' CD 072D'    00870         CALL    ACONVI
06D9' ED 53       00871         LD      (NBUF),DE
06DB' 00E6'
06DD' 21 00E6'    00872         LD      HL,NBUF
06E0' D1          00873         POP     DE
06E1' 01 0004     00874         LD      BC,4
06E4' ED B0       00875         LDIR
06E6' F1          00876         POP     AF
06E7' C1          00877         POP     BC
06E8' E1          00878         POP     HL
06E9' C9          00879         RET
                  00880         PAGE
```

```
06EA'
06EA' E5          00881 NUMB8:   PUSH     HL        ;==============================
06EB' 7E          00882          LD       A,(HL)    ;WRITES A SINGLE HEX BYTE
06EC' D5          00883          PUSH     DE        ;AS 2 ASCII CHARS TO (DE)
06ED' CD 072D'    00884          CALL     ACONVI    ;==============================
06F0' ED 53       00885          LD       (NBUF),DE
06F2' 00E6'
06F4' D1          00886          POP      DE
06F5' 21 00E6'    00887          LD       HL,NBUF
06F8' 01 0002     00888          LD       BC,2
06FB' ED B0       00889          LDIR
06FD' E1          00890          POP      HL
06FE' C9          00891          RET
                  00892 ;
06FF' F5          00893 REG38:   PUSH     AF        ;PICKS REGISTER CODE FROM
0700' E6 38       00894          AND      38        ;BITS 3-5 OF ACCUMULATOR.
0702' 1F          00895          RRA
0703' 1F          00896          RRA
0704' 1F          00897          RRA
0705' 18 01       00898          JR       REG81
0707' F5          00899 REG8:    PUSH     AF        ;==============================
0708' C5          00900 REG81:   PUSH     BC        ;EXPECTS 3-BIT CODE IN ACC
0709' E5          00901          PUSH     HL        ;WRITES REGISTER NAME TO (DE)
070A' 21 412A     00902          LD       HL,'A*'   ;==============================
070D' 22 0112'    00903          LD       (RNAM),HL         ;SET UP TABLE TAIL
0710' 21 010C'    00904          LD       HL,RNAMES         ;POINT TO TABLE
0713' E6 07       00905          AND      07
0715' 4F          00906          LD       C,A
0716' 06 00       00907          LD       B,0
0718' 09          00908          ADD      HL,BC     ;  INDEX INTO TABLE
0719' 01 0001     00909          LD       BC,1
071C' 3E 2A       00910          LD       A,'*'
071E' BE          00911          CP       (HL)
071F' 20 06       00912          JR       NZ,REG8LD
0721' 21 008F'    00913          LD       HL,HLIND
0724' 01 0004     00914          LD       BC,4
0727' ED B0       00915 REG8LD:  LDIR
0729' E1          00916          POP      HL
072A' C1          00917          POP      BC
072B' F1          00918          POP      AF
072C' C9          00919          RET
                  00920                    ;WHERE "*" IS REPLACED BY "(HL)"
                  00921 ;                     \
                  00922 ;                    -=(O)=-
                  00923 ;                     \
                  00924 ;ACONVI TAKES BINARY IN ACCUMULATOR, RETURNS ASCII IN
                  00925 ;THE DE PAIR IN LOW-HIGH ORDER.
072D' E5          00926 ACONVI:  PUSH     HL
072E' 21 0765'    00927          LD       HL,ALOC
0731' 77          00928          LD       (HL),A
0732' 3E 33       00929          LD       A,33H
0734' ED 6F       00930          RLD
0736' ED 6F       00931          RLD
0738' FE 3A       00932          CP       3AH
073A' 38 02       00933          JR       C,ANAJI1
```

```
073C' C6 07      00934           ADD     A,7
073E' 57         00935 ANAJI1:   LD      D,A
073F' 7E         00936           LD      A,(HL)
0740' FE 3A      00937           CP      3AH
0742' 38 02      00938           JR      C,ANAJI2
0744' C6 07      00939           ADD     A,7
0746' 5F         00940 ANAJI2:   LD      E,A
0747' E1         00941           POP     HL
0748' C9         00942           RET
0749' E5         00943 ACONV:    PUSH    HL
074A' 21 0765'   00944           LD      HL,ALOC
074D' 77         00945           LD      (HL),A
074E' 3E 33      00946           LD      A,33H
0750' ED 6F      00947           RLD
0752' ED 6F      00948           RLD
0754' FE 3A      00949           CP      3AH
0756' 38 E6      00950           JR      C,ANAJI1
0758' C6 07      00951           ADD     A,7
075A' 57         00952 ANAJ1:    LD      D,A
075B' 7E         00953           LD      A,(HL)
075C' FE 3A      00954           CP      3AH
075E' 38 E6      00955           JR      C,ANAJI2
0760' C6 07      00956           ADD     A,7
0762' 5F         00957 ANAJ2:    LD      E,A
0763' E1         00958           POP     HL
0764' C9         00959           RET
0765'            00960 ALOC:     DEFS    1
                 00961 ;
0766' 11 001E'   00962 CONDX:    LD      DE,OPRNDS
0769' E6 38      00963           AND     38
076B' 1F         00964           RRA
076C' 1F         00965           RRA
076D' 06 00      00966           LD      B,0
076F' 4F         00967           LD      C,A
0770' 3E 2E      00968           LD      A,'.'
0772' 21 0114'   00969           LD      HL,CONDXM
0775' 09         00970           ADD     HL,BC
0776' 0E 02      00971           LD      C,2
0778' ED B0      00972           LDIR
077A' 2B         00973           DEC     HL
077B' BE         00974           CP      (HL)
077C' C0         00975           RET     NZ
077D' 1B         00976           DEC     DE
077E' 3E 20      00977           LD      A,' '
0780' 12         00978           LD      (DE),A
0781' C9         00979           RET
                 00980 ;
                 00981 ;                                 -=(0)=-
0782' 1F         00982 LOOKUP:   RRA
0783' 10 FD      00983           DJNZ    LOOKUP
0785' C5         00984           PUSH    BC
0786' 4F         00985           LD      C,A
0787' 09         00986           ADD     HL,BC
0788' C1         00987           POP     BC
0789' ED B0      00988           LDIR
078B' C9         00989           RET
```

```
00990                                                   -=(O)=-
00991           PAGE
```

```
078C'
                00992 ;         ***  BIT MANIPULATIONS HANDLED HERE  ***
                00993 ;
078C' 23        00994 TWIDDL:   INC     HL          ;SCOOT PAST REDUNDANT BYTE
078D' 7E        00995           LD      A,(HL)      ;HERE'S THE INSTRUCTION INFO
078E' CD 0794'  00996           CALL    CBINF
0791' C3 01E9'  00997           JP      BACK
0794' FE C0     00998 CBINF:    CP      0C0         ;A REG HAS INSTR BYTE
0796' 38 10     00999           JR      C,BR        ;<C0 IS A TEST OR SHIFT
0798' CD 079C'  01000           CALL    MEMCK
079B' C9        01001           RET
079C' F6 F8     01002 MEMCK:    OR      0F8         ;DOES IT WORK ON (HL)?
079E' FE FE     01003           CP      0FE
07A0' 28 03     01004           JR      Z,MEM
07A2' 3E 09     01005           LD      A,09
07A4' C9        01006           RET
07A5' 3E 05     01007 MEM:      LD      A,05
07A7' C9        01008           RET
                01009 ;
07A8' FE 40     01010 BR:       CP      40
07AA' 38 03     01011           JR      C,SR        ;<040 IS A SHIFT OR ROTATE
07AC' 3E 41     01012           LD      A,41
07AE' C9        01013           RET
07AF' FE 30     01014 SR:       CP      30          ;30-38 NOT USED ("SLL")
07B1' 38 07     01015           JR      C,OK
07B3' FE 38     01016           CP      38
07B5' 30 03     01017           JR      NC,OK
07B7' 3E 00     01018           LD      A,00
07B9' C9        01019           RET
07BA' CD 079C'  01020 OK:       CALL    MEMCK
07BD' F6 40     01021           OR      40
07BF' C9        01022           RET
07C0' 23        01023 NDEXD:    INC     HL          ;GET PAST REDUNDANT BYTE
07C1' 7E        01024           LD      A,(HL)
07C2' 47        01025           LD      B,A         ;SAVE A COPY
07C3' FE CB     01026           CP      0CB         ;INDEXED BIT TWIDDLE?
07C5' 28 1C     01027           JR      Z,DDCB
07C7' CD 0821'  01028           CALL    LOOK        ;SAME AS AN (HL) INSTRUCTION
07CA' 67        01029           LD      H,A
07CB' 78        01030           LD      A,B         ;LOOK AT INSTRUCTION AGAIN
07CC' E6 0F     01031           AND     0F          ;LEAST SIGNIFICANT DIGIT COUNTS:
07CE' FE 09     01032           CP      09
07D0' 28 0C     01033           JR      Z,JUST1     ;SOME INDEXED INSTRUCTIONS DO
07D2' 78        01034           LD      A,B         ;NOT USE THE DISPLACEMENT BYTE.
07D3' E6 F0     01035           AND     0F0         ;...MOST SIGNIFICANT DIGIT
07D5' FE 20     01036           CP      20
07D7' 28 05     01037           JR      Z,JUST1
07D9' FE E0     01038           CP      0E0
07DB' 28 01     01039           JR      Z,JUST1     ;BUT ALL HAVE THE EXTRA PREFIX
07DD' 24        01040           INC     H
07DE' 24        01041 JUST1:    INC     H           ;FIX UP LENGTH FIELD
07DF' 7C        01042           LD      A,H         ;RESTORE INFO BYTE
07E0' C3 01E9'  01043           JP      BACK
                01044 ;
07E3' 23        01045 DDCB:     INC     HL
```

```
07E4' 23          01046           INC     HL        ;LOOK AT 4TH BYTE;
07E5' 7E          01047           LD      A,(HL)    ;THAT'S THE REAL INSTRUCTION
07E6' CD 0794'    01048           CALL    CBINF
07E9' F6 03       01049           OR      03                ;LENGTH=4
07EB' C3 01E9'    01050           JP      BACK
                  01051 ;
07EE' 23          01052 SPCIAL:   INC     HL        ;SECOND BYTE
07EF' 7E          01053           LD      A,(HL)    ;IS THE INSTRUCTION
07F0' FE 40       01054           CP      40
07F2' 38 16       01055           JR      C,ZZ0
07F4' FE 80       01056           CP      80
07F6' 38 1E       01057           JR      C,SOME0
07F8' FE A0       01058           CP      0A0
07FA' 38 0E       01059           JR      C,ZZ0
07FC' FE BC       01060           CP      0BC
07FE' 30 0A       01061           JR      NC,ZZ0
0800' E6 0F       01062           AND     0F                ;HERE WE DEAL WITH
0802' FE 04       01063           CP      04                ;BLOCK INSTRUCTIONS
0804' 38 09       01064           JR      C,BLK0
0806' FE 08       01065           CP      08
0808' 30 05       01066           JR      NC,BLK0
080A' 3E 01       01067 ZZ0:      LD      A,01      ;INVALID INSTRUCTION; LENGTH=2
080C' C3 01E9'    01068           JP      BACK
080F' E6 03       01069 BLK0:     AND     03
0811' 21 08CC'    01070           LD      HL,SPEC
0814' 18 05       01071           JR      CALK
0816' 21 08D0'    01072 SOME0:    LD      HL,SPEC1
0819' D6 40       01073           SUB     40
081B' CD 083A'    01074 CALK:     CALL    LK
081E' C3 01E9'    01075           JP      BACK
                  01076 ; LOOKUP ROUTINE;  EXPECTS AN INSTRUCTION BYTE IN
                  01077 ; REGISTER A;  RETURNS INFORMATION BYTE IN REGISTER A
                  01078 ;
0821' FE 40       01079 LOOK:     CP      40
0823' 38 0B       01080           JR      C,LOOK1
0825' FE C0       01081           CP      0C0
0827' 30 0C       01082           JR      NC,LOOK2
0829' FE 80       01083           CP      80
082B' 38 15       01084           JR      C,LOOK3
082D' 3E 48       01085           LD      A,48      ;8-BIT ARITHMETIC
082F' C9          01086           RET
                  01087 ;
0830' 21 084C'    01088 LOOK1:    LD      HL,TABL1          ;00-3F
0833' 18 05       01089           JR      LK
0835' 21 088C'    01090 LOOK2:    LD      HL,TABL2          ;C0-FF
0838' D6 C0       01091           SUB     0C0
083A' C5          01092 LK:       PUSH    BC
083B' 06 00       01093           LD      B,0
083D' 4F          01094           LD      C,A
083E' 09          01095           ADD     HL,BC     ;INDEX INTO TABLE
083F' C1          01096           POP     BC
0840' 7E          01097           LD      A,(HL)    ;BYTE FROM TABLE
0841' C9          01098           RET
                  01099 ;
0842' FE 76       01100 LOOK3:    CP      76        ;HALT INSTRUCTION?
0844' 20 03       01101           JR      NZ,LOOK4          ;ELSE 8-BIT LOAD
```

```
0846' 3E 00     01102           LD      A,0
0848' C9        01103           RET
                01104 ;
0849' 3E 08     01105 LOOK4:    LD      A,08    ;8-BIT LOAD
084B' C9        01106           RET
                01107 $EJECT
```

```
084C'
                    01108 ;TABLES FOR LOOKUP ROUTINES
                    01109 ;
084C' 00 0A 04      01110 TABL1:   DEFB    00,0A,04,48,48,48,09,48   ;0
084F' 48 48 48
0852' 09 48
0854' 48 48 08      01111          DEFB    48,48,08,48,48,48,09,48   ;
0857' 48 48 48
085A' 09 48
085C' 39 0A 04      01112          DEFB    39,0A,04,48,48,48,09,48   ;1
085F' 48 48 48
0862' 09 48
0864' 11 48 08      01113          DEFB    11,48,08,48,48,48,09,48   ;
0867' 48 48 48
086A' 09 48
086C' 31 0A 06      01114          DEFB    31,0A,06,48,48,48,09,48   ;2
086F' 48 48 48
0872' 09 48
0874' 31 48 0A      01115          DEFB    31,48,0A,48,48,48,09,48   ;
0877' 48 48 48
087A' 09 48
087C' 31 0A 06      01116          DEFB    31,0A,06,48,44,44,05,40   ;3
087F' 48 44 44
0882' 05 40
0884' 31 48 0A      01117          DEFB    31,48,0A,48,48,48,09,40   ;
0887' 48 48 48
088A' 09 40
088C' 38 08 32      01118 TABL2:   DEFB    38,08,32,12,3E,0C,49,1C   ;C
088F' 12 3E 0C
0892' 49 1C
0894' 38 18 32      01119          DEFB    38,18,32,00,3E,1E,49,1C   ;
0897' 00 3E 1E
089A' 49 1C
089C' 38 08 32      01120          DEFB    38,08,32,81,3E,0C,49,1C   ;D
089F' 81 3E 0C
08A2' 49 1C
08A4' 38 08 32      01121          DEFB    38,08,32,81,3E,00,49,1C   ;
08A7' 81 3E 00
08AA' 49 1C
08AC' 38 08 32      01122          DEFB    38,08,32,0C,3E,0C,49,1C   ;E
08AF' 0C 3E 0C
08B2' 49 1C
08B4' 38 10 32      01123          DEFB    38,10,32,08,3E,00,49,1C   ;
08B7' 08 3E 00
08BA' 49 1C
08BC' 38 08 32      01124          DEFB    38,08,32,00,3E,0C,49,1C   ;F
08BF' 00 3E 0C
08C2' 49 1C
08C4' 38 08 32      01125          DEFB    38,08,32,00,3E,00,49,1C   ;
08C7' 00 3E 00
08CA' 49 1C
08CC' 4D 41 CD      01126 SPEC:    DEFB    4D,41,0CD,0C9
08CF' C9
08D0' C9 81 49      01127 SPEC1:   DEFB    0C9,81,49,07,49,11,01,09  ;4
08D3' 07 49 11
```

```
08D6' 01 09
08D8' C9 81 49  01128           DEFB    0C9,81,49,0B,01,11,01,01  ;
08DB' 0B 01 11
08DE' 01 01
08E0' C9 81 49  01129           DEFB    0C9,81,49,07,01,01,01,49  ;5
08E3' 07 01 01
08E6' 01 49
08E8' C9 81 49  01130           DEFB    0C9,81,49,0B,01,01,01,01  ;
08EB' 0B 01 01
08EE' 01 01
08F0' C9 81 49  01131           DEFB    0C9,81,49,01,01,01,01,4D  ;6
08F3' 01 01 01
08F6' 01 4D
08F8' C9 81 49  01132           DEFB    0C9,81,49,01,01,01,01,4D  ;
08FB' 01 01 01
08FE' 01 4D
0900' 01 01 49  01133           DEFB    01,01,49,07,01,01,01,01   ;7
0903' 07 01 01
0906' 01 01
0908' C9 81 49  01134           DEFB    0C9,81,49,0B,01,01,01,01  ;
090B' 0B 01 01
090E' 01 01
0010            01135 .RADIX  16
                01136 ;
                01137 .COMMENT>
                01138           HEREIN ARE DECODED THE SPECIAL Z80 INSTRUCTIONS,
                01139 THOSE PREFIXED BY "ED."
                01140 .>
0910' E1        01141 SPECL:    POP     HL
0911' E5        01142           PUSH    HL
0912' 23        01143           INC     HL
0913' 7E        01144           LD      A,(HL)              ;GET INSTR BYTE
0914' 32 0000'  01145           LD      (BYTE),A            ;SAVE IT
0917' FE A0     01146           CP      0A0
0919' D2 09ED'  01147           JP      NC,BLOX
091C' FE 40     01148           CP      40
091E' DA 09A1'  01149           JP      C,STARS
0921' FE 63     01150           CP      63
0923' 28 7C     01151           JR      Z,STARS
0925' FE 6B     01152           CP      6B
0927' 28 78     01153           JR      Z,STARS
0929' FE 70     01154           CP      70
092B' 28 74     01155           JR      Z,STARS
092D' FE 71     01156           CP      71
092F' 28 70     01157           JR      Z,STARS
0931' E6 07     01158           AND     07
0933' FE 00     01159           CP      00
0935' CA 0A67'  01160           JP      Z,CINS
0938' FE 01     01161           CP      01
093A' CA 0A87'  01162           JP      Z,COUTS
093D' FE 02     01163           CP      02
093F' CA 0AA9'  01164           JP      Z,ADSBC
0942' FE 03     01165           CP      03
0944' CA 0AD9'  01166           JP      Z,LDED
                01167 ;
                01168 ; THIS SECTION DISASSEMBLES THE MISCELLANEOUS
```

```
                        01169 ; ED-TYPE INSTRUCTIONS -- G. D. BUZZARD
0947' 3A 0000'  01170           LD      A,(BYTE)
094A' 11 0016'  01171           LD      DE,OPCODE       ; DESTINATION
094D' FE 44     01172           CP      44              ; 'NEG' OPCODE
094F' 21 09B9'  01173           LD      HL,MNEM         ; POINT HL TO 'NEG'
0952' 28 50     01174           JR      Z,PUT
0954' FE 45     01175           CP      45              ; 'RETN'
0956' 21 09BD'  01176           LD      HL,MNEM+4
0959' 28 49     01177           JR      Z,PUT
095B' FE 4D     01178           CP      4D              ; 'RETI'
095D' 21 09C1'  01179           LD      HL,MNEM+8
0960' 28 42     01180           JR      Z,PUT
0962' FE 67     01181           CP      67              ; 'RRD'
0964' 21 09C5'  01182           LD      HL,MNEM+0C
0967' 28 3B     01183           JR      Z,PUT
0969' FE 6F     01184           CP      6F              ; 'RLD'
096B' 21 09C9'  01185           LD      HL,MNEM+10
096E' 28 34     01186           JR      Z,PUT
0970' FE 46     01187           CP      46              ; 'IM 0'
0972' 21 09CD'  01188           LD      HL,MNEM+14
0975' 28 2D     01189           JR      Z,PUT
0977' FE 56     01190           CP      56              ; 'IM 1'
0979' 21 09D1'  01191           LD      HL,MNEM+18
097C' 28 26     01192           JR      Z,PUT
097E' FE 5E     01193           CP      5E              ; 'IM 2'
0980' 21 09D5'  01194           LD      HL,MNEM+1C
0983' 28 1F     01195           JR      Z,PUT
0985' FE 47     01196           CP      47              ; 'LD I,A'
0987' 21 09D9'  01197           LD      HL,MNEM+20
098A' 28 20     01198           JR      Z,LDLD
098C' FE 57     01199           CP      57              ; 'LD A,I'
098E' 21 09DD'  01200           LD      HL,MNEM+24
0991' 28 19     01201           JR      Z,LDLD
0993' FE 5F     01202           CP      5F              ; 'LD A,R'
0995' 21 09E1'  01203           LD      HL,MNEM+28
0998' 28 12     01204           JR      Z,LDLD
099A' FE 4F     01205           CP      4F              ; 'LD R,A'
099C' 21 09E5'  01206           LD      HL,MNEM+2C
099F' 28 0B     01207           JR      Z,LDLD
09A1' 21 09E9'  01208 STARS:    LD      HL,MNEM+30      ; ERROR CHECK
09A4' 01 0004   01209 PUT:      LD      BC,0004
09A7' ED B0     01210           LDIR                    ; WRITE MNEMONIC
09A9' C3 0B26'  01211           JP      SPRET           ; DONE
09AC' E5        01212 LDLD:     PUSH    HL              ; WRITE 'LD' (OPCODE)
09AD' EB        01213           EX      DE,HL           ; MOVE POINTERS TO
09AE' 36 4C     01214           LD      (HL),'L'        ; OPRNDS SOURCE & DEST
09B0' 23        01215           INC     HL              ; FIELDS
09B1' 36 44     01216           LD      (HL),'D'
09B3' 11 001E'  01217           LD      DE,OPRNDS       ; NEW DESTINATION
09B6' E1        01218           POP     HL              ; RESTORE SOURCE
09B7' 18 EB     01219           JR      PUT             ; WRITE OPRNDS
09B9' 4E 45 47  01220 MNEM:     DEFM    'NEG RETNRETIRRD RLD IM 0IM 1'
09BC' 20 52 45
09BF' 54 4E 52
09C2' 45 54 49
09C5' 52 52 44
```

```
09C8' 20 52 4C
09CB' 44 20 49
09CE' 4D 20 30
09D1' 49 4D 20
09D4' 31
09D5' 49 4D 20  01221           DEFM    'IM 2I,A A,I A,R R,A ****'
09D8' 32 49 2C
09DB' 41 20 41
09DE' 2C 49 20
09E1' 41 2C 52
09E4' 20 52 2C
09E7' 41 20 2A
09EA' 2A 2A 2A
                01222 ;
                01223 ; THIS ROUTINE DISASSEMBLES THE ED-TYPE
                01224 ; BLOCK TRANSFER INSTRUCTIONS: LDI, LDIR,
                01225 ; LDD, LDDR, CPI, CPIR, CPD, CPDR.
                01226 ; OBSERVE PRECEDING JUMP STATEMENT WHEN MODIFYING
                01227 ; LINES DENOTED WITH '(***)' -- G. D. BUZZARD
09ED' 21 0A27'  01228 BLOX:     LD      HL,BLMNEM       ; BLOCK MENMONICS
09F0' 3A 0000'  01229           LD      A,(BYTE)        ; GET OPCODE
09F3' CB 57     01230           BIT     2,A
09F5' C2 09A1'  01231           JP      NZ,STARS
09F8' CB 77     01232           BIT     6,A
09FA' C2 09A1'  01233           JP      NZ,STARS
09FD' 47        01234           LD      B,A             ; USE REG B
09FE' 3E 00     01235           LD      A,0
0A00' CB 48     01236           BIT     1,B
0A02' 28 02     01237           JR      Z,$+4
0A04' C6 20     01238           ADD     A,20
0A06' CB 40     01239           BIT     0,B             ; TEST INST. TYPE
0A08' 28 02     01240           JR      Z,$+4           ; TRANSFER INSTRUCTION
                01241                                   ; (SKIP 2 BYTES)
0A0A' C6 10     01242           ADD     A,10            ; SEARCH INSTR (***)
0A0C' CB 58     01243           BIT     3,B
0A0E' 28 02     01244           JR      Z,$+4           ; INC HL INSTRUCTION
                01245                                   ; (SKIP 2 BYTES)
0A10' C6 08     01246           ADD     A,8             ; DEC HL INSTR (***)
0A12' CB 60     01247           BIT     4,B
0A14' 28 02     01248           JR      Z,$+4           ; NON-REPEAT INSTR
                01249                                   ; (SKIP 2 BYTES)
0A16' C6 04     01250           ADD     A,4             ; REPEATING INSTR(***)
0A18' 16 00     01251           LD      D,0
0A1A' 5F        01252           LD      E,A
0A1B' 19        01253           ADD     HL,DE           ; PICK RIGHT MNEMONIC
0A1C' 11 0016'  01254           LD      DE,OPCODE       ; DESTINATION
0A1F' 01 0004   01255           LD      BC,4
0A22' ED B0     01256           LDIR                    ; WRITE MNEMONIC
0A24' C3 0B26'  01257           JP      SPRET           ; DONE
0A27' 4C 44 49  01258 BLMNEM:   DEFM    'LDI LDIRLDD LDDRCPI CPIRCPD CPDR'
0A2A' 20 4C 44
0A2D' 49 52 4C
0A30' 44 44 20
0A33' 4C 44 44
0A36' 52 43 50
0A39' 49 20 43
```

```
0A3C' 50 49 52
0A3F' 43 50 44
0A42' 20 43 50
0A45' 44 52
0A47' 49 4E 49  01259           DEFM    'INI INIRIND INDROUTIOTIROUTDOTDR'
0A4A' 20 49 4E
0A4D' 49 52 49
0A50' 4E 44 20
0A53' 49 4E 44
0A56' 52 4F 55
0A59' 54 49 4F
0A5C' 54 49 52
0A5F' 4F 55 54
0A62' 44 4F 54
0A65' 44 52
                01260 ;
                01261 ; THIS ROUTINE HANDLES THE 'IN R,(C)' INSTRUCTION
0A67' 21 0016'  01262 CINS:   LD      HL,OPCODE       ; WRITE 'IN'
0A6A' 36 49     01263         LD      (HL),'I'
0A6C' 23        01264         INC     HL
0A6D' 36 4E     01265         LD      (HL),'N'        ; END WRITE 'IN'
0A6F' 11 001E'  01266         LD      DE,OPRNDS       ; DESTINATION
0A72' 3A 0000'  01267         LD      A,(BYTE)
0A75' CD 06FF'  01268         CALL    REG38           ; WRITE 'R'
0A78' 01 0004   01269         LD      BC,4
0A7B' 21 0A83'  01270         LD      HL,BRACK        ; WRITE ',(C)'
0A7E' ED B0     01271         LDIR                    ;   "
0A80' C3 0B26'  01272         JP      SPRET           ; DONE
0A83' 2C 28 43  01273 BRACK:  DEFM    ',(C)'
0A86' 29
                01274 ;
                01275 ; THIS ROUTINE HANDLES THE 'OUT (C),R' INSTRUCTION
0A87' 11 0016'  01276 COUTS:  LD      DE,OPCODE       ; DESTINATION
0A8A' 21 0AA2'  01277         LD      HL,COMNEM       ; SOURCE
0A8D' 01 0003   01278         LD      BC,3
0A90' ED B0     01279         LDIR                    ; WRITE 'OUT'
0A92' 11 001E'  01280         LD      DE,OPRNDS       ; NEW DESTINATION
0A95' 0E 04     01281         LD      C,4
0A97' ED B0     01282         LDIR                    ; WRITE '(C),'
0A99' 3A 0000'  01283         LD      A,(BYTE)        ; GET OPCODE
0A9C' CD 06FF'  01284         CALL    REG38           ; WRITE 'R'
0A9F' C3 0B26'  01285         JP      SPRET           ; DONE
0AA2' 4F 55 54  01286 COMNEM: DEFM    'OUT(C),'
0AA5' 28 43 29
0AA8' 2C
                01287 ;
                01288 ; THIS ROUTINE HANDLES THE ADC, AND SBC
                01289 ; INSTRUCTIONS.
0AA9' 11 0016'  01290 ADSBC:  LD      DE,OPCODE       ; DESTINAION
0AAC' 01 0003   01291         LD      BC,3
0AAF' 3A 0000'  01292         LD      A,(BYTE)        ; GET OPCODE
0AB2' CB 5F     01293         BIT     3,A             ; TEST FOR ADD/SUB
0AB4' 20 05     01294         JR      NZ,AD           ; ADD INSTRUCTION
0AB6' 21 0AD3'  01295         LD      HL,SBMNEM       ; SUB MNEMONIC
0AB9' 18 03     01296         JR      N1
0ABB' 21 0AD6'  01297 AD:     LD      HL,ADMNEM       ; ADD MNEMONIC
```

```
0ABE' ED B0      01298 N1:      LDIR                    ; WRITE MNEMONIC
0AC0' 21 0AD0'   01299          LD      HL,ASMNEM       ; NEW SOURCE
0AC3' 11 001E'   01300          LD      DE,OPRNDS       ; NEW DESTINATION
0AC6' 01 0003    01301          LD      BC,3
0AC9' ED B0      01302          LDIR                    ; WRITE 'HL,'
0ACB' CD 0698'   01303          CALL    SS              ; WRITE 'SS'
0ACE' 18 56      01304          JR      SPRET
0AD0' 48 4C 2C   01305 ASMNEM:  DEFM    'HL,'
0AD3' 53 42 43   01306 SBMNEM:  DEFM    'SBC'
0AD6' 41 44 43   01307 ADMNEM:  DEFM    'ADC'
                 01308 ;
                 01309 ; THIS ROUTINE HANDLES THE ED-TYPE SIXTEEN BIT
                 01310 ; LOAD INSTRUCTIONS.                G. D. BUZZARD
0AD9' 23         01311 LDED:    INC     HL              ; POINT TO NEXT BYTE
0ADA' 7E         01312          LD      A,(HL)
0ADB' 32 0B24'   01313          LD      (ADDRL),A       ; STORE IT
0ADE' 23         01314          INC     HL              ; NEXT BYTE
0ADF' 7E         01315          LD      A,(HL)
0AE0' 32 0B25'   01316          LD      (ADDRH),A       ; STORE IT
0AE3' 21 0016'   01317          LD      HL,OPCODE       ; DESTINATION
0AE6' 36 4C      01318          LD      (HL),'L'
0AE8' 23         01319          INC     HL
0AE9' 36 44      01320          LD      (HL),'D'        ; WRITE 'LD'
0AEB' 21 001E'   01321          LD      HL,OPRNDS       ; NEW DESTINATION
0AEE' 3A 0000'   01322          LD      A,(BYTE)
0AF1' CB 5F      01323          BIT     3,A
0AF3' 28 08      01324          JR      Z,N2
0AF5' EB         01325          EX      DE,HL
0AF6' CD 0698'   01326          CALL    SS
0AF9' EB         01327          EX      DE,HL
0AFA' 36 2C      01328          LD      (HL),','
0AFC' 23         01329          INC     HL
0AFD' 36 28      01330 N2:      LD      (HL),'('        ; WRITE '('
0AFF' 23         01331          INC     HL
0B00' 3A 0B25'   01332          LD      A,(ADDRH)       ; WRITE ADDRESS
0B03' 06 02      01333          LD      B,2
0B05' CD 0749'   01334 N3:      CALL    ACONV           ; CONVERT TO ASCII
0B08' 73         01335          LD      (HL),E
0B09' 23         01336          INC     HL
0B0A' 72         01337          LD      (HL),D          ; WRITE ADDRESS
0B0B' 23         01338          INC     HL
0B0C' 3A 0B24'   01339          LD      A,(ADDRL)
0B0F' 10 F4      01340          DJNZ    N3              ; FINISH WRITING ADDRESS
0B11' 36 29      01341          LD      (HL),')'
0B13' 3A 0000'   01342          LD      A,(BYTE)
0B16' CB 5F      01343          BIT     3,A
0B18' 20 0C      01344          JR      NZ,SPRET
0B1A' 23         01345          INC     HL
0B1B' 36 2C      01346          LD      (HL),','
0B1D' 23         01347          INC     HL
0B1E' EB         01348          EX      DE,HL           ; PUT DESTINATION IN DE
0B1F' CD 0698'   01349          CALL    SS              ; WRITE 'SS'
0B22' 18 02      01350          JR      SPRET           ; DONE
0B24'            01351 ADDRL:   DEFS    1
0B25'            01352 ADDRH:   DEFS    1
                 01353 ;
```

```
0B26' C3 01C6'  01354 SPRET:  JP      INSRET
                01355         END
```

```
ACROS:
EXPT    PUT

YMBOLS:
ONV     0749'   ACONVI  072D'   AD      0ABB'   ADDHL   049C'   ADDRH   0B25'
DRL     0B24'   ADMNEM  0AD6'   ADSBC   0AA9'   AGAIN   0580'   ALOC    0765'
ES      00EA'   ANAJ1    075A'   ANAJ2   0762'   ANAJI1  073E'   ANAJI2  0746'
RITH    039E'   ASMNEM  0AD0'   BACK    01E9'   BANKSW  01EE*   BLKO    080F'
MNEM    0AZ7'   BLOX    09ED'   BR      07A8'   BRACK   0A83'   BUMP    05BD'
TE      0000I'  CALK    081B'   CALLO   0274'   CALLM   0073'   CALRET  0Z4E'
BBUF    00B6'   CBINF   0794'   CINS    0A67'   CODE    000A'   COMM    0403'
MNEM    0AA2'   CONDX   0766'   CONDXM  0114'   CORE    0168'   COUTS   0A87'
BUF     00B3'   DDCB    07E3'   DEC16   046C'   DEC8    04B5'   DECM    0038'
EHLM    00AB'   DISABL  030B'   DISASS  0124I'  DISPLC  06B3'   DISPX   00B0'
JNZM    004B'   EXM     0043'   EXPLCA  068F'   EXRET   039B'   EXS     037A'
O       046F'   GO8     04BD'   HEX     0145I'  HIQTR   01F2'   HLIND   008F'
LM      03FB'   HLMES   0035'   HLTM    00E2'   INC16   0467'   INC8    04BA'
NCM     003C'   INDEXD  0544'   INDI8   03EF'   INFO    01CBI'  INM     009E'
NOUT    033D'   INSRET  01C6I'  INSTR   0187I'  INTERM  009C'   INTEWR  030D'
NK      02FC'   JPM     0087'   JPS     0236'   JRDIS   053B'   JRS     04CE'
RS1     04E0'   JRS2    04F7'   JRS3    050F'   JRSM    0050'   JRZ     0533'
UST1    07DE'   LD8     03CD'   LDDE    0535'   LDED    0AD9'   LDLD    09AC'
M       008B'   LDS     0645'   LENGTH  0001I'  LINE    0003I'  LK      083A'
AD16    044C'   LOADS   0633'   LOCN    0004'   LOGIC   0660'   LOGTAB  00EC'
OOK     0821'   LOOK1   0830'   LOOK2   0835'   LOOK3   0842'   LOOK4   0849'
OOKA    047D'   LOOKA2  0489'   LOOKA3  048C'   LOOKUP  0782'   LTAB1   0053'
TAB2    0063'   LTRLD   0430'   MEM     07A5'   MEMCK   079C'   MINUS   002D
NEM     09B9'   N1      0ABE'   N2      0AFD'   N3      0B05'   NBUF    00E6'
DEXD    07C0'   NODISP  056C'   NOPM    0040'   NOTJP   032D'   NOTWID  057A'
UMB16   06C8'   NUMB8   06EA'   OK      07BA'   ONDEX   05AF'   OPCODE  0016I'
PRNDS   001EI'  OUTM    00A1'   OUTO    0361'   OUTW    033A'   PLM     007F'
LOCK    02B2'   PLUS    002B    POPM    007B'   POPS    0292'   POPUSH  0282'
O8      05BB'   POSC    0525'   PUSHM   0077'   PUT     09A4'   PUTS    0295'
EG38    06FF'   REG8    0707'   REG81   0708'   REG8LD  0727'   RETS    0266'
EVRS    043C'   RG16    041A'   RNAM    0112'   RNAMES  010C'   RSEND   0616'
SOPND   0313'   RSTJNK  02DC'   RSTM    0099'   SBMNEM  0AD3'   SEE     0588'
H&T     05F1'   SHPROP  06B8'   SHROTA  061F'   SHTAB   00C3'   SOMEO   0816'
F1      0008'   SPCIAL  07EE'   SPEC    08CC'   SPEC1   08D0'   SPECL   0910'
PHLI    0393'   SPHLM   00A4'   SPRET   0B26'   SR      07AF'   SS      0698'
TARS    09A1'   TABL1   084C'   TABL2   088C'   TARG    0245'   TWIDDL  078C'
WIDL    05E2'   TWITAB  00B8'   VIDEO   00ED    VIDOFF  0012    WCHWAY  041D'
HOLE    05F6'   XBUF    00B1'   XIND    054C'   YIND    054E'   ZZO     080A'


O FATAL ERROR(S)
```

```
                    00001 ;         TRACK  80:VII:23
                    00002 ;                                        BY BILL MACLEOD
                    00003 ;                                        &  GREG BUZZARD
                    00004 ;                                  LAST UPDATED 07-09-81
                    00005 ;
                    00006 ;   THIS ROUTINE MAKES A TABLE OF ORIGINS AND END POINTS
                    00007 ;   OF INSTRUCTION FIELDS IN A LOADED PROGRAM.
                    00008 ;        ORG = FIRST BYTE OF INSTRUCTION FIELD
                    00009 ;        END = FIRST BYTE OF NON-INSTR FIELD
                    00010 ;   ORG-END PAIRS ARE CREATED AS 4-BYTE
                    00011 ;    FIELDS IN THE TABLE AT "ORGEND."
                    00012 ;   IT IS ASSUMED THAT CODE IS NOT SELF-MODIFYING.
                    00013 ;   A RETURN TO THE OPERATING SYSTEM IS CONSIDERED AN
                    00014 ;   ENDPOINT;  I.E., THE OPERATING SYSTEM IS NOT TRACED.
                    00015 ;
                    00016 ;   THE ROUTINE EXPECTS THE STARTING ADDRESS
                    00017 ;   TO BE PASSED IN REGISTER PAIR HL.
                    00018 ;
0010                00019 RADIX   16
                    00020         EXTRN   INFO
                    00021         EXTRN   CURSOR
                    00022         EXTRN   CURSES
                    00023         EXTRN   KEYIN
                    00024         EXTRN   HEXIT
                    00025         EXTRN   SYN
                    00026         EXTRN   CICO
                    00027         ENTRY   TRACK
                    00028         ENTRY   ORGEND
010C                00029 CO      EQU     010C
F8F0                00030 INMESS  EQU     0F8F0
0000                00031 CURSH   EQU     00
00FF                00032 CURSL   EQU     0FF
0082                00033 COML    EQU     82
0005                00034 COMH    EQU     05
F8FF                00035 SCLOC   EQU     0F8FF
                    00036 ;
0000' F5            00037 TRACK:  PUSH    AF        ;SAVE CALLER'S REGISTERS
0001' C5            00038         PUSH    BC
0002' D5            00039         PUSH    DE
0003' E5            00040         PUSH    HL
                    00041 ;
0004' 22 0326'      00042         LD      (ORGEND),HL       ;START @ IS 1ST ORG
0007' 21 0326'      00043         LD      HL,ORGEND
000A' 22 031C'      00044         LD      (CURRNT),HL
000D' 21 0000       00045         LD      HL,0000
0010' 22 0328'      00046         LD      (ORGEND+2),HL
0013' 22 032C'      00047         LD      (ORGEND+6),HL
0016' 22 0319'      00048         LD      (INDFLG),HL
0019' 2B            00049         DEC     HL
001A' 22 032A'      00050         LD      (ORGEND+4),HL
001D' 21 032A'      00051         LD      HL,ORGEND+4       ;NEXT AVAILABLE
0020' 22 031E'      00052         LD      (HEADER),HL       ;ORG-END FIELD
0023' 11 032E'      00053         LD      DE,ORGEND+8
0026' 01 03F8       00054         LD      BC,03F8
0029' ED B0         00055         LDIR
002B' 2A 0326'      00056         LD      HL,(ORGEND)       ;FIRST ORG
```

```
                        00057 ;
002E' CD 0000*          00058 FIRST:    CALL    INFO
0031' F5                00059           PUSH    AF
0032' E6 03             00060           AND     03
0034' 3C                00061           INC     A
0035' 06 00             00062           LD      B,0
0037' 4F                00063           LD      C,A
0038' ED 43             00064           LD      (LENGTH),BC
003A' 031A'
003C' F1                00065           POP     AF
003D' CB 67             00066           BIT     4,A
003F' 20 03             00067           JR      NZ,BRANCH
0041' 09                00068           ADD     HL,BC   ;REF PTR + LENGTH
0042' 18 EA             00069           JR      FIRST
                        00070 ;
0044' CB 6F             00071 BRANCH:   BIT     5,A     ;IS IT CONDITIONAL?
0046' 28 06             00072           JR      Z,NOPE
0048' CD 0144'          00073 B4:       CALL    TARGET
004B' 09                00074           ADD     HL,BC
004C' 18 E0             00075           JR      FIRST
                        00076 ;
                        00077 ; ***     UNCONDITIONAL BRANCH HANDLED HERE:
004E' 7E                00078 NOPE:     LD      A,(HL)
004F' FE CD             00079           CP      0CD     ;CALL?
0051' 28 F5             00080           JR      Z,B4    ;JUST LIKE CONDITIONAL BRANCH
0053' FE C9             00081           CP      0C9     ;RETURN?
0055' 28 03             00082           JR      Z,N?    ;IF SO, NO TARGET
0057' CD 0144'          00083           CALL    TARGET
005A' 09                00084 N?:       ADD     HL,BC
005B' DD 2A             00085           LD      IX,(CURRNT)
005D' 031C'
005F' DD 75 02          00086           LD      (IX+2),L        ;END <-- REF POINTER
0062' DD 74 03          00087           LD      (IX+3),H
0065' DD E5             00088           PUSH    IX
0067' E1                00089           POP     HL      ;POINT AT CURRENT ORG
0068' 01 0004           00090           LD      BC,0004
006B' 09                00091           ADD     HL,BC
006C' ED 43 031C'       00092           LD      (CURRNT),HL     ;CURRNT<--CURRNT+4
006F' DD 66 05          00093           LD      H,(IX+5)        ;REF PTR <-- ((CURRNT))
0072' DD 6E 04          00094           LD      L,(IX+4)        ;((CURRNT)) IS NEXT ORG
0075' 0E 01             00095           LD      C,1
0077' ED 4A             00096           ADC     HL,BC
0079' 28 03             00097           JR      Z,SORTT
                        00098 ;         IF NEXT ORG-END IS FFFF-0000, YOU'RE DONE,
                        00099 ;         GO SORT THE ORG-END LIST.
                        00100 ;         ELSE GO BACK & CHECK ANOTHER INSTRUCTION.
007B' 2B                00101           DEC     HL
007C' 18 B0             00102           JR      FIRST
                        00103 ;
                        00104 $EJECT
```

```
007E'
                        00105 ;
007E' 3A 0319'          00106 SORTT:    LD      A,(INDFLG)      ;WAS "JP (HL)" USED?
0081' FE 00             00107           CP      0
0083' CA 0136'          00108           JP      Z,TRRET         ; IF NOT USED
0086' 21 02B0'          00109           LD      HL,MESS1
0089' CD 029D'          00110           CALL    MESS
                        00111 ; GET USER DEFINED ORG-END PAIRS
008C' 11 F8F0           00112           LD      DE,INMESS       ; PRINT PROMPT
008F' 21 030C'          00113           LD      HL,MESS3
0092' 01 000D           00114           LD      BC,0D
0095' ED B0             00115           LDIR
                        00116 ;
0097' DD 21             00117           LD      IX,ORGEND       ; START OF TABLE
0099' 0326'
009B' D9                00118           EXX
009C' 01 01FF           00119           LD      BC,1FF          ; STORE OVERFLOW
009F' D9                00120           EXX                     ; COUNTER
                        00121 ;
00A0' 21 0000*          00122 T2:       LD      HL,CURSES       ; MOVE CURSOR POSITION
00A3' 23                00123           INC     HL              ; (OUR LINKER DOES
00A4' 3E 00             00124           LD      A,CURSH         ; NOT ALLOW OFFSETS ON
00A6' 77                00125           LD      (HL),A          ; EXTERNAL SYMBOLS
00A7' 23                00126           INC     HL
00A8' 23                00127           INC     HL
00A9' 3E FF             00128           LD      A,CURSL
00AB' 77                00129           LD      (HL),A
00AC' CD 0000*          00130           CALL    CURSOR
                        00131 ;
00AF' E5                00132           PUSH    HL              ; SAVE CURRENT VALUE
00B0' 06 19             00133           LD      B,19
00B2' 3E 20             00134           LD      A,20            ; ASCII BLANK
00B4' 21 F8FF           00135           LD      HL,SCLOC        ; SCREEN LOC'N
00B7' 77                00136 T5:       LD      (HL),A          ; BLANK OUT LINE
00B8' 23                00137           INC     HL
00B9' 10 FC             00138           DJNZ    T5
00BB' E1                00139           POP     HL
                        00140 ;
00BC' 21 F8FF           00141           LD      HL,SCLOC        ; SCREEN LOC'N FOR
                        00142                                   ; INPUT
00BF' CD 0000*          00143           CALL    CICO            ; GET INPUT
                        00144 ;
00C2' DD E5             00145           PUSH    IX              ; POINT TO NEXT LOC'N
00C4' E1                00146           POP     HL              ; IN TABLE
00C5' DD 23             00147           INC     IX              ; INCREMENT POINTER
00C7' DD 23             00148           INC     IX
00C9' 23                00149           INC     HL              ; POINT TO HIGH BYTE
                        00150                                   ; FIRST
                        00151 ; CHECK VALIDITY OF INPUT
00CA' 06 02             00152           LD      B,2
00CC' 11 0000*          00153           LD      DE,KEYIN        ; POINT TO INPUT
                        00154 ;
00CF' CD 0000*          00155 T1:       CALL    HEXIT           ; CONVERT TO HEX
00D2' CB 7F             00156           BIT     7,A             ; ERROR ?
00D4' 20 29             00157           JR      NZ,SYNZ         ; IF SO, JUMP
```

```
                        00158 ;
00D6' ED 6F             00159           RLD                             ; STORE VALUE
00D8' 13                00160           INC     DE                      ; NEXT ASCII
                        00161 ;
00D9' CD 0000*          00162           CALL    HEXIT                   ; CONVERT TO HEX
00DC' CB 7F             00163           BIT     7,A                     ; ERROR ?
00DE' 20 1F             00164           JR      NZ,SYN2                 ; IF SO, JUMP
                        00165 ;
00E0' ED 6F             00166           RLD                             ; STORE VALUE
00E2' 13                00167           INC     DE                      ; NEXT ASCII
00E3' 2B                00168           DEC     HL                      ; POINT TO LOW BYTE
00E4' 10 E9             00169           DJNZ    T1                      ; RETURN FOR LOW
                        00170 ;
00E6' 1A                00171           LD      A,(DE)                  ; NEXT ASCII
00E7' FE 0D             00172           CP      0D                      ; IS IT <CR> ?
00E9' 20 14             00173           JR      NZ,SYN2                 ; IF NOT, JUMP
                        00174 ;
                        00175 ; CHECK FOR OVERFLOW.
00EB' D9                00176           EXX
00EC' 0B                00177           DEC     BC                      ; DECREMENT COUNTER
00ED' 21 0000           00178           LD      HL,00
00F0' BF                00179           CP      A                       ; ZERO CARRY FLAG
00F1' ED 4A             00180           ADC     HL,BC                   ; CHECK FOR OVERFLOW
00F3' D9                00181           EXX
00F4' 20 24             00182           JR      NZ,T3                   ; JUMP IF ^OVERFLOW
00F6' 21 02F4'          00183           LD      HL,MESS2                ; PRINT MESSAGE
00F9' CD 029D'          00184           CALL    MESS
00FC' C3 0123           00185           JP      0123                    ; RETURN TO SYSTEM
                        00186 ;
                        00187 ; SYNTAX ERROR HANDLER.
00FF' DD E5             00188 SYN2:     PUSH    IX                      ; ERROR HANDLER
0101' DD 21             00189           LD      IX,SYN3                 ; RETURN ADDRESS
0103' 010D'
0105' 21 0000*          00190           LD      HL,SYN                  ; IN EXEC ROUTINE
0108' 01 0004           00191           LD      BC,04
010B' 09                00192           ADD     HL,BC                   ; JUMP TARGET
010C' E9                00193           JP      (HL)
                        00194 ;
010D' DD E1             00195 SYN3:     POP     IX                      ; RESTORE IX
010F' DD E5             00196           PUSH    IX
0111' E1                00197           POP     HL                      ; GET HL
0112' 2B                00198           DEC     HL                      ; RESTORE TO PREVIOUS
                        00199                                           ; VALUE
0113' 06 02             00200           LD      B,2
0115' 11 0000*          00201           LD      DE,KEYIN
0118' 18 B5             00202           JR      T1                      ; TRY  AGAIN
                        00203 ;
                        00204 ; CHECK FOR END OF INPUTS
011A' 23                00205 T3:       INC     HL                      ; GET MSBYTE OF THIS
011B' 7E                00206           LD      A,(HL)                  ; INPUT
011C' FE 00             00207           CP      00                      ; IS IT 00
011E' 20 80             00208           JR      NZ,T2                   ; IF NOT, LOOP
                        00209 ;
0120' 2B                00210           DEC     HL                      ; GET PREVIOUS INPUT
0121' 2B                00211           DEC     HL
0122' 7E                00212           LD      A,(HL)                  ; MSBYTE
```

```
0123' FE FF     00213          CP      OFF             ; WAS IF OFF ?
0125' C2 00A0'  00214          JP      NZ,T2           ; IF NOT, LOOP
                00215 ;
0128' 21 0000*  00216          LD      HL,CURSES       ; PUT CURSOR BACK
012B' 23        00217          INC     HL              ; TO COMMAND AREA
012C' 3E 05     00218          LD      A,COMH
012E' 77        00219          LD      (HL),A
012F' 23        00220          INC     HL
0130' 23        00221          INC     HL
0131' 3E 82     00222          LD      A,COML
0133' 77        00223          LD      (HL),A
0134' 18 09     00224          JR      T6
                00225 ;
                00226 ;
0136' CD 023A'  00227 TRRET:   CALL    SORT
0139' CD 01F8'  00228          CALL    MERGE
013C' CD 023A'  00229          CALL    SORT
                00230 ;
013F' E1        00231 T6:      POP     HL
0140' D1        00232          POP     DE
0141' C1        00233          POP     BC
0142' F1        00234          POP     AF
0143' C9        00235          RET
                00236 $EJECT
```

```
0144'
                    00237 ;TARGET FINDER EXPECTS HL TO POINT AT INSTRUCTION
                    00238 ;AND EXPECTS ACCUMULATOR TO HOLD INFO BYTE
                    00239 ;
0144' F5            00240 TARGET: PUSH    AF
0145' C5            00241         PUSH    BC
0146' E5            00242         PUSH    HL
0147' 3A 031A'      00243         LD      A,(LENGTH)      ;"LENGTH" IS 2 BYTES
014A' FE 01         00244         CP      01
014C' 28 10         00245         JR      Z,RSTS  ;IT'S A RESTART (OR JP (HL) )
014E' FE 02         00246         CP      02
0150' 28 28         00247         JR      Z,JRS   ;IT'S A RELATIVE JUMP
                    00248
                    00249                         ;TO GET THIS FAR, IT MUST BE
0152' 23            00250         INC     HL      ;AN ABSOLUTE JUMP
0153' E5            00251         PUSH    HL
0154' DD E1         00252         POP     IX
0156' DD 66 01      00253         LD      H,(IX+1)
0159' DD 6E 00      00254         LD      L,(IX+0)
015C' 18 2C         00255         JR      CHEKR
                    00256 ;
015E' 7E            00257 RSTS:   LD      A,(HL)
015F' FE E9         00258         CP      0E9     ;IS IT "JP  (HL)"?
0161' 28 0C         00259         JR      Z,HLIND
0163' FE C9         00260         CP      0C9     ;RET?
0165' CA 01EF'      00261         JP      Z,TARET
0168' E6 38         00262         AND     38      ;STRIP OUT ADDRESS BITS
016A' 26 00         00263         LD      H,0
016C' 6F            00264         LD      L,A
016D' 18 1B         00265         JR      CHEKR
                    00266 ;
016F' 3E FF         00267 HLIND:  LD      A,0FF
0171' 32 0319'      00268         LD      (INDFLG),A
0174' E1            00269         POP     HL
0175' C1            00270         POP     BC
0176' D1            00271         POP     DE
0177' C3 007E'      00272         JP      SORTT
                    00273 ;
017A' 23            00274 JRS:    INC     HL      ;POINT AT DISPLACEMENT
017B' 7E            00275         LD      A,(HL)
017C' FE E9         00276         CP      0E9
017E' 28 EF         00277         JR      Z,HLIND ;"JP (IX)"?
0180' 23            00278         INC     HL      ;NOW POINT TO JUMP'S BASE
0181' 4F            00279         LD      C,A
0182' 06 08         00280         LD      B,8
0184' CB 2F         00281 SIGN:   SRA     A
0186' 10 FC         00282         DJNZ    SIGN    ;COPY SIGN BIT THRU
0188' 47            00283         LD      B,A
0189' 09            00284         ADD     HL,BC   ;NOW SEND HL TO CHEKR
                    00285 $EJECT
```

```
018A'                   00286 ;
018A' 22 0320'          00287 CHEKR:    LD      (TEMP1),HL
018D' 01 0123           00288           LD      BC,0123
0190' 97                00289           SUB     A
0191' ED 42             00290           SBC     HL,BC
0193' 28 5A             00291           JR      Z,TARET ;TARGET = SPDOS ?
0195' DD 21             00292           LD      IX,ORGEND
0197' 0326'
0199' DD 46 01          00293 NEXT1:    LD      B,(IX+1)
019C' DD 4E 00          00294           LD      C,(IX+0)
019F' 21 0000           00295           LD      HL,0
01A2' 97                00296           SUB     A
01A3' ED 42             00297           SBC     HL,BC
01A5' 28 20             00298           JR      Z,NEW1  ;IS IT A NEW ONE?
01A7' 2A 0320'          00299           LD      HL,(TEMP1)
01AA' 97                00300           SUB     A
01AB' ED 42             00301           SBC     HL,BC       ;ORG>TARGET ==>CARRY
01AD' 28 40             00302           JR      Z,TARET     ;IF ORG=TARGET DO NOTHING
01AF' 30 07             00303           JR      NC,CKEND    ;GO CHECK END
01B1' 01 0004           00304 NEXT2:    LD      BC,0004
01B4' DD 09             00305           ADD     IX,BC
01B6' 18 E1             00306           JR      NEXT1
01B8' 2A 0320'          00307 CKEND:    LD      HL,(TEMP1)
01BB' DD 46 03          00308           LD      B,(IX+3)
01BE' DD 4E 02          00309           LD      C,(IX+2)
01C1' ED 42             00310           SBC     HL,BC
01C3' 38 2A             00311           JR      C,TARET
01C5' 18 EA             00312 .         JR      NEXT2
01C7' DD 2A             00313 NEW1:     LD      IX,(HEADER)
01C9' 031E'
01CB' 2A 0320'          00314           LD      HL,(TEMP1)
01CE' DD 74 01          00315           LD      (IX+1),H
01D1' DD 75 00          00316           LD      (IX+0),L
01D4' 2A 031E'          00317           LD      HL,(HEADER)
01D7' 01 0004           00318           LD      BC,0004
01DA' 09                00319           ADD     HL,BC
01DB' 22 031E'          00320           LD      (HEADER),HL
01DE' 01 0722'          00321           LD      BC,ORGEND+3FC
01E1' 97                00322           SUB     A
01E2' ED 42             00323           SBC     HL,BC
01E4' 38 09             00324           JR      C,TARET
01E6' 21 02F4'          00325           LD      HL,MESS2
01E9' CD 029D'          00326           CALL    MESS
01EC' C3 0123           00327           JP      0123
01EF' E1                00328 TARET:    POP     HL
01F0' C1                00329           POP     BC
01F1' F1                00330           POP     AF
01F2' C9                00331           RET
                        00332 $EJECT
```

```
01F3'
01F3' DD 21      00333 MERGE:  LD      IX,ORGEND
01F5' 0326'
01F7' DD 66 03   00334 MERG1:  LD      H,(IX+3)
01FA' DD 6E 02   00335         LD      L,(IX+2)        ;THIS END
01FD' DD 46 05   00336         LD      B,(IX+5)
0200' DD 4E 04   00337         LD      C,(IX+4)        ;NEXT ORG
0203' 97         00338         SUB     A
0204' ED 42      00339         SBC     HL,BC
0206' 38 21      00340         JR      C,NEXPR
0208' DD 7E 06   00341         LD      A,(IX+6)        ;MERGE THIS PAIR
                 00342 ;                                 WITH THE NEXT
020B' DD 77 02   00343         LD      (IX+2),A
020E' DD 7E 07   00344         LD      A,(IX+7)
0211' DD 77 03   00345         LD      (IX+3),A
0214' DD 36 04   00346         LD      (IX+4),0FF      ;STUFF NEXT PAIR
0217' FF
0218' DD 36 05   00347         LD      (IX+5),0FF
021B' FF
021C' DD 36 06   00348         LD      (IX+6),0        ;WITH NULL VALUES
021F' 00
0220' DD 36 07   00349         LD      (IX+7),0
0223' 00
0224' CD 023A'   00350         CALL    SORT
0227' 18 CE      00351         JR      MERG1
0229' 01 0004    00352 NEXPR:  LD      BC,0004         ;NEXT PAIR
022C' DD 09      00353         ADD     IX,BC
022E' DD E5      00354         PUSH    IX
0230' E1         00355         POP     HL
0231' 01 0722'   00356         LD      BC,ORGEND+3FC
0234' 97         00357         SUB     A
0235' ED 42      00358         SBC     HL,BC
0237' 38 BE      00359         JR      C,MERG1         ;IF NOT TO END YET
0239' C9         00360         RET
                 00361 $EJECT
```

```
023A'
023A' DD E5      00362 SORT:   PUSH    IX        ;BUBBLE SORT; MAX N=3FF  *******
023C' 2A 031E'   00363         LD      HL,(HEADER)      ;SORTS 4-BYTE FIELDS   *
023F' 01 0326'   00364         LD      BC,ORGEND        ;ON 1ST 2 BYTES      ****
0242' 97         00365         SUB     A         ;ZERO IT OUT & CLEAR CARRY FLAG
0243' ED 42      00366         SBC     HL,BC
0245' 3E 03      00367         LD      A,03      ;NOW DIVIDE BY 4
0247' A4         00368         AND     H
0248' 06 06      00369         LD      B,6
024A' CB 27      00370 SH6:    SLA     A
024C' 10 FC      00371         DJNZ    SH6
024E' CB 3D      00372         SRL     L
0250' CB 3D      00373         SRL     L
0252' B5         00374         OR      L
0253' 4F         00375         LD      C,A       ;#OF FIELDS (<100H)
0254' 21 0326'   00376 AGAIN:  LD      HL,ORGEND
0257' 97         00377         SUB     A         ;CLEAR CARRY FLAG
0258' 08         00378         EX      AF,AF'    ;SAVE IT OUT OF SIGHT
0259' 41         00379         LD      B,C
025A' 05         00380         DEC     B
025B' C5         00381 LOOP:   PUSH    BC
025C' E5         00382         PUSH    HL
025D' DD E1      00383         POP     IX
025F' DD 56 01   00384         LD      D,(IX+1)
0262' DD 5E 00   00385         LD      E,(IX+0)
0265' DD 66 05   00386         LD      H,(IX+5)
0268' DD 6E 04   00387         LD      L,(IX+4)
026B' 97         00388         SUB     A
026C' ED 52      00389         SBC     HL,DE
026E' 01 0004    00390         LD      BC,0004
0271' 30 1D      00391         JR      NC,NOSWAP
0273' 11 0320'   00392         LD      DE,TEMP1
0276' DD E5      00393         PUSH    IX
0278' E1         00394         POP     HL
0279' ED B0      00395         LDIR
027B' 01 0004    00396         LD      BC,0004
027E' DD E5      00397         PUSH    IX
0280' D1         00398         POP     DE
0281' ED B0      00399         LDIR
0283' 01 0004    00400         LD      BC,0004
0286' 21 0320'   00401         LD      HL,TEMP1
0289' ED B0      00402         LDIR
028B' 37         00403         SCF               ;SWAP FLAG, WE'LL CALL IT
028C' 08         00404         EX      AF,AF'    ;PUT IT AWAY FOR A WHILE
028D' 01 0004    00405         LD      BC,0004
0290' DD E5      00406 NOSWAP: PUSH    IX
0292' E1         00407         POP     HL
0293' 09         00408         ADD     HL,BC
0294' C1         00409         POP     BC
0295' 10 C4      00410         DJNZ    LOOP
0297' 08         00411         EX      AF,AF'    ;BRING BACK SWAP FLAG
0298' 38 BA      00412         JR      C,AGAIN
029A' DD E1      00413         POP     IX
029C' C9         00414         RET
                 00415 $EJECT
```

```
029D'
029D' 11 F990    00416 MESS:   LD      DE,0F990        ; SCREEN LOC'N
02A0' 01 02B0'   00417         LD      BC,MESS1        ; CHECK WHICH MESS
02A3' 7D         00418         LD      A,L
02A4' B9         00419         CP      C
02A5' 01 0018    00420         LD      BC,18           ; ASSUME SHORT ONE
02A8' 20 03      00421         JR      NZ,N1           ; JUMP IF SO
02AA' 01 0044    00422         LD      BC,44
                 00423 ;
02AD' ED B0      00424 N1:     LDIR
02AF' C9         00425         RET
                 00426 ;
02B0' 20 59 4F   00427 MESS1:  DEFM    ' YOU HAVE USED COMPUTED JUMPS -- '
02B3' 55 20 48
02B6' 41 56 45
02B9' 20 55 53
02BC' 45 44 20
02BF' 43 4F 4D
02C2' 50 55 54
02C5' 45 44 20
02C8' 4A 55 4D
02CB' 50 53 20
02CE' 2D 2D 20
02D1' 50 4C 45   00428         DEFM    'PLEASE ENTER ORG-END TABLE VALUES.
02D4' 41 53 45
02D7' 20 45 4E
02DA' 54 45 52
02DD' 20 4F 52
02E0' 47 2D 45
02E3' 4E 44 20
02E6' 54 41 42
02E9' 4C 45 20
02EC' 56 41 4C
02EF' 55 45 53
02F2' 2E 20
02F4' 20 4F 52    00429 MESS2:  DEFM    ' ORG-END TABLE OVERFLOW '
02F7' 47 2D 45
02FA' 4E 44 20
02FD' 54 41 42
0300' 4C 45 20
0303' 4F 56 45
0306' 52 46 4C
0309' 4F 57 20
030C' 54 41 42    00430 MESS3:  DEFM    'TABLE ENTRIES'
030F' 4C 45 20
0312' 45 4E 54
0315' 52 49 45
0318' 53
0319'             00431 INDFLG: DEFS    1
031A'             00432 LENGTH: DEFS    2
031C'             00433 CURRNT: DEFS    2
031E'             00434 HEADER: DEFS    2
0320'             00435 TEMP1:  DEFS    4
0324'             00436 BUFF:   DEFS    2
0326'             00437 ORGEND: DEFS    400
```

```
00438          END
```

```
ICROS:

'MBOLS:
AIN    0254'   B4      0048'   BRANCH  0044'   BUFF    0324'   CHEKR   018A'
 CO    00C0*   CKEND   01B8'   CO      010C    COMH    0005    COML    0082
RRNT   031C'   CURSES  0129*   CURSH   0000    CURSL   00FF    CURSOR  00AD*
.RST   002E'   HEADER  031E'   HEXIT   00DA*   HLIND   016F'   INDFLG  0319'
IFO    002F*   INMESS  F8F0    JRS     017A'   KEYIN   0116*   LENGTH  031A'
OP     025B'   MERG1   01F7'   MERGE   01F3'   MESS    029D'   MESS1   02B0'
SS2    02F4'   MESS3   030C'   N1      02AD'   N9      005A'   NEW1    01C7'
XPR    0229'   NEXT1   0199'   NEXT2   01B1'   NOPE    004E'   NOSWAP  0290'
GEND   0326I'  RST8    013E'   SCLOC   F8FF    SH6     024A'   SIGN    0184'
RT     023A'   SORTT   007E'   SYN     0106*   SYN2    00FF'   SYN3    010D'
.      00CF'   T2      00A0'   T3      011A'   T5      00B7'   T6      013F'
RET    01EF'   TARGET  0144'   TEMP1   0320'   TRACK   0000I'  TRRET   0136'


) FATAL ERROR(S)
```

```
0010                00001 .RADIX 16
                    00002 ;
                    00003 ; THIS ROUTINE SIMULATES THE EXECUTION OF A Z-80
                    00004 ; INSTRUCTION.  THE STATE OF THE SIMULATEE'S REGISTERS
                    00005 ; (WITH THE EXCEPTION OF THE REFRESH REGISTER, AND THE
                    00006 ; PROGRAM COUNTER) IS RESTORED BEFORE THE EXECUTION OF
                    00007 ; EACH (SIMULATED) INSTRUCTION, AND SAVED IMMEDIATELY
                    00008 ; AFTER.  THE STATE OF IFF-1 IS TREATED SIMILARLY.
                    00009 ; CALLS TO THE OPERATING SYSTEM ROUTINES, AS WELL AS TO
                    00010 ; INTERRUPT SERVICING ROUTINES (VIA IM-2) ARE  EXECUTED
                    00011 ; COMPLETELY BEFORE RETURNING CONTROL TO THE HOST
                    00012 ; PROGRAM.
                    00013 ;
                    00014 ; THE ABILITY TO TRACE INTERRUPT SERVICE ROUTINES
                    00015 ; DYNAMICALLY WITHIN A PROGRAM WILL BE ADDED AT A LATER
                    00016 ; DATE.  PRESENTLY INTERRUPT SERVICE ROUTINES CAN BE
                    00017 ; TRACED INDEPENDENTLY BY FORCING ZIP TO REQUEST ORG-
                    00018 ; END TABLE VALUES AND THEN GIVING IT THE ORG-END FOR
                    00019 ; THE SERVICE ROUTINE.  AT THAT POINT YOU A FREE TO USE
                    00020 ; THE SET INSTRUCTION TO CHANGE THE REGISTER CONTENTS TO
                    00021 ; WHATEVER VALUES THEY WOULD NORMALLY HOLD.
                    00022 ;
                    00023 ; THE SIMUL ROUTINE EXPECTS THE INSTRUCTION TO BE
                    00024 ; SIMULATED TO BE LOCATED IN  THE LOCATION
                    00025 ; POINTED BY THE CONTENTS OF REGPC.
                    00026 ;
                    00027 ; BY GREG BUZZARD 3-81
                    00028 ; LAST UPDATED 08-08-81
                    00029 ;
                    00030           EXTRN   INFO    ; INSTRUCTION INFO ROUTINE
                    00031 ;
                    00032           ENTRY   PRESAV
                    00033           ENTRY   SIMUL
                    00034           ENTRY   TARGT
                    00035           ENTRY   REGSAV  ;USER REGISTER SAVE AREA
                    00036           ENTRY   REGAF
                    00037           ENTRY   REGF
                    00038           ENTRY   REGA
                    00039           ENTRY   REGBC
                    00040           ENTRY   REGC
                    00041           ENTRY   REGB
                    00042           ENTRY   REGDE
                    00043           ENTRY   REGE
                    00044           ENTRY   REGD
                    00045           ENTRY   REGHL
                    00046           ENTRY   REGL
                    00047           ENTRY   REGH
                    00048           ENTRY   REGIX
                    00049           ENTRY   REGSP
                    00050           ENTRY   REGI
                    00051           ENTRY   REGIY
                    00052           ENTRY   REGPC
                    00053           ENTRY   REGR
                    00054           ENTRY   TEMP
                    00055           ENTRY   XRAF
                    00056           ENTRY   XRBC
```

```
                        00057 ;
0047                    00058 OPSYS   EQU     47          ENDING HIGH ADDRESS OF OPSYS
                        00059 ;
0000' F5                00060 SIMUL:  PUSH    AF
0001' C5                00061         PUSH    BC
0002' D5                00062         PUSH    DE
0003' E5                00063         PUSH    HL
                        00064 ;
0004' 01 0007           00065         LD      BC,7
0007' 11 0174'          00066         LD      DE,WORK
000A' 21 0233'          00067         LD      HL,ZERO
000D' ED B0             00068         LDIR                        ; ZERO THE WORK AREA
                        00069 ;
000F' ED 43             00070         LD      (REPT),BC           ; ZERO REPT
0011' 0272'
0013' ED 43             00071         LD      (REG),BC            ; ZERO REG
0015' 0270'
                        00072 ;
                        00073 ; GET INSTRUCTION, INFO, AND LOAD INSTRUCTION INTO
                        00074 ; THE WORK AREA
0017' 2A 026D'          00075         LD      HL,(REGPC)          ; LOAD USER'S REGPC
001A' CD 0000*          00076         CALL    INFO                ; GET INSTR INFO
001D' 32 023B'          00077         LD      (SAVE),A            ; SAVE INFO BYTE
0020' E6 03             00078         AND     03
0022' 3C                00079         INC     A                   ; # OF BYTES IN INSTR
0023' 32 023A'          00080         LD      (LEN),A             ; SAVE IT
                        00081 ;
0026' 4F                00082         LD      C,A
0027' 06 00             00083         LD      B,0
0029' E5                00084         PUSH    HL                  ; SAVE HL
                        00085 ;
002A' 09                00086         ADD     HL,BC               ; INCREMENT REGPC VALUE
002B' 22 026D'          00087         LD      (REGPC),HL          ; STORE IT
                        00088 ;
002E' E1                00089         POP     HL                  ; GET HL BACK
002F' 11 0174'          00090         LD      DE,WORK
0032' ED B0             00091         LDIR                        ; MOVE INSTR TO WORK
                        00092 ;
                        00093 ; CHECK FOR SOME OF THE SPECIAL CASE INSTRUCTIONS
0034' 3A 0174'          00094         LD      A,(WORK)            ; CHECK FOR DI OR EI
0037' FE FB             00095         CP      0FB
0039' 20 08             00096         JR      NZ,S1               ; JUMP IF NOT EI
003B' 3E FB             00097         LD      A,0FB
003D' 32 0173'          00098         LD      (EINT),A            ; ENABLE INTERRUPT
0040' C3 01EC'          00099         JP      D3
                        00100 ;
0043' FE F3             00101 S1:     CP      0F3
0045' 20 08             00102         JR      NZ,S2               ; JUMP IF NOT DI
0047' 3E 00             00103         LD      A,0
0049' 32 0173'          00104         LD      (EINT),A            ; DISABLE INTERRUPT
004C' C3 01EC'          00105         JP      D3
                        00106 ;
004F' FE 08             00107 S2:     CP      08                  ; WAS IT EX AF ?
0051' CA 01FA'          00108         JP      Z,EXAF              ; IF SO JUMP
0054' FE D9             00109         CP      0D9                 ; WAS IT EXX ?
0056' CA 020A'          00110         JP      Z,EXX               ; IF SO JUMP
```

```
                    00111 ;
0059' FE ED         00112           CP      0ED             ; FIRST BYTE ED ?
005B' 20 0E         00113           JR      NZ,S3           ; JUMP IF NOT
005D' 3A 0175'      00114           LD      A,(WORK+1)      ; LOOK AT NEXT BYTE
0060' FE 4F         00115           CP      4F              ; IS IT A LD R,A ?
0062' 20 07         00116           JR      NZ,S3           ; JUMP IF NOT
                    00117 ;
0064' 3A 025F'      00118           LD      A,(REGA)        ; GET A, PUT A+1
0067' 3C            00119           INC     A               ; INTO REGR, IT IS
0068' 32 026F'      00120           LD      (REGR),A        ; DECREMENTED LATER
                    00121 ;
                    00122 ; CHECK FOR REPEATING INSTRUCTIONS (I.E. LDDR, OTIR, ETC
                                                            )
006B' 3A 0174'      00123 S3:       LD      A,(WORK)        ; GET FIRST BYTE
006E' FE ED         00124           CP      0ED
0070' 20 15         00125           JR      NZ,CONT         ; IF NOT OED, CONT
                    00126 ;
0072' 3A 0175'      00127           LD      A,(WORK+1)      ; GET 2ND BYTE
0075' E6 F4         00128           AND     0F4
0077' FE B0         00129           CP      0B0             ; IS IT GONE?
0079' 20 0C         00130           JR      NZ,CONT         ; IF NOT, CONT
                    00131 ;
007B' CB 57         00132           BIT     2,A             ; CHECK BIT 2
007D' 20 08         00133           JR      NZ,CONT         ; IF 1, CONT
                    00134 ;
007F' ED 4B         00135           LD      BC,(REGBC)      ; SAVE REGBC
0081' 0260'
0083' ED 43         00136           LD      (REPT),BC
0085' 0272'
                    00137 ;
                    00138 ; CHECK IF THE INSTRUCTION AFFECTS THE PC
                    00139 ; IF IT DOESN'T, JUMP TO N1
0087' 3A 023B'      00140 CONT:     LD      A,(SAVE)        ; GET INFO BYTE
008A' CB 67         00141           BIT     4,A             ; DOES IT CHANGE PC?
008C' CA 0151'      00142           JP      Z,N1            ; IF NOT JUMP
                    00143 ;
008F' 3E CD         00144           LD      A,0CD           ; 1ST BYTE OF CALL
0091' 32 0178'      00145           LD      (CATCH),A
0094' 21 0228'      00146           LD      HL,CAUGHT       ; ADDRESS OF CALL
0097' 22 0179'      00147           LD      (CATCH+1),HL
                    00148 ;
009A' 2A 026D'      00149           LD      HL,(REGPC)      ; SAVE IT
009D' 22 0231'      00150           LD      (OLDPC),HL      ; FOR NON-JUMP CC'S
                    00151 ;
00A0' 3A 0174'      00152           LD      A,(WORK)        ; FIRST INSTR BYTE
00A3' CB 7F         00153           BIT     7,A             ; IS IT A RELATIVE JUMP
00A5' 20 1C         00154           JR      NZ,NZ           ; IF NOT JUMP
                    00155 ;
                    00156 ; HERE WE DEAL WITH RELATIVE JUMPS
00A7' 3A 0175'      00157           LD      A,(WORK+1)      ; GET DISPLACEMENT
00AA' 6F            00158           LD      L,A
00AB' 26 00         00159           LD      H,0
00AD' CB 7F         00160           BIT     7,A             ; CHECK FOR NEG.
00AF' 28 02         00161           JR      Z,CNT1          ; JUMP IF POS.
                    00162 ;
00B1' 26 FF         00163           LD      H,0FF           ; PROPIGATE THE 1
```

```
00B3' ED 5B      00164 CNT1:   LD      DE,(REGPC)        ; GET USER REGPC
00B5' 026D'
00B7' 19         00165         ADD     HL,DE             ; ADD IN DISPLACEMENT
00B8' 22 026D'   00166         LD      (REGPC),HL        ; PUT IT BACK
00BB' 3E 05      00167         LD      A,5
00BD' 32 0175'   00168         LD      (WORK+1),A        ; INSERT PSEUDO-DISPL.
00C0' C3 0151'   00169         JP      N1
                 00170 ;
                 00171 ; HERE WE DEAL WITH ABSOLUTE JUMPS
00C3' FE C3      00172 N2:     CP      0C3               ; IS IT ABS. JUMP?
00C5' 20 14      00173         JR      NZ,N2.1           ; IF NOT, JUMP
00C7' 2A 0175'   00174         LD      HL,(WORK+1)       ; GET TARGET
00CA' 3E 47      00175         LD      A,OPSYS           ; IS IT IN OPSYS?
00CC' BC         00176         CP      H
00CD' F2 0151'   00177         JP      P,N1              ; IF SO, JUMP
00D0' 22 026D'   00178         LD      (REGPC),HL        ; ELSE, UPDATE REGPC
00D3' 21 017B'   00179         LD      HL,TARGT          ; INSTALL PSEUDO TARGET
00D6' 22 0175'   00180         LD      (WORK+1),HL
00D9' 18 76      00181         JR      N1
                 00182 ;
                 00183 ; HERE WE DEAL WITH COMPUTED JUMPS
00DB' 11 017B'   00184 N2.1:   LD      DE,TARGT          ; GET PSEUDO-TARGET
00DE' FE E9      00185         CP      0E9               ; IS IT JP(HL)?
00E0' 20 0C      00186         JR      NZ,C1             ; IF NOT JUMP
00E2' 2A 0264'   00187         LD      HL,(REGHL)        ; SAVE HL
00E5' 22 0270'   00188         LD      (REG),HL
00E8' ED 53      00189         LD      (REGHL),DE        ; INSERT PSEUDO-TARGET
00EA' 0264'
00EC' 18 63      00190         JR      N1
                 00191 ;
00EE' FE DD      00192 C1:     CP      0DD               ; IS IT JP(IX)?
00F0' 20 0C      00193         JR      NZ,C2             ; IF NOT JUMP
00F2' 2A 0266'   00194         LD      HL,(REGIX)        ; SAVE IX
00F5' 22 0270'   00195         LD      (REG),HL
00F8' ED 53      00196         LD      (REGIX),DE        ; INSERT PSEUDO-TARGET
00FA' 0266'
00FC' 18 53      00197         JR      N1
                 00198 ;
00FE' FE FD      00199 C2:     CP      0FD               ; IS IT JP(IY)?
0100' 20 0C      00200         JR      NZ,N3             ; IF NOT JUMP
0102' 2A 026B'   00201         LD      HL,(REGIY)        ; SAVE IY
0105' 22 0270'   00202         LD      (REG),HL
0108' ED 53      00203         LD      (REGIY),DE        ; INSERT PSEUDO-TARGET
010A' 026B'
010C' 18 43      00204         JR      N1
                 00205 ;
                 00206 ; HERE WE DEAL WITH RESETS, RETURNS, AND INSTRUCTIONS
                 00207 ; WHICH AFFECT THE STACK
010E' FE ED      00208 N3:     CP      0ED               ; RETURN INSTR?
0110' 28 2D      00209         JR      Z,CSTK            ; IF SO JUMP
                 00210 ;
0112' E6 C7      00211         AND     0C7               ; CHECK FOR RST
0114' FE C7      00212         CP      0C7               ; IS IT RST?
0116' CA 0173'   00213         JP      Z,EINT            ; IF SO JUMP
                 00214 ;
0119' E6 07      00215         AND     07
```

```
011B' FE 02      00216          CP      2                ; TARGET IN STACK?
011D' FA 013F'   00217          JP      M,CSTK           ; IF SO JUMP
                 00218 ;
0120' FE 05      00219          CP      5                ; IS IT CALL?
0122' FA 012B'   00220          JP      M,N4             ; IF NOT JUMP
                 00221 ;
0125' 2A 026D'   00222          LD      HL,(REGPC)       ; CALL INSTR
0128' 22 0270'   00223          LD      (REG),HL         ; SAVE RETURN ADDRESS
                 00224 ,
012B' 2A 0175'   00225 N4:      LD      HL,(WORK+1)      ; GET JUMP TARGET
012E' 3E 47      00226          LD      A,OPSYS
0130' BC         00227          CP      H                ; IS IT CALL TO SPDS?
0131' F2 0151'   00228          JP      P,N1             ; IF SO LET IT THRU
                 00229 ;
0134' 22 026D'   00230          LD      (REGPC),HL       ; ELSE STORE IT IN REGP
0137' 21 017B'   00231          LD      HL,TARGT
013A' 22 0175'   00232          LD      (WORK+1),HL      ; INSERT PSEUDO-TARGET
013D' 18 12      00233          JR      N1
                 00234 ;
013F' ED 73      00235 CSTK:    LD      (STKPT),SP       ; SAVE HOST'S SP
0141' 023C'
0143' ED 7B      00236          LD      SP,(REGSP)       ; GET USER'S SP
0145' 0268'
0147' E1         00237          POP     HL               ; GET RETURN ADDRESS
0148' 22 026D'   00238          LD      (REGPC),HL       ; STORE IT
014B' 21 017B'   00239          LD      HL,TARGT         ; GET PSEUDO-RET ADDR
014E' E5         00240          PUSH    HL               ; STACK IT
014F' 18 03      00241          JR      $+5              ; !!!!RELATIVE JUMP!!!!
                 00242 ;
                 00243 ; LOAD THE USER'S REGISTERS AND SET UP THE SIMULATION
0151' ED 73      00244 N1:      LD      (STKPT),SP       ; SAVE HOST'S SP
0153' 023C'
0155' 3A 026A'   00245          LD      A,(REGI)
0158' ED 47      00246          LD      I,A              ; RESTORE USER I REG
015A' 3A 026F'   00247          LD      A,(REGR)
015D' D6 0F      00248          SUB     0F               ; ADJUST FOR CORRECT RE
                                                              R
015F' 00         00249          NOP                      ; IN WORK AREA
0160' ED 4F      00250          LD      R,A              ; RESTORE USER R REG
0162' 31 025E'   00251          LD      SP,REGSAV        ; RETRIEVE USER REG'S
0165' F1         00252          POP     AF
0166' C1         00253          POP     BC
0167' D1         00254          POP     DE
0168' E1         00255          POP     HL
0169' DD E1      00256          POP     IX
016B' FD 2A      00257          LD      IY,(REGIY)
016D' 026B'
016F' ED 7B      00258          LD      SP,(REGSP)       ; LOAD USER'S SP
0171' 0268'
                 00259 ;
                 00260 ; SIMULATION AREA
0173' 00         00261 EINT:    DEFB    00               ; EI OR DI
0174'            00262 WORK:    DEFS    4
0178'            00263 CATCH:   DEFS    3
017B' F3         00264 TARGT:   DI
                 00265 ;
```

```
                        00266 ; RESAVE USER'S REGISTERS
017C' ED 73             00267           LD      (REGSP),SP      ; SAVE USER'S SP
017E' 0268'
0180' 31 0268'          00268           LD      SP,REGSP
0183' DD E5             00269           PUSH    IX              ; SAVE USER'S REGS
0185' E5                00270           PUSH    HL
0186' D5                00271           PUSH    DE
0187' C5                00272           PUSH    BC
0188' F5                00273           PUSH    AF
0189' FD 22             00274           LD      (REGIY),IY
018B' 026B'
018D' ED 57             00275           LD      A,I
018F' 32 026A'          00276           LD      (REGI),A        ; SAVE USER'S I REG
0192' ED 7B             00277           LD      SP,(STKPT)      ; RESTORE HOST'S SP
0194' 023C'
                        00278 ;
                        00279 ; CHECK IF INSTRUCTION WAS A REPEATER (OTIR ETC.), IF
                        00280 ; SO ADJUST THE R-REGISTER ACCORDINGLY.
0196' 01 0000           00281           LD      BC,00
0199' 2A 0272'          00282           LD      HL,(REPT)       ; CHECK FOR ZERO
019C' BF                00283           CP      A               ; CLEAR CARRY FLAG
019D' ED 4A             00284           ADC     HL,BC
019F' 28 11             00285           JR      Z,D00           ; IF NOT ZERO, JUMP
                        00286 ;
01A1' BF                00287           CP      A               ; ZERO CARRY FLAG
01A2' ED 4B             00288           LD      BC,(REGBC)
01A4' 0260'
01A6' ED 42             00289           SBC     HL,BC           ; # TIMES EXECUTED
01A8' 3A 026F'          00290           LD      A,(REGR)        ; GET REGR
01AB' 95                00291           SUB     L               ; DECREMENT IT
01AC' 3C                00292           INC     A               ; ADD 1 (IT GETS
                        00293                                   ; RE-DECREMENTED
                        00294                                   ; AT D3)
01AD' CB BF             00295           RES     7,A             ; ZERO THE MSB
01AF' 32 026F'          00296           LD      (REGR),A
                        00297 ;
                        00298 ;
01B2' 2A 0270'          00299 D00:      LD      HL,(REG)        ; GET SAVED REG
                        00300 ;
01B5' 3E 00             00301           LD      A,0
01B7' BC                00302           CP      H               ; IS HIGH BYTE EMPTY?
01B8' 20 03             00303           JR      NZ,D0           ; IF NOT JUMP
                        00304 ;
01BA' BD                00305           CP      L               ; IS LOW BYTE EMPTY?
01BB' 28 2F             00306           JR      Z,D3            ; IF SO  DONE
                        00307 ;
                        00308 ; RESTORE REGISTERS WHICH MAY HAVE BEEN MODIFIED
01BD' 3A 0174'          00309 D0:       LD      A,(WORK)        ; GET SIM-INSTR
01C0' FE E9             00310           CP      0E9             ; WAS IT JP(HL)?
01C2' 20 06             00311           JR      NZ,D1           ; IF NOT JUMP
                        00312 ;
01C4' 22 0264'          00313           LD      (REGHL),HL      ; REPLACE REG
01C7' 22 026D'          00314           LD      (REGPC),HL      ; UPDATE REGPC
                        00315 ;
01CA' FE DD             00316 D1:       CP      0DD             ; WAS IT JP(IX)?
01CC' 20 06             00317           JR      NZ,D2           ; IF NOT JUMP
```

```
                        00318 ;
01CE' 22 0266'          00319           LD      (REGIX),HL      ; REPLACE REG
01D1' 22 026D'          00320           LD      (REGPC),HL      ; UPDATE REGPC
                        00321 ;
01D4' FE FD             00322 D2:       CP      0FD             ; WAS IT JP(IY)?
01D6' 20 06             00323           JR      NZ,D2.5         ; IF NOT JUMP
                        00324 ;
01D8' 22 026B'          00325           LD      (REGIY),HL      ; REPLACE REG
01DB' 22 026D'          00326           LD      (REGPC),HL      ; UPDATE REGPC
                        00327 ;
01DE' ED 73             00328 D2.5:     LD      (STKPT),SP      ; WAS CALL
01E0' 023C'
01E2' ED 7B             00329           LD      SP,(REGSP)      ; GET USER SP
01E4' 0268'
01E6' D1                00330           POP     DE              ; STRIP TOP VALUE
01E7' E3                00331           PUSH    HL              ; PUSH RETURN ADDRESS
01E8' ED 7B             00332           LD      SP,(STKPT)
01EA' 023C'
                        00333 ;
                        00334 ; DECREMENT THE R REGISTER AND RETURN
01EC' 3A 026F'          00335 D3:       LD      A,(REGR)        ; GET OLD R VALUE
01EF' 3D                00336           DEC     A               ; DECREMENT IT
01F0' CB BF             00337           RES     7,A
01F2' 32 026F'          00338           LD      (REGR),A        ; PUT R BACK
01F5' E1                00339           POP     HL
01F6' D1                00340           POP     DE
01F7' C1                00341           POP     BC
01F8' F1                00342           POP     AF
01F9' C9                00343           RET
                        00344 ;
                        00345 ; THIS SECTION HANDLES THE ALTERNATE REGISTER SET
                        00346 ; SWAPS
01FA' 2A 0256'          00347 EXAF:     LD      HL,(XRAF)       ; AF'
01FD' ED 5B             00348           LD      DE,(REGAF)      ; AF
01FF' 025E'
0201' ED 53             00349           LD      (XRAF),DE       ; EXCHANGE THEM
0203' 0256'
0205' 22 025E'          00350           LD      (REGAF),HL
0208' 18 E2             00351           JR      D3
                        00352 ;
020A' 11 0250'          00353 EXX:      LD      DE,TEMP         ; TEMPORARY STORAGE
020D' 21 0258'          00354           LD      HL,XRBC         ; SOURCE
0210' 01 0006           00355           LD      BC,6
0213' ED B0             00356           LDIR
0215' 13                00357           INC     DE              ; SKIP OVER XRAF
0216' 13                00358           INC     DE
0217' 23                00359           INC     HL
0218' 23                00360           INC     HL              ; SKIP OVER REGAF
0219' 0E 06             00361           LD      C,6
021B' ED B0             00362           LDIR
021D' 21 0250'          00363           LD      HL,TEMP         ; SOURCE
0220' 13                00364           INC     DE              ; SKIP OVER REGAF
0221' 13                00365           INC     DE
0222' 0E 06             00366           LD      C,6
0224' ED B0             00367           LDIR
0226' 18 C4             00368           JR      D3
```

```
                        00369 ;
                        00370 ; HANDLES JP CC'S WHEN JUMP WAS NOT TAKEN
0228' E5                00371 CAUGHT:  PUSH    HL                ; REPLACE REGPC
0229' 2A 0231'          00372          LD      HL,(OLDPC)        ; WITH ITS NON-JUMP
022C' 22 026D'          00373          LD      (REGPC),HL        ; VALUE
022F' E1                00374          POP     HL
0230' C9                00375          RET
                        00376 ;
0231'                   00377 OLDPC:   DEFS    2
0233' 00 00 00          00378 ZERO:    DEFB    00,00,00,00,00,00,00
0236' 00 00 00
0239' 00
023A'                   00379 LEN:     DEFS    1
023B'                   00380 SAVE:    DEFS    1
023C'                   00381 STKPT:   DEFS    2
023E'                   00382 PRESAV:  DEFS    6         ; PREVIOUS REG AREA
0244'                   00383 USERHL:  DEFS    2
0246'                   00384 USERIX:  DEFS    2
0248'                   00385 USESTK:  DEFS    3
024B'                   00386 USERIY:  DEFS    5
0250'                   00387 TEMP:    DEFS    6
0256'                   00388 XRAF:    DEFS    2
0258'                   00389 XRBC:    DEFS    2
025A'                   00390 XRDE:    DEFS    2
025C'                   00391 XRHL:    DEFS    2
025E'                   00392 REGSAV:  DEFS    0         ; USER REGISTER SAVE AREA
025E'                   00393 REGAF:   DEFS    0
025E'                   00394 REGF:    DEFS    1
025F'                   00395 REGA:    DEFS    1
0260'                   00396 REGBC:   DEFS    0
0260'                   00397 REGC:    DEFS    1
0261'                   00398 REGB:    DEFS    1
0262'                   00399 REGDE:   DEFS    0
0262'                   00400 REGE:    DEFS    1
0263'                   00401 REGD:    DEFS    1
0264'                   00402 REGHL:   DEFS    0
0264'                   00403 REGL:    DEFS    1
0265'                   00404 REGH:    DEFS    1
0266'                   00405 REGIX:   DEFS    2
0268'                   00406 REGSP:   DEFS    2
026A'                   00407 REGI:    DEFS    1
026B'                   00408 REGIY:   DEFS    2
026D'                   00409 REGPC:   DEFS    2
026F'                   00410 REGR:    DEFS    1
0270'                   00411 REG:     DEFS    2
0272'                   00412 REPT:    DEFS    2
                        00413          END
```

MACROS:

SYMBOLS:

C1	00EE'	C2	00FE'	CATCH	0178'	CAUGHT	0228'	CNT1	00B3'
CONT	0087'	CSTK	013F'	DO	01BD'	DOO	01B2'	D1	01CA'
D2	01D4'	D2.5	01DE'	D3	01EC'	EINT	0173'	EXAF	01FA'
EXX	020A'	INFO	001B*	LEN	023A'	N1	0151'	N2	00C3'
N2.1	00DB'	N3	010E'	N4	012B'	OLDPC	0231'	OPSYS	0047
PRESAV	023EI'	REG	0270'	REGA	025FI'	REGAF	025EI'	REGB	0261I'
REGBC	0260I'	REGC	0260I'	REGD	0263I'	REGDE	0262I'	REGE	0262I'
REGF	025EI'	REGH	0265I'	REGHL	0264I'	REGI	026AI'	REGIX	0266I'
REGIY	026BI'	REGL	0264I'	REGPC	026DI'	REGR	026FI'	REGSAV	025EI'
REGSP	0268I'	REPT	0272'	S1	0043'	S2	004F'	S3	006B'
SAVE	023B'	SIMUL	0000I'	STKPT	023C'	TARGT	017BI'	TEMP	0250I'
USERHL	0244'	USERIX	0246'	USERIY	024B'	USESTK	0248'	WORK	0174'
XRAF	0256I'	XRBC	0258I'	XRDE	025A'	XRHL	025C'	ZERO	0233'

NO FATAL ERROR(S)