

```
        .PABS
        .PHEX

.SLIST
; *****
; * BALLY BASIC INTERPRETER *
; *
; * (C) JULY 1978 BALLY MFG *
; * (C) DEC 1980 REVISED *
; *
; * WRITTEN BY: JAY FENTON *
; *
; * BALLY BASIC IS BASED ON *
; * PALO ALTO TINY BASIC BY *
; * LICHEN WANG *
; *
; *****
;TINY BASIC INTERPRETER
;MACROS:
        .DEFINE TOKEN[TINDX,TGOTO]=
[
        .BYTE TINDX
        DEFF TGOTO
]

        .DEFINE DEFF[WORDY]=
[
        .BYTE (WORDY>8)!80H
        .BYTE WORDY&0FFH
]

        .DEFINE TSTC[CAT,DOG]=
[
        RST 1
        .BYTE 'CAT'
        .BYTE DOG-.-1
]

        .DEFINE TSTCC[CAT1,DOG1]=
[
        RST 1
        .BYTE CAT1
        .BYTE DOG1-.-1
]

        .DEFINE ITEM[STRANG,JUMPTO]=
[
        .ASCII 'STRANG'
        DEFF JUMPTO
]

4E20 BOTSCR == 04E20H
4FEA TOPSCR == 04FEAH
A000 BOTRAM == 0A000H
A70C DFILMT == 0A70CH
2000 BOTROM == 02000H

;
0015 HLTPORT == 15H ; KEYPAD KEY WITH HALT ONIT
;
000D CR == 0DH
001F RUBOUT == 1FH
002C COMMA == 44
0066 EDKEY == 66H
0067 NLLN == 67H
```

.MAIN. -

```

; EQUATES FOR RESTART INSTRUCTIONS
0002 RSTEXP == 2 ; EXPR
0003 RSTLDE == 3 ; LDE
0004 RSTIGN == 4 ; IGNBLK
0005 RSTPAR == 5 ; PARN
0006 RSTFIN == 6 ; FINISH
;
4E20 .LOC BOTSCR
4E20 TXTUNF: .BLKB 2
4E22 VARBGN: .BLKB 2*26
4E56 DEVVAR: ; DEVICE VARIABLES:
4E56 DEVCL0: .BLKB 2 ; BACKGROUND COLOR
4E58 DEVCL1: .BLKB 2 ; FOREGROUND COLOR
4E5A DEVTEM: .BLKB 2 ; TEMPO
4E5C VDMX: .BLKB 2 ; VDM X COORDINATE
4E5E VDMY: .BLKB 2 ; VDM Y COORDINATE
4E60 OLDXY: .BLKB 2 ; PREVIOUS COORDINATES FROM VECTOR DRAW
4E62 DEVMO: .BLKB 2 ; MASTER OSC
4E64 DEVOA: .BLKB 2 ; OSC A
4E66 DEVOB: .BLKB 2 ; OSC B
4E68 DEVOC: .BLKB 2 ; OSC C
4E6A DEVVD: .BLKB 2 ; VIBRATO DEPTH
4E6C DEVVR: .BLKB 2 ; VIBRATO RATE
4E6E DEVVC: .BLKB 2 ; VOL C
4E70 DEVNM: .BLKB 2 ; NOISE MODE
4E72 DEVVA: .BLKB 2 ; VOLUME A
4E74 DEVVB: .BLKB 2 ; VOLUME B
4E76 DEVNV: .BLKB 2 ; NOISE VOLUME
4E78 REMAIN: .BLKB 2 ; REMAINDER FROM LAST DIVIDE
4E7A SCRMOD: .BLKB 2 ; SCROLL MODE
4E7C VDMNLF: .BLKB 1 ; VDM NEW LINE FLAG
4E7D KEYTMR: .BLKB 1 ; KEYBOARD SCAN TIMER
4E7E MUZTMR: .BLKB 1 ; MUSIC NOTE TIMER
4E7F NEWTMR: .BLKB 1 ; NEW MUSIC TIMER VALUE
4E80 MUZMO: .BLKB 1 ; MASTER OSC FOR DICK
4E81 MUZTON: .BLKB 1 ; TONE VALUE
4E82 SHARPF: .BLKB 1 ; SHARP-FLAT
;
4E83 KEYTRK: .BLKB 1 ; KEYBOARD TRACKER
4E84 LINEND: .BLKB 2
4E86 EDFLG:
4E86 PIXVAL: .BLKB 1 ; PIXEL TO DRAW VECTOR WITH
4E87 EDPTR:
4E87 MNMX: .BLKB 2 ; MIN - MAX DELTAS FOR VECTOR DRAW
4E89 INCRO: .BLKB 2 ; COORDINATE INCREMENTS FOR VECTOR DRAW
4E8B NLLNLN: .BLKB 2 ; AUTO LINE # STUFF
4E8D NLLNCT: .BLKB 1
4E8E NLLNZS: .BLKB 1 ; AUTO LINE NUMBER ZERO SURPRESS FLAG
4E8F OLDLN: .BLKB 2 ; PREVIOUS LINE # TYPED
;
4E91 CHECKER: .BLKB 1 ; PLACE FOR CHECKSUM
4E92 HKVECT:
4E92 HKLPINT: .BLKB 3

```

.MAIN. -

```

4E95          HKINT:  .BLKB  3
4E98          CHKIO:  .BLKB  3
4E9B          OUTCH:  .BLKB  3
4E9E          STACKP: .BLKB  2      ; STACK POINTER
4EA0          ALTFON: .BLKB  7      ; ALTERNATE FONT DESCRIPTOR
4EA7          OLDCUR: .BLKB  2
4EA9          CURRNT: .BLKB  2
4EAB          STKGOS: .BLKB  2
4EAD          VARNXT == .
4EAD          STKINP: .BLKB  2
4EAF          LOPVAR: .BLKB  2
4EB1          LOPINC: .BLKB  2
4EB3          LOPLMT: .BLKB  2
4EB5          LOPLN:  .BLKB  2
4EB7          LOPPT:  .BLKB  2
4EB9          .BLKB  1
4EBA          XQTBUF:
4EBA          BUFFER: .BLKB 104
4F22          BUFEND == .
4F22          .BLKB  72
4F6A          STKLMT == .
4FEA          .LOC   TOPSCR
4FEA          STACK == .
A000          .LOC   BOTRAM
A000          TEXT:  .BLKB  2
2000          .LOC   BOTROM
2000 C3 24F7   JMP    BEGIN ; ** AUTOSTART CASSETTE **
2003 80       PIXTBL: .BYTE 080H
2004 20       .BYTE 020H
2005 08       .BYTE 08H
2006 02       .BYTE 2H
; TRANSFER VECTORS TO RESTART ROUTINES
2007 C3 2AAE   JMP    TSTCH ; * RST 8
200A C3 27FA   JMP    EXPR  ; * RST 16
200D C3 2FC6   JMP    LDE   ; * RST 24
2010 C3 2F57   JMP    IGNBLK ; * RST 32
2013 C3 28FC   JMP    PARN  ; * RST 40
2016 F1       POP    PSW  ; * RST 48
2017 C3 29AC   JMP    FINISH
201A 574841543F WHAT: .ASCII 'WHAT?'
201F 0D       .BYTE  CR
2020 4E92     .WORD  HKLPINT
2022 4E95     ITAB:  .WORD  HKINT
; INITIAL VALUES FOR PARAMETER VECTOR
2024 0007     INIDEV: .WORD  7      ; BACKGROUND COLOR
2026 0000     .WORD  0      ; FOREGROUND COLOR
2028 0002     .WORD  2      ; MUSIC TEMPO
202A FFB3     .WORD -77     ; VDM X COORDINATE
202C 0028     .WORD  40     ; VDM Y COORDINATE
; BIT BANGER GOODIES FOLLOW:
3C00         BANGIN = 3C00H ; BIT BANGER READ PORT
3800         BANG1  = 3800H ; BIT BANG CODE TO WRITE A ONE
3C00         BANGO  = 3C00H ; BIT BANG CODE TO WRITE A ZERO

```

```

        .DEFINE WASTE[TIME,%LAB]=[
            MVI A,TIME
%LAB:   DCR A
            JRNZ %LAB]

; :PRINT COMMAND
; IF VARIABLE NAME, BLOCKSIZE GIVEN, WE WILL WRITE
; OUT THE SPECIFIED BLOCK RATHER THAN THE PROGRAM STORAGE AREA
202E   CD 22C9   TOUTPU: CALL   IGNATNL ; ANY ARGS?
2031   280E           JRZ   YYSPGM ; JUMP TO SAVE PGM IF SO
2033   CD 22F3   CALL   TSTVFF ; ELSE GET START ADDR
2036   E5           PUSH  H      ; SAVE THAT
2037   CF           TSTCC  COMMA,BADSAV ; CHECK FOR COMMA[      RST      1
2038   2C           +       .BYTE  COMMA
2039   34           +       .BYTE  BADSAV  --1
        +]

203A   D7           RST    RSTEXP ; GET BLOCK SIZE
203B   29           DAD    H      ; CONVERT TO BYTES
203C   EB           XCHG
203D   E3           XTHL           ; PUSH DE ON STACK
203E   EB           XCHG           ; DE=START, HL=SIZE
203F   1807         JMPR   YYOUTB ; JUMP TO OUTPUTER
2041   21 0E91     YYSPGM: LXI   H,CHECKER-4000H
2044   D5           PUSH  D
2045   11 4000     LXI   D,4000H
        ; SAVE PROGRAM ON TAPE
2048   FF           YYOUTB: EMUSIC 1[INTP%[.IFE .INTP.,[RST 7]]
2049   15           +.BYTE 20+1]
204A   F3           DI
204B   CD 212A     CALL   LEADER ; WRITE OUT SOME LEADER
204E   3EA5         MVI   A,0A5H
2050   CD 20F5     CALL   OUTBYA
2053   DF           ..OBL: RST   RSTLDE
2054   CD 20F5     CALL   OUTBYA ; TWEEDLE IT OUT
2057   13           INX   D      ; BUMP BLOCK PTR
2058   2B           DCX   H      ; DECREMENT BLOCK SIZE
2059   7C           MOV   A,H    ; LOOP END YET?
205A   B5           ORA   L
205B   20F6         JRNZ  ..OBL
205D   DF           RST   RSTLDE
205E   78           MOV   A,B    ; OUTPUT CHECKSUM
205F   2F           CMA           ; COMPLEMENT FOR LATER TEST
2060   CD 20F5     CALL   OUTBYA
2063   0602         MVI   B,2
2065   DF           RST   RSTLDE
2066   E3           XTHL
2067   E3           XTHL
2068   CD 2132     CALL   LEADR1 ; PUT OUT TRAILER
206B   D1           POP   D
206C   FB           EI
206D   F7           RST   RSTFIN ; BYE BYE
206E   C3 29C6     BADSAV: JMP   QWHAT
2071   D5           TVLIST: PUSH  D
    
```

.MAIN. -

```

2072  CD 2098          CALL    TVLLNK
2075  D1              POP     D
2076  F7              RST     RSTFIN
                ; SPECIAL ENTRY TO LOAD COMBINED SCREEN AND PGM

                ; :INPUT COMMAND
                ; IF VARIABLE ADDRESS IS GIVEN, WE WILL INPUT
                ; THE BLOCK INTO THE SPECIFIED AREA, OTHERWISE
                ; WE HANDLE IT LIKE A PROGRAM
2077  CD 22C9        TINPUT: CALL    IGNATNL      ; ANY ARGS?
207A  21 4000        LXI     H,4000H
207D  C4 22F3        CNZ     TSTVFF      ; GET VAR ADDR
2080  D5              PUSH    D
2081  CD 208A        CALL    INBLK
2084  D1              XXELOD: POP    D
2085  F7              RST     RSTFIN
                ; : RUN COMMAND - LOADS BOOTSTRAP INTO RAM
                ; AND JUMPS TO IT
2086  21 4000        TLOAD: LXI     H,4000H ; HL= SCREEN TOP
2089  E5              PUSH    H
                ; SUBROUTINE TO INPUT A BLOCK, HL=STORE ADDR
                ; FIRST - AN ENTRY TO REVEAL FEEDBACK AREA
208A  CD 20A9        INBLK: CALL    SENWAI
                ; LOOP TO GRAB CHARS AND STORE EM
208D  CD 20B7        ZZCHRL: CALL   INCHAR
2090  280E           JRZ     ZZEOT
2092  CD 2FE2        CALL    STHL
2095  23              INX     H
2096  18F5           JMPR   ZZCHRL
2098  CD 20A9        TVLLNK: CALL   SENWAI
209B  CD 20B7        ZZKIL:  CALL   INCHAR
209E  20FB           JRNZ   ZZKIL
20A0  2B              ZZEOT:  DCX   H
20A1  FB              EI
20A2  14              INR     D          ; SHOULD HAVE BEEN FF
20A3  C8              RZ
20A4  3E3F           OUTCHQ: MVI   A,'?'
20A6  C3 4E9B        JMP     OUTCH

20A9  FF              SENWAI: EMUSIC 1[INTP%[.IFE .INTP.,[RST 7]]
20AA  15              +.BYTE 20+1]
20AB  F3              DI
20AC  CD 20B7        ..SENW: CALL   INCHAR ; WAIT FOR THE SENTINEL
20AF  28FB           JRZ     ..SENW
20B1  FEA5           CPI     0A5H
20B3  20F7           JRNZ   ..SENW
20B5  57              MOV     D,A
20B6  C9              RET
                ;INCHAR CLOBBERS A, BC, DE
20B7  INCHAR:        ;NZ IF NO TIMEOUT, Z IF TIMEOUT, CHAR IN C
20B7  3A 3C02        LDA     BANGIN+2
20BA  E601           ANI     1
20BC  5F              MOV     E,A          ; PRIME THE PUMP

```

.MAIN. -

```

20BD 01 0810      LXI B,810H      ;8 BITS, 10=TIMEOUT FACTOR
20C0 CD 20D9      ..SBW:  CALL INBIT   ;WAIT FER START BIT
20C3 3004          JRNC ..GETL   ;GOTIT
20C5 0D           DCR C         ;TIMEOUT?
20C6 20F8          JRNZ ..SBW    ;NOT YET
20C8 C9           RET          ;Z SET
20C9 CD 20D9      ..GETL:  CALL INBIT
20CC DC 20D9          CC          INBIT   ; GET ANOTHER 1
20CF CB19          RARR C
20D1 10F6          DJNZ ..GETL   ;GET 8 BITS
20D3 79           MOV      A,C     ; UPDATE CHECKSUM
20D4 82           ADD      D
20D5 57           MOV      D,A
20D6 05           DCR      B       ; SET NONZERO
20D7 79           MOV      A,C     ; RETURN VALUE
20D8 C9           RET          ;RETURN
20D9              INBIT:
                ; CHECK FOR ABORT LOAD KEY
20D9 DB15          IN          HLTPORT
20DB 0F           RRC
20DC DA 2531       JC          INIT0
20DF 3A 3C00       LDA          BANGIN
20E2 AB           XRA          E       ; CHECK FOR CHANGE
20E3 0F           RRC
20E4 30F3          JRNC          INBIT   ; NO - WAIT
                WASTE 23[
20E6 3E17          + MVI A,23
20E8 3D           + ..0001: DCR A
20E9 20FD          + JRNZ ..0001]
20EB 3A 3C01       LDA          BANGIN+1
20EE E601          ANI          1
20F0 BB           CMP          E
20F1 5F           MOV          E,A
20F2 C8           RZ
20F3 37           STC
20F4 C9           RET
                ;OUTBYT CLOBBERS A, BC
20F5 4F           OUTBYA: MOV      C,A     ; GET CHAR FROM A
20F6 80           ADD      B       ; ADD CHECKSUM ACCUM
20F7 47           MOV      B,A     ; AND SAVE
20F8 C5           PUSH     B
20F9 CD 214C       CALL WRZERO   ;WRITE START BIT
                WASTE 14
                ;VERY TIME SENSITIVE[
20FC 3E0E          + MVI A,14
20FE 3D           + ..0002: DCR A
20FF 20FD          + JRNZ ..0002]
2101 0608          MVI      B,8     ; WRITE 8 DATA, 1 STOP
2103 37           ..WRL:  STC
2104 CB19          RARR      C     ; GET BIT, INSERT 1 FOR STOP
2106 380C          JRC ..WRL1    ;IF ONE, WRITE ONE
2108 CD 214C       CALL WRZERO   ;ELSE WRITE ZERO
                WASTE 12[
210B 3E0C          + MVI A,12

```

.MAIN. -

```

210D 3D      +..0003: DCR A
210E 20FD    +      JRNZ ..0003]
2110 1800    JMPR      ..NXT
2112 1808    ..NXT: JMPR  ..WRE      ;LOOP COUNTER
2114 CD 2140 ..WR1: CALL WRONE      ;WRITE ONE-BIT
                WASTE 32[
2117 3E20    +      MVI A,32
2119 3D      +..0004: DCR A
211A 20FD    +      JRNZ ..0004]
211C 10E5    ..WRE: DJNZ ..WRL      ;TILL 8 BITS DONE
211E 1800    JMPR      ..SEX
2120 C1      ..SEX: POP      B
2121 CD 2140 CALL      WRONE      ; WRITE A ONE BIT FOR STOP
                WASTE 24[
2124 3E18    +      MVI A,24
2126 3D      +..0005: DCR A
2127 20FD    +      JRNZ ..0005]
2129 C9      RET

                ;LEADER CLOBBERS BC AND A
212A 060F    LEADER: MVI      B,15      ; APROX 3 SECS
212C          LEADR2: WASTE 32[
212C 3E20    +      MVI A,32
212E 3D      +..0006: DCR A
212F 20FD    +      JRNZ ..0006]
2131 00      NOP
2132 CD 2140 LEADR1: CALL WRONE      ;LEADER IS ALL ONES
2135 0B      DCX B
2136 78      MOV A,B
2137 B1      ORA C
2138 20F2    JRNZ      LEADR2
                WASTE 29[
213A 3E1D    +      MVI A,29
213C 3D      +..0007: DCR A
213D 20FD    +      JRNZ ..0007]
213F C9      RET

                ;WRONE WRITES ONE HALF CYCLE OF ONE-BIT (1/1200 SEC)
2140 3A 3800 WRONE: LDA BANG1      ;CHANGE ITS STATE
                WASTE 36      ; WAIT SOME, THEN FALL INTO ...[
2143 3E24    +      MVI A,36
2145 3D      +..0008: DCR A
2146 20FD    +      JRNZ ..0008]
2148 3A 3C00 LDA      BANG0
214B C9      RET

                ;WRZERO WRITES ONE HLAF CYCLE OF ZERO BIT (1/2400 SEC)
214C 3A 3800 WRZERO: LDA BANG1
                WASTE 17[
214F 3E11    +      MVI A,17

```

.MAIN. -

```

2151 3D      +..0009: DCR A
2152 20FD    +      JRNZ ..0009]
2154 00      NOP
2155 00      NOP
2156 3A 3C00 LDA BANG0
2159 C9      RET

;
;
;
215A 7C      CKHLDE: MOV   A,H
215B AA      XRA   D
215C F2 2160 JP    COMP
215F EB      XCHG

; ...
2160 7C      COMP:  MOV   A,H
2161 BA      CMP   D
2162 C0      RNZ
2163 7D      MOV   A,L
2164 BB      CMP   E
2165 C9      RET
2166 484F573F HOW:   .ASCII  'HOW?'
216A 0D      .BYTE  CR
216B 534F525259 SORRY: .ASCII  'SORRY'
2170 0D      .BYTE  CR

;
; TABLE GIVING JUMP TO ADDRESS FOR COMMANDS
2171 263C    TOKJT:  .WORD  LIST
2173 22FA    .WORD  CLRSCR
2175 260B    .WORD  RUN
2177 273E    .WORD  NEXT
2179 23AC    .WORD  LINEDR
217B 2600    .WORD  IFF
217D 2631    .WORD  GOTO
217F 26AB    .WORD  GOSUB
2181 26D2    .WORD  RETURN
2183 2304    .WORD  BOXDRW
2185 26EB    .WORD  FOR
2187 27AB    .WORD  INPUT
2189 2670    .WORD  PRINT

; INITIAL HOOK VECTOR ITEMS
218B C3 228C  HOOKER: JMP   LPINT
218E C3 21FD  JMP   TBIINT
2191 C3 2E0E  JMP   XCHKIO
2194 C3 2D0B  JMP   XOUTCH
2197 4FEA    .WORD  STACK

; TABLE GIVING ASCII CHARS FOR TOKENS
2199 TOKTXT:
2199 4C4953    .ASCII  'LIS'
219C D4        .BYTE  'T'+80H
219D 434C4541 .ASCII  'CLEA'
21A1 D2        .BYTE  'R'+80H
21A2 5255    .ASCII  'RU'
21A4 CE      .BYTE  'N'+80H

```



.MAIN. -

```

21A5 4E4558      .ASCII      'NEX'
21A8 D4           .BYTE      'T'+80H
21A9 4C494E      .ASCII      'LIN'
21AC C5           .BYTE      'E'+80H
21AD 49           .BYTE      'I'
21AE C6           .BYTE      'F'+80H
21AF 474F54      .ASCII      'GOT'
21B2 CF           .BYTE      'O'+80H
21B3 474F5355    .ASCII      'GOSU'
21B7 C2           .BYTE      'B'+80H
21B8 5245545552  .ASCII      'RETUR'
21BD CE           .BYTE      'N'+80H
21BE 424F        .ASCII      'BO'
21C0 D8           .BYTE      'X'+80H
21C1 464F        .ASCII      'FO'
21C3 D2           .BYTE      'R'+80H
21C4 494E5055    .ASCII      'INPU'
21C8 D4           .BYTE      'T'+80H
21C9 5052494E    .ASCII      'PRIN'
21CD D4           .BYTE      'T'+80H
21CE 535445      .ASCII      'STE'
21D1 D0           .BYTE      'P'+80H
21D2 524E        .ASCII      'RN'
21D4 C4           .BYTE      'D'+80H
21D5 54           .BYTE      'T'
21D6 CF           .BYTE      'O'+80H

```

```

;
; DEVICE VARIABLE TABLE
; THIS TABLE IS IN INVERSE ORDER OF APPEARENCE IN MEMORY

```

```

0013 PARNUM == 19
21D7 DEVLST:
21D7 13      .BYTE      'S'-'@'
21D8 4D      .BYTE      'M'
21D9 12      .BYTE      'R'-'@'
21DA 4D      .BYTE      'M'
21DB 0E      .BYTE      'N'-'@'
21DC 56      .BYTE      'V'
21DD 16      .BYTE      'V'-'@'
21DE 42      .BYTE      'B'
21DF 16      .BYTE      'V'-'@'
21E0 41      .BYTE      'A'
21E1 0E      .BYTE      'N'-'@'
21E2 4D      .BYTE      'M'
21E3 16      .BYTE      'V'-'@'
21E4 43      .BYTE      'C'
21E5 16      .BYTE      'V'-'@'
21E6 46      .BYTE      'F'
21E7 16      .BYTE      'V'-'@'
21E8 52      .BYTE      'R'
21E9 14      .BYTE      'T'-'@'
21EA 43      .BYTE      'C'
21EB 14      .BYTE      'T'-'@'
21EC 42      .BYTE      'B'

```

.MAIN. -

```

21ED 14          .BYTE  'T'-'@'
21EE 41          .BYTE  'A'
21EF 0D          .BYTE  'M'-'@'
21F0 4F          .BYTE  'O'
21F1 18          .BYTE  'X'-'@'
21F2 59          .BYTE  'Y'
21F3 03          .BYTE  'C'-'@'
21F4 59          .BYTE  'Y'
21F5 03          .BYTE  'C'-'@'
21F6 58          .BYTE  'X'
21F7 0E          .BYTE  'N'-'@'
21F8 54          .BYTE  'T'
21F9 06          .BYTE  'F'-'@'
21FA 43          .BYTE  'C'
21FB 02          .BYTE  'B'-'@'
21FC 43          .BYTE  'C'

; TINY BASIC INTERRUPT ROUTINE
21FD F5          TBIINT: PUSH  PSW      ; SAVE REGISTERS
21FE C5          PUSH  B
21FF E5          PUSH  H

; DEAL WITH KEYBOARD SCAN TIMER
2200 21 4E7D     LXI    H,KEYTMR
2203 7E          MOV    A,M
2204 A7          ANA    A
2205 2801        JRZ    TBIN0
2207 35          DCR    M
2208 23          TBIN0: INX   H

; HAS MUSIC TIMER COUNTED DOWN?
2209 7E          MOV    A,M
220A A7          ANA    A
220B 2809        JRZ    TBIN1 ; YEP - PLAY NEXT NOTE
220D 35          DCR    M ; ELSE DECREMENT IT
220E 2026        JRNZ   TBIN3 ; JUMP IF NOT NOW ZERO
2210 AF          XRA    A
2211 32 4E64     STA    DEVOA
2214 181D        JMPR   TBIN2

; MUSIC TIMER IS AT ZERO - ARE NEW PARAMETERS READY?
2216 23          TBIN1: INX   H ; STEP TO NEW TIMER VALUE
2217 B6          ORA    M ; IS IT NON ZERO?
2218 281C        JRZ    TBIN3 ; JUMP IF NOT
221A 3600        MVI    M,0 ; SAY WE GOT IT
221C FA 2236     JM     TBIN3 ; IF MINUS UPDATE NOTHING
221F 2B          DCX    H ; ELSE SET OFFICIAL TIMER
2220 77          MOV    M,A
2221 23          INX   H
2222 23          INX   H
2223 7E          MOV    A,M ; SET NEW MASTER
2224 32 4E62     STA    DEVMO
2227 3647        MVI    M,OA2
2229 23          INX   H
222A 7E          MOV    A,M ; AND NEW TONE
222B 32 4E64     STA    DEVOA
222E A7          ANA    A ; REST WANTED?

```

.MAIN. -

```

222F 2805          JRZ     TBIN3   ; YES - JUMP AROUND VOLUME UPDATE
2231 3E0F          MVI     A,15
2233 32 4E72      TBIN2:  STA     DEVVA
                ; SET COLOR REGISTERS TO VALUES IN PARAMETER VARS %0 AND %1
2236 3A 4E56      TBIN3:  LDA     DEVCL0
2239 D304          OUT     COL0L
223B D305          OUT     COL1L
223D 3A 4E58      LDA     DEVCL1
2240 D306          OUT     COL2L
2242 D307          OUT     COL3L
                ; UPDATE THE MUSIC PROCESSOR
2244 3A 4E5A      LDA     DEVTEM
2247 07           RLC
2248 383F          JRC     INTDON
224A 01 0410      LXI     B,410H
224D 21 4E62      LXI     H,DEVMO
2250 7E           ..LP1:  MOV     A,M
2251 ED79          OUTP    A
2253 23           INX     H
2254 23           INX     H
2255 0C           INR     C
2256 10F8         DJNZ    ..LP1
2258 46           MOV     B,M   ; B=VD
2259 23           INX     H
225A 23           INX     H
225B 7E           MOV     A,M   ; A=VR
225C 0F           RRC
225D 0F           RRC
225E A8           XRA     B
225F E6C0         ANI     0C0H
2261 A8           XRA     B
2262 ED79          OUTP    A
2264 23           INX     H
2265 23           INX     H
2266 46           MOV     B,M   ; B=VOLC
2267 23           INX     H
2268 23           INX     H
2269 7E           MOV     A,M   ; A=NM
226A 07           RLC
226B 07           RLC
226C 07           RLC
226D 07           RLC
226E A8           XRA     B
226F E630         ANI     30H
2271 A8           XRA     B
2272 D315         OUT     15H
2274 23           INX     H
2275 23           INX     H
2276 46           MOV     B,M   ; VA
2277 23           INX     H
2278 23           INX     H
2279 7E           MOV     A,M   ; VB
227A 07           RLC

```

.MAIN. -

```

227B 07          RLC
227C 07          RLC
227D 07          RLC
227E A8          XRA      B
227F E6F0        ANI      0F0H
2281 A8          XRA      B
2282 D316        OUT      16H
2284 23          INX      H
2285 23          INX      H
2286 7E          MOV      A,M      ; GET NOISE VOLUME
2287 D317        OUT      17H
                ; DONE - RESTORE REGISTERS AND GO BACK
2289 E1          INTDON: POP   H
228A C1          POP     B
228B F1          POP     PSW
228C FB          LPINT:  EI
228D C9          RET
                ; COMMAND TO SILENCE MUSIC PORTS
228E            SILENCE:
228E D5          PUSH     D
228F FF          FILL     1[INTP%[.IFE .INTP.,[RST 7]]
2290 1B          +.BYTE  26+1]
2291 4E64        .WORD   DEVOA
2293 0014        .WORD   20
2295 00          .BYTE   0
2296 D1          POP     D
2297 F7          RST     RSTFIN
                ; ROUTINE TO MOVE PROGRAM LINE FROM PGM STORAGE AREA
                ; INTO EXECUTION BUFFER
2298 2A 4EA9      EXPAND: LHLD  CURRNT
229B ED4B 4EA7   LBCD   OLDCUR
229F A7          ANA     A
22A0 ED42        DSBC   B
22A2 C8          RZ
22A3 2A 4EA9      EXPMAN: LHLD  CURRNT
22A6 CB7C        BIT    7,H      ; IN LINE BUFFER ALREADY?
22A8 C8          RZ          ; YES - KICKOUT
22A9 22 4EA7      SHLD   OLDCUR
22AC 23          INX     H
22AD 23          INX     H
22AE 29          DAD     H
22AF 01 4EBA     LXI    B,XQTBUF
22B2 7E          ..EXP1: MOV   A,M
22B3 07          RLC
22B4 23          INX     H
22B5 AE          XRA     M
22B6 E6AA        ANI     0AAH
22B8 AE          XRA     M
22B9 02          STAX   B
22BA 23          INX     H
22BB 03          INX     B
22BC FE0D        CPI     CR
22BE 20F2        JRNZ   ..EXP1

```

.MAIN. -

```

22C0 37          STC
22C1 CB1C       RARR   H
22C3 CB1D       RARR   L
22C5 22 4E84    SHLD   LINEND
22C8 C9         RET
; SUBROUTINE TO RETURN ZERO STATUS IF CHARACTER IN A IS NL OR
; ';'
22C9 E7         IGNATNL: RST   RSTIGN ; IGNORE ANY BLANKS
22CA FE3B       ATNL:  CPI   ';' ; CHECK FOR CONTINUATION
22CC C8         RZ
22CD FE0D       CPI   CR ; AND FOR CR
22CF C9         RET
; FUNCTION TO RETURN STATE OF ADDRESSED PIXEL
; IE... PIX(X,Y)= 1 IF PIXEL IS 1, 0 IF 0
22D0 CF         PIXFUN: TSTC  '(' ,PIXDUD[ RST 1
22D1 28         + .BYTE  '('
22D2 24         + .BYTE  PIXDUD-. -1
+ ]
22D3 C5         PUSH   B
22D4 D7         RST   RSTEXP
22D5 E5         PUSH   H
22D6 CF         TSTCC  COMMA,PIXDUD[ RST 1
22D7 2C         + .BYTE  COMMA
22D8 1E         + .BYTE  PIXDUD-. -1
+ ]
22D9 D7         RST   RSTEXP
22DA CF         TSTC  ')',PIXDUD[ RST 1
22DB 29         + .BYTE  ')'
22DC 1A         + .BYTE  PIXDUD-. -1
+ ]
22DD C1         POP    B
22DE D5         PUSH   D ; SAVE PTR
22DF 55         MOV    D,L
22E0 59         MOV    E,C
22E1 CD 24A9    CALL   R2A
22E4 EB         XCHG
22E5 FF         INDEXB 1[INTP%[.IFE .INTP.,[RST 7]]
22E6 5D         +.BYTE  92+1]
22E7 2003       .WORD  PIXTBL
22E9 1A         LDAX   D ; GET BYTE FROM SCREEN
22EA A6         ANA    M ; MASK OFF NONSENSE
22EB 2600       MVI    H,0
22ED 6C         MOV    L,H
22EE D1         POP    D
22EF C1         POP    B
22F0 C8         RZ
22F1 23         INX    H
22F2 C9         RET
; SUBROUTINE TO GET VARIABLE MAKING SURE IT IS ONE
22F3 CD 2A49    TSTVFF: CALL  TSTV
22F6 D0         RNC ; GO BACK IF GOOD
; ELSE FALL INTO ...
22F7 C3 29C6    PIXDUD: JMP   QWHAT

```

.MAIN. -

```

22FA  CD 2DE2      CLRSCR: CALL  CLRENT
22FD  21 0000          LXI   H,0
2300  22 4E60          SHLD  OLDXY
2303  F7              RST   RSTFIN
      ; BOX DRAW ROUTINE
2304  D7      BOXDRW: RST   RSTEXP ; GET X
2305  E5          PUSH  H
2306  CF          TSTCC  COMMA,BOXDUD ; FIND COMMA[ RST 1
2307  2C      +      .BYTE  COMMA
2308  54      +      .BYTE  BOXDUD -. -1
      +]
2309  D7          RST   RSTEXP ; GET Y
230A  E5          PUSH  H
230B  CF          TSTCC  COMMA,BOXDUD[ RST 1
230C  2C      +      .BYTE  COMMA
230D  4F      +      .BYTE  BOXDUD-. -1
      +]
230E  D7          RST   RSTEXP
230F  7D          MOV   A,L
2310  F5          PUSH  PSW
2311  CF          TSTCC  COMMA,BOXDUD[ RST 1
2312  2C      +      .BYTE  COMMA
2313  49      +      .BYTE  BOXDUD-. -1
      +]
2314  D7          RST   RSTEXP
2315  45          MOV   B,L
2316  C5          PUSH  B
2317  CF          TSTCC  COMMA,BOXDUD[ RST 1
2318  2C      +      .BYTE  COMMA
2319  43      +      .BYTE  BOXDUD-. -1
      +]
231A  D7          RST   RSTEXP
231B  D5          PUSH  D
231C  DDE1        POP   X
231E  C1          POP   B ; RESTORE YS
231F  F1          POP   PSW ; AND XS
2320  4F          MOV   C,A
2321  7D          MOV   A,L ; PRESERVE FLAG
2322  E1          POP   H
2323  55          MOV   D,L
2324  E1          POP   H
2325  5D          MOV   E,L
2326  6F          MOV   L,A
      ; NOW WE HAVE: B=YS, C=XS, D=Y, E=X, L=FLAG
      ; LIMIT CHECK Y
2327  60          MOV   H,B
2328  CB3C        SRLR  H
232A  7A          MOV   A,D
232B  CD 2360      CALL  SABS
232E  84          ADD   H
232F  FE2D        CPI   45
2331  3026        JRNC  BOXNDR
2333  78          MOV   A,B ; DIVIDE SIZE AGAIN

```

.MAIN. -

```

2334 3D          DCR    A      ; THIS TIME WITH PRESUB
2335 CB3F       SRLR   A
2337 82        ADD    D
2338 57        MOV    D,A
                ; AND X
2339 61        MOV    H,C
233A CB3C       SRLR   H
233C 7B        MOV    A,E
233D CD 2360    CALL   SABS
2340 84        ADD    H
2341 FE51       CPI    81
2343 3014       JRNC  BOXNDR
2345 7B        MOV    A,E
2346 94        SUB    H
2347 5F        MOV    E,A
                ; DIDDLE WITH FLAG BYTE
2348 7D        MOV    A,L
2349 E603       ANI    3      ; MODULO 4
234B 280C       JRZ   BOXNDR ; SKIP DRAW IF ZERO
234D D602       SUI    2      ; ELSE SUBTRACT 2 FOR MASK
234F F5        BOXDR1: PUSH  PSW
2350 CD 24A9    CALL   R2A
                ; HL = ABS ADDR, A = SA, B=YS, C=XS
2353 D30C       OUT    MAGIC
2355 F1        POP    PSW
2356 CD 2365    CALL   BOXPUT
2359 DDE5       BOXNDR: PUSH  X
235B D1        POP    D
235C F7        RST   RSTFIN
235D C3 29C6    BOXDUD: JMP  QWHAT
2360 A7        SABS:  ANA   A
2361 F0        RP
2362 2F        CMA
2363 3C        INR   A
2364 C9        RET
                ; SUBROUTINE TO DRAW A BOX ON SCREEN
2365 5F        BOXPUT: MOV  E,A
2366 79        MOV   A,C    ; D = X / 4
2367 0F        RRC
2368 0F        RRC
2369 E63F       ANI   3FH
236B 3C        INR   A
236C 57        MOV   D,A
                ; PAINT FULL BYTE STRIPES
236D 15        MPT1:  DCR   D
236E 2807       JRZ   MPT2
2370 3EAA       MVI   A,10101010B
2372 CD 238A    CALL   STRIPE
2375 18F6       JMPR  MPT1
2377 79        MPT2:  MOV  A,C
2378 E603       ANI   3
237A 3C        INR   A
237B 4F        MOV   C,A

```

.MAIN. -

```

237C  AF          XRA    A
237D  0D          MPT3:  DCR    C
237E  2806        JRZ    MPT4
2380  0F          RRC
2381  0F          RRC
2382  F680        ORI    10000000B
2384  18F7        JMPR   MPT3
2386  CD 238A    MPT4:  CALL  STRIPE
2389  AF          XRA    A
          ; FALL INTO ...
          ; SUBROUTINE TO PAINT A STRIPE
238A  E5          STRIPE: PUSH  H
238B  C5          PUSH  B
238C  32 0FFF    STA  URINAL
238F  3A 4FFF    LDA  URINAL+4000H
2392  4F          MOV  C,A
2393  7B          STRP1: MOV  A,E
2394  FE01        CPI   1
2396  2002        JRNZ  STRP2
2398  7E          MOV  A,M
2399  A9          XRA  C
239A  AE          STRP2: XRA  M
239B  A1          ANA  C
239C  AE          XRA  M
239D  77          MOV  M,A
239E  7D          MOV  A,L
239F  C628        ADI  BYTEPL
23A1  6F          MOV  L,A
23A2  7C          MOV  A,H
23A3  CE00        ACI  0
23A5  67          MOV  H,A
23A6  10EB        DJNZ  STRP1
23A8  C1          POP  B
23A9  E1          POP  H
23AA  23          INX  H
23AB  C9          RET
          ; LINE DRAWER
23AC  D7          LINEDR: RST  RSTEXP
23AD  7D          MOV  A,L
23AE  F5          PUSH PSW
23AF  CF          TSTCC COMMA,LINED4[  RST    1
23B0  2C          +    .BYTE COMMA
23B1  27          +    .BYTE LINED4-.-1
          +]
23B2  D7          RST  RSTEXP
23B3  7D          MOV  A,L
23B4  F5          PUSH PSW
23B5  CF          LINED1: TSTCC COMMA,LINED4[  RST    1
23B6  2C          +    .BYTE COMMA
23B7  21          +    .BYTE LINED4-.-1
          +]
23B8  D7          RST  RSTEXP
23B9  4D          MOV  C,L

```



.MAIN. -

```

23BA D5          PUSH   D
23BB DDE1        POP    X
23BD ED5B 4E60   LDED   OLDXY
23C1 F1          POP    PSW
23C2 67          MOV    H,A
23C3 F1          POP    PSW
23C4 6F          MOV    L,A
23C5 22 4E60     SHLD   OLDXY ; SET NEW LAST PLACE
                ; DIDDLE WITH FLAG BYTE
23C8 79          MOV    A,C
23C9 E603        ANI    3
23CB 2808        JRZ    LINED3
23CD D602        SUI    2
23CF 32 4E86     LINED2: STA   PIXVAL ; SET PIXVAL
23D2 CD 23DC     CALL   DVECT
23D5 DDE5        LINED3: PUSH  X
23D7 D1          POP    D
23D8 F7          RST    RSTFIN
23D9 C3 2AE6     LINED4: JMP   QHOW
                ; LARRY LIVERMORE'S VECTOR DRAWING ALGORITHM
                ; H=Y1, L=X1, D=Y2, E=X2
23DC D5          DVECT: PUSH  D
23DD 45          MOV    B,L
23DE 4B          MOV    C,E
23DF CD 247B     CALL   CDELTA
23E2 58          MOV    E,B
23E3 69          MOV    L,C
23E4 44          MOV    B,H
23E5 4A          MOV    C,D
23E6 CD 247B     CALL   CDELTA
23E9 61          MOV    H,C
23EA 50          MOV    D,B
                ; WE NOW HAVE: H=SGN(DY), L=SGN(DX)
                ; D=ABS(DY), E=ANS(DX)
23EB 22 4E89     SHLD   INCRO
                ; DECIDE WHICH DELTA IS LARGER
                ; CALL BIGGER MX, SMALLER MN
23EE 0E00        MVI    C,0
23F0 7A          MOV    A,D
23F1 BB          CMP    E
23F2 3803        JRC    VECT1
23F4 53          MOV    D,E
23F5 5F          MOV    E,A
23F6 0C          INR    C
23F7 7A          VECT1: MOV   A,D ; MX TO A
23F8 CB3F        SRLR   A
23FA 47          MOV    B,A
23FB EB          XCHG
23FC 22 4E87     SHLD   MNMX
23FF D1          POP    D
2400 7D          MOV    A,L
2401 3C          INR    A ; MAKE SURE LAST PIXEL WRITTEN
                ; THE INFAMOUS PIXEL PAINTING LOOP

```

.MAIN. -

```

2402  F5          VECT2:  PUSH   PSW
2403  CD 249D      CALL   R2ACLP
2406  3019        JRNC   VECT2A
2408  C5          PUSH   B
2409  E5          PUSH   H
240A  4F          MOV    C,A
240B  0600        MVI    B,0
240D  21 2003     LXI    H,PIXTBL
2410  09          DAD    B
2411  46          MOV    B,M
2412  E1          POP    H
2413  3A 4E86     LDA    PIXVAL
2416  FE01        CPI    1
2418  2002        JRNZ   VECT9
241A  7E          MOV    A,M
241B  A8          XRA    B
241C  AE          VECT9:  XRA    M
241D  A0          ANA    B
241E  AE          XRA    M
241F  77          MOV    M,A
2420  C1          POP    B
          ; INCREMENT COORDINATES
2421  2A 4E87     VECT2A: LHLD  MNMX
2424  78          MOV    A,B
2425  84          ADD    H
2426  B8          CMP    B          ; DID WRAP AROUND UNIVERSE?
2427  3803        JRC    ..FUZZ
2429  BD          CMP    L
242A  380D        JRC    VECT4
242C  95          ..FUZZ: SUB  L
242D  47          MOV    B,A
242E  2A 4E89     LHLD  INCRO
2431  7A          MOV    A,D
2432  84          ADD    H
2433  57          MOV    D,A
2434  7B          VECT3:  MOV    A,E
2435  85          ADD    L
2436  5F          MOV    E,A
2437  180B        JMPR  VECT5
2439  47          VECT4:  MOV    B,A
243A  2A 4E89     LHLD  INCRO
243D  79          MOV    A,C
243E  0F          RRC
243F  30F3        JRNC   VECT3
2441  7A          MOV    A,D
2442  84          ADD    H
2443  57          MOV    D,A
          ; END OF LOOP
2444  F1          VECT5:  POP    PSW
2445  3D          DCR    A
2446  20BA        JRNZ   VECT2
2448  C9          RET
          ; SUBROUTINE TO LOAD HL WITH VDM COORDINATES

```

.MAIN. -

```

; FROM DEVICE VARIABLES
2449 F5 LDVDMC: PUSH PSW
244A 3A 4E5E LDA VDMY
244D 2F CMA
244E C629 ADI 41
2450 FE51 CPI 81 ; OUT OF RANGE?
2452 3801 JRC LDVDM1 ; NO
2454 AF XRA A
2455 67 LDVDM1: MOV H,A
2456 3A 4E5C LDA VDMX ; DIDDLE WITH X
2459 C64D ADI 77
245B FE9D CPI 157
245D 3801 JRC LDVDM2
245F AF XRA A
2460 6F LDVDM2: MOV L,A
2461 F1 POP PSW
2462 C9 RET

; SUBROUTINE TO STORE HL INTO VDM COORDINATE DEVICE VARIABLES
2463 E5 STVDMC: PUSH H
2464 7C MOV A,H
2465 D629 SUI 41
2467 2F CMA
2468 6F MOV L,A
2469 CD 293E CALL SGNEXT
246C 22 4E5E SHLD VDMY
246F E1 POP H
2470 7D MOV A,L
2471 D64D SUI 77
2473 6F MOV L,A
2474 CD 293E CALL SGNEXT
2477 22 4E5C SHLD VDMX
247A C9 RET

; SUBROUTINE TO COMPUTE DELTA AND INCREMENT FOR TWO COORDINATES
247B E5 CDELTA: PUSH H
247C D5 PUSH D
247D 69 MOV L,C
247E CD 293E CALL SGNEXT
2481 EB XCHG
2482 68 MOV L,B
2483 CD 293E CALL SGNEXT
2486 AF XRA A
2487 ED52 DSBC D

; COMPUTE SGN(DELTA) AND ABS(DELTA)
2489 B4 ORA H
248A 2807 JRZ CDELT1
248C 4F MOV C,A
248D 7D MOV A,L
248E 2F CMA
248F 3C INR A
2490 47 MOV B,A
2491 1807 JMPR CDELT3
2493 B5 CDELT1: ORA L ; POS CASE 0?
2494 2802 JRZ CDELT2

```

.MAIN. -

```

2496 3E01          MVI    A,1
2498 45           CDELT2: MOV    B,L
2499 4F           MOV    C,A
249A D1           CDELT3: POP    D
249B E1           POP    H
249C C9           RET
          ; RELATIVE TO ABSOLUTE CONVERSION WITH CLIPPING
249D 7B           R2ACLP: MOV   A,E
249E C650         ADI    80
24A0 FE00         CPI    160    ; IN RANGE 0-159 - CY FOR OK
24A2 D0           RNC
24A3 7A           MOV    A,D
24A4 C62C         ADI    44
24A6 FE58         CPI    88
24A8 D0           RNC
          ; ...
          ; RELATIVE TO ABSOLUTE CONVERSION
24A9 D5           R2A:   PUSH   D
24AA 7A           MOV    A,D
24AB 2F           CMA
24AC C62C         ADI    44
24AE 57           MOV    D,A
24AF 7B           MOV    A,E
24B0 C650         ADI    80
24B2 5F           MOV    E,A
24B3 AF           XRA    A
24B4 FF           RELAB1[INTP%[.IFE .INTP.,[RST 7]]
24B5 3A           +.BYTE 58+0]
24B6 EB           XCHG
24B7 D1           POP    D
24B8 37           STC
24B9 C9           RET
          ; KB - FUNCTION TO RETURN NEXT CHARACTER FROM KEYBOARD
24BA C5           GETKB:  PUSH   B
24BB D5           PUSH   D
24BC CD 4E98      CALL   CHKIO
24BF D1           POP    D
24C0 C1           KBLNKX: POP   B
24C1 6F           MOV    L,A
24C2 2600         MVI    H,0
24C4 C9           RET
          ; DEVICE VARIABLE TO OUTPUT TO REFERENCED IO PORT
24C5 EF           PUTIO:  RST    RSTPAR ; GET PORT #
24C6 CF           TSTC   '=' ,PUTCD2    ; GET EQUALS[ RST    1
24C7 3D           + .BYTE  '='
24C8 25           + .BYTE  PUTCD2  -. -1
          +]
24C9 E5           PUSH   H    ; SAVE PORT #
24CA D7           RST    RSTEXP ; EVALUATE EXPRESSION FOLLOWING
24CB 7D           MOV    A,L  ; A=VALUE TO OUTPUT
24CC E1           POP    H    ; RESTORE PORT #
24CD C5           PUSH   B
24CE 44           MOV    B,H

```

.MAIN. -

```

24CF  4D          MOV     C,L
24D0  ED79        OUTP   A      ; IT   1
24D2  C1          POP    B
24D3  F7          RST    RSTFIN ; GO HOME
          ; FUNCTION TO RETURN VALUE OF A GIVEN IO PORT
24D4  EF          IOFUN: RST    RSTPAR ; GET PORT NUMBA
24D5  C5          PUSH   B
24D6  44          MOV    B,H
24D7  4D          MOV    C,L
24D8  ED78        INP    A
24DA  18E4        JMPR   KBLNKX
          ; DEVICE VARIABLE TO PLAY NOTE WITHOUT PRINTING
24DC  CF          PUTMU: TSTC  '=' ,PUTCD2[ RST  1
24DD  3D          +     .BYTE  '='
24DE  0F          +     .BYTE  PUTCD2--1
          +]
24DF  D7          RST    RSTEXP
24E0  7D          MOV    A,L
24E1  CD 2EBE     CALL   PNOTE
24E4  F7          RST    RSTFIN
          ; DEVICE VARIABLE TO OUTPUT CHARACTER ON VDM
24E5  CF          PUTCD: TSTC  '=' ,PUTCD2[ RST  1
24E6  3D          +     .BYTE  '='
24E7  06          +     .BYTE  PUTCD2--1
          +]
24E8  D7          RST    RSTEXP
24E9  7D          MOV    A,L
24EA  CD 4E9B     CALL   OUTCH
24ED  F7          RST    RSTFIN
24EE  C3 29C6     PUTCD2: JMP    QWHAT
          ; ROUTINE TO TRANSFER CONTROL TO ASSEMBLY LANGUAGE SUBROUTINE
24F1  21 25DD     DOCALL: LXI   H,BBRET ; PUSH RETURN ADDR ON STACK
24F4  E5          PUSH   H
24F5  D7          RST    RSTEXP ; GET ADDRESS
24F6  E9          PCHL           ; AND JUMP TO IT
          ; ** TINY BASIC EXECUTION STARTS HERE **
          ; CLEAR WHOLE KIT AND KABOODLE
24F7  AF          BEGIN: XRA    A
24F8  D30C        OUT    MAGIC
24FA  67          MOV    H,A
24FB  6F          MOV    L,A
24FC  77          MOV    M,A ; MAKE SURE SHIFTER FLUSHED
24FD  77          BEGIN1: MOV   M,A
24FE  23          INX    H
24FF  CB64        BIT    4,H
2501  28FA        JRZ    BEGIN1
2503  31 4FCE     LXI    SP,SYSRAM
2506  FF          ENTER[RST 7
2507  00          +.BYTE 0
0001          +.INTP.=1]
          SETOUT 1[INTP%[.IFE .INTP.,[RST 7]]
2508  17          +.BYTE 22+1]
2509  B0          .BYTE 176

```

.MAIN. -

```

250A 2C          .BYTE 44
250B 18          .BYTE 18H
          ; INITIALIZE DEVICE VARIABLES
          MOVE 1[INTP%[.IFE .INTP.,[RST 7]]]
250C 5F          +.BYTE 94+1]
250D 4E56        .WORD DEVVAR
250F 000A        .WORD 10
2511 2024        .WORD INIDEV
          MOVE 1[INTP%[.IFE .INTP.,[RST 7]]]
2513 5F          +.BYTE 94+1]
2514 4EA0        .WORD ALTFON
2516 0007        .WORD 7
2518 0206        .WORD FNTSYS
          MOVE 1[INTP%[.IFE .INTP.,[RST 7]]]
251A 5F          +.BYTE 94+1]
251B 4E92        .WORD HKVECT
251D 000E        .WORD 14
251F 218B        .WORD HOOKER
          SETW 1[INTP%[.IFE .INTP.,[RST 7]]]
2521 7D          +.BYTE 124+1]
2522 06A0        .WORD 6A0H
2524 4EA0        .WORD ALTFON
          SETW 1[INTP%[.IFE .INTP.,[RST 7]]]
2526 7D          +.BYTE 124+1]
2527 A004        .WORD TEXT+4
2529 4E20        .WORD TXTUNF
          SETW 1[INTP%[.IFE .INTP.,[RST 7]]]
252B 7D          +.BYTE 124+1]
252C 5555        .WORD 5555H
252E 4002        .WORD 4002H ; TEXT+2
2530 02          EXIT[.BYTE 2
0000          +.INTP.=0]
2531 F3          INIT0: DI
2532 ED5E        IM2
2534 3E20        MVI A,ITAB>8
2536 ED47        STAI
2538 3E22        MVI A,ITAB&0FFH
253A D30D        OUT INFBK
253C 3EC8        MVI A,200
253E D30F        OUT INLIN
2540 FB          EI
2541 CD 2D06     INIT: CALL CRLF
          ; DIRECT COMMAND - TEXT COLLECTOR
2544          TELL:
2544          STOP:
2544 2A 4E9E     RSTART: LHLD STACKP
2547 F9          SPHL
2548 21 254F     LXI H,XXST1+1
254B 22 4EA9     SHLD CURRNT
254E 21 0000     XXST1: LXI H,0
2551 22 4EA7     SHLD OLDCUR
2554 22 4EAF     SHLD LOPVAR
2557 22 4EAB     SHLD STKGOS

```

.MAIN. -

```

255A 3E3E      XXST2: MVI    A, '>'
255C CD 2C5F      CALL   GETLN
255F D5         PUSH   D
2560 11 4EBA     LXI    D, BUFFER
2563 CD 2ABF      CALL   TSTNUM
2566 E7         RST    RSTIGN
2567 7C         MOV    A, H
2568 B5         ORA    L
2569 C1         POP    B
256A 284C      JRZ    EXEC0
256C 22 4E8F     SHLD  OLDLN
256F 1B         DCX    D
2570 7C         MOV    A, H
2571 12         STAX  D
2572 1B         DCX    D
2573 7D         MOV    A, L
2574 12         STAX  D
2575 C5         PUSH  B
2576 D5         PUSH  D
2577 79         MOV    A, C
2578 93         SUB    E
2579 F5         PUSH  PSW
257A CD 2A04     CALL  FNDLN
257D D5         PUSH  D
257E 2010      JRNZ  XXST3
2580 D5         PUSH  D
2581 CD 2A1D     CALL  FNDNXT
2584 C1         POP    B
2585 2A 4E20     LHLD  TXTUNF
2588 CD 2AED     CALL  MVUP
258B 60         MOV    H, B
258C 69         MOV    L, C
258D 22 4E20     SHLD  TXTUNF
2590 C1         XXST3: POP    B
2591 2A 4E20     LHLD  TXTUNF
2594 F1         POP    PSW
2595 E5         PUSH  H
2596 FE03      CPI    3
2598 28AA      JRZ    RSTART
259A 85         ADD    L
259B 5F         MOV    E, A
259C 3E00      MVI    A, 0
259E 8C         ADC    H
259F 57         MOV    D, A
25A0 21 A70C     LXI    H, DFTLMT
25A3 EB         XCHG
25A4 CD 2160     CALL  COMP
25A7 D2 29FE     JNC   QSORRY
25AA 22 4E20     SHLD  TXTUNF
25AD D1         POP    D
25AE CD 2AFD     CALL  MVDOWN
25B1 D1         POP    D
25B2 E1         POP    H

```

.MAIN. -

```

25B3  CD 2AED          CALL    MVUP
25B6  18A2             JMPR    XXST2
                ; DIRECT AND EXEC
25B8  E7             EXEC0:  RST    RSTIGN ; GET FIRST    NONBLANK
25B9  D5             PUSH    D      ;SAVE POINTER
25BA  D668           SUI    68H    ; IS SHE A TOKEN?
25BC  3813           JRC    EXEC0A ; NO
25BE  FE0D           CPI    0DH
25C0  300F           JRNC   EXEC0A
                ; WE FOUND A TOKEN - LOOKUP IN TABLE AND JUMP TO IT
25C2  07             RLC
25C3  5F             MOV    E,A
25C4  1600           MVI    D,0
25C6  21 2171        LXI    H,TOKJT
25C9  19             DAD    D
25CA  5E             MOV    E,M
25CB  23             INX    H
25CC  56             MOV    D,M
25CD  EB             XCHG
25CE  D1             POP    D
25CF  13             INX    D
25D0  E9             PCHL
                ; NOT A TOKEN - A VARIABLE PERHAPS?
25D1  CD 2A49        EXEC0A: CALL   TSTV   ;TEST FOR VARIABLE
25D4  3808           JRC    EXEC0B ; NO - SEARCH 1
25D6  CF             TSTC   '=' ,EXEC0B ; MAYBE[      RST    1
25D7  3D             +      .BYTE  '='
25D8  05             +      .BYTE  EXEC0B  -. -1
                +]
25D9  C1             POP    B      ; THROW OUT OLD PTR
25DA  CD 29A3        CALL   SETV1  ; ASSIGNMENT 1
25DD  F7             BBRET:  RST   RSTFIN
25DE  D1             EXEC0B:  POP   D
25DF  21 2BCB        LXI    H,TAB2-1
25E2  E7             EXEC:   RST   RSTIGN ;EXEC
25E3  D5             PUSH   D      ;SAVE POINTER
25E4  1A             EX1:   LDAX  D      ; ZAPPED LDE
25E5  13             INX    D
25E6  23             INX    H
25E7  BE             CMP    M
25E8  28FA           JRZ    EX1
25EA  3E7F           MVI    A,07FH
25EC  1B             DCX    D
25ED  BE             CMP    M
25EE  3808           JRC    EX5
25F0  23             EX2:   INX    H
25F1  BE             CMP    M
25F2  30FC           JRNC   EX2
25F4  23             INX    H
25F5  D1             POP    D
25F6  18EA           JMPR   EXEC
25F8  7E             EX5:   MOV    A,M      ;LOAD HL WITH THE JUMP
25F9  23             INX    H      ;ADDRESS FROM TABLE

```



.MAIN. -

```

25FA 6E          MOV     L,M
25FB E67F        ANI     07FH
25FD 67          MOV     H,A
25FE F1          POP     PSW
25FF E9          PCHL

; IF AND REM
2600 D7          IFF:    RST     RSTEXP
2601 7C          MOV     A,H
2602 B5          ORA     L
2603 2027        JRNZ    RUNSML
2605            REM:
2605 ED5B 4E84    RUNNXL: LDED   LINEND
2609 1803        JMPR   RUNX1
260B 11 A000     RUN:    LXI   D,TEXT
260E 21 0000     RUNX1: LXI   H,0
2611 CD 2A0C     CALL   FNDLP
2614 DA 2544     JC     RSTART
2617 ED53 4EA9    RUNTSL: SDED   CURRNT
261B DB14        IN     14H
261D 6F          MOV     L,A
261E DB15        IN     15H
2620 A5          ANA     L
2621 FE20        CPI     20H
2623 CC 2B4C     CZ     PRTLNS
2626            ..OK:
2626 CD 2298     CALL   EXPAND
2629 11 4EBA     LXI   D,XQTBUF
262C CD 2E79     RUNSML: CALL  WHATSU ; CHECK FOR INTERRUPT KEY
262F 1887        JMPR   EXECO
2631 D7          GOTO:   RST     RSTEXP
2632 D5          PUSH   D
2633 CD 2A04     CALL   FNDLN
2636 C2 2AE7     JNZ    AHOW
2639 F1          POP     PSW
263A 18DB        JMPR   RUNTSL

; LIST AND PRINT
; NEW - IMPROVED LIST COMMAND
; LETS YOU PUT IT IN A PROGRAM
263C 21 0000     LIST:  LXI   H,0 ; ASSUME AT EOL
263F CD 22C9     CALL   IGNATNL
2642 2805        JRZ    LS3
2644 FE2C        CPI     ',' ; LEADING COMMA?
2646 2801        JRZ    LS3 ; YEP - SKIP FIRST      EXPR GET

; NOT AT END - GET FIRST      EXPR
2648 D7          LS2:   RST     RSTEXP
2649 E5          LS3:   PUSH   H
264A 21 FFFF     LXI   H,0FFFFH
264D CF          TSTCC COMMA,LS4[ RST 1
264E 2C          +     .BYTE COMMA
264F 01          +     .BYTE LS4-.-1
                +]

2650 D7          RST     RSTEXP
2651 D5          LS4:   PUSH   D

```

.MAIN. -

```

2652 FDE1          POP      Y
2654 E3           XTHL
2655 CD 2A04      CALL      FNDLN
2658 3812        LS5:    JRC      LSQUIT
265A E3           XTHL
265B 7C           MOV      A,H
265C B5           ORA      L
265D 280D        JRZ      LSQUIT
265F 2B           DCX      H
2660 E3           XTHL
2661 CD 2B4C      CALL      PRTLNS
2664 CD 2E79      CALL      WHATSU
2667 CD 2A0C      CALL      FNDLP
266A 18EC        JMPR     LS5
266C FDE5        LSQUIT:  PUSH   Y
266E D1           POP      D
266F F7           RST     RSTFIN
2670 0E08        PRINT:  MVI     C,8      ; C=# OF SPACES
2672 CF          TSTCC   59,PR1   ; IF NULL LIST & ";"[ RST      1
2673 3B          +       .BYTE  59
2674 05          +       .BYTE  PR1    -.-1
                +]
2675 CD 2D06      CALL      CRLF    ; GIVE CR-LF AND
2678 18B2        JMPR     RUNSML   ; CONTINUE SAME LINE
267A CF          PR1:    TSTCC   CR,PR6   ; IF NULL LIST (CR)[ RST      1
267B 0D          +       .BYTE  CR
267C 15          +       .BYTE  PR6    -.-1
                +]
267D CD 2D06      CALL      CRLF    ; ALSO GIVE CR-LF AND
2680 C3 29BC      JMP      IMCHEK   ; GO TO NEXT LINE IF POSSIBLE
2683 CF          PR2:    TSTC   '#',PR4  ;ELSE IS IT FORMAT?{ RST      1
2684 23          +       .BYTE  '#'
2685 04          +       .BYTE  PR4    -.-1
                +]
2686 D7          PR3:    RST     RSTEXP   ; YES,EVALUATE EXPR.
2687 4D          MOV      C,L      ; AND SAVE IT IN C
2688 1805        JMPR     PR5      ; LOOK FOR MORE TO PRINT
268A CD 2B5D      PR4:    CALL     QTSTG   ; OR IS IT A STRING?
268D 1814        JMPR     PR9      ; IF NOT, MUST BE EXPR.
268F CF          PR5:    TSTCC   COMMA,PR8   ; IF COMMA,GO FIND NEXT[ RST      1
2690 2C          +       .BYTE  COMMA
2691 0D          +       .BYTE  PR8    -.-1
                +]
2692 CF          PR6:    TSTCC   COMMA,PR7[ RST      1
2693 2C          +       .BYTE  COMMA
2694 05          +       .BYTE  PR7-.-1
                +]
2695 CD 2BC7      CALL      SPOUTCH
2698 18F8        JMPR     PR6
269A CD 29B1      PR7:    CALL     FIN      ; IN THE LIST.
269D 18E4        JMPR     PR2      ; LIST CONTINUES
269F CD 2D06      PR8:    CALL     CRLF    ; LIST ENDS
26A2 F7          RST     RSTFIN

```

.MAIN. -

```

26A3 D7 PR9: RST RSTEXP ; EVALUATE THE EXPR
26A4 C5 PUSH B
26A5 CD 2B7F CALL PRTRNUM ; PRINT THE VALUE
26A8 C1 POP B
26A9 18E4 JMPR PR5 ; PRINT THE VALUE
; * *****
; *
; * *** GOSUB *** & RETURN***
; *
; * 'GOSUB EXPR;' OR 'GOSUB EXPR (CR)' IS LIKE THE 'GOTO' COMMAND,
; * EXCEPT THAT THE CURRENT TEXT POINTER, STACK POINTER ETC. ARE SAVED
; * SO THAT EXECUTION CAN BE CONTINUED AFTER THE SUBROUTINE 'RETURN'.
; * IN ORDER THAT 'GOSUB' CAN BE NESTED (AND EVEN RECURSIVE), THE SAVE
; * AREA MUST BE STACKED. THE STACK POINTER IS SAVED IN 'STKGOS'. THE
; * OLD 'STKGOS' IS SAVED IN THE STACK. IF WE ARE IN THE MAIN ROUTINE,
; * 'STKGOS' IS ZERO (THIS WAS DONE BY THE "MAIN" SECTION OF CODE).
; * BUT WE STILL SAVE IT AS A FLAG FOR NO FURTHER 'RETURN'S.
; *
; * 'RETURN(CR)' UNDOES EVERYTHING THAT 'GOSUB' DID, AND THUS RETURN THE
; * EXECUTION TO THE COMMAND AFTER THE MOST RECENT 'GOSUB'. IF 'STKGOS'
; * IS ZERO, IT INDICATES THAT WE NEVER HAD A 'GOSUB' AND IS THUS AN
; * ERROR
; *
GOSUB:
26AB LDA CURRNT+1 ; DISALLOW FROM COMMAND LINE
26AB 3A 4EAA
26AE 07 RLC
26AF D2 2AE6 JNC QHOW
26B2 CD 2B27 CALL PUSHA ; SAVE THE CURRENT "FOR"
26B5 D7 RST RSTEXP ; PARAMETERS
26B6 D5 PUSH D ; AND TEXT POINTER
26B7 CD 2A04 CALL FNLDN ; FIND THE TARGET LINE
26BA C2 2AE7 JNZ AHOW ; NOT THERE. SAY "HOW?"
26BD 2A 4EA9 LHLD CURRNT ; SAVE OLD
26C0 E5 PUSH H ; 'CURRNT' OLD 'STKGOS'
26C1 2A 4EAB LHLD STKGOS
26C4 E5 PUSH H
26C5 21 0000 LXI H,0 ; AND LOAD NEW ONES
26C8 22 4EAF SHLD LOPVAR
26CB 39 DAD SP
26CC 22 4EAB SHLD STKGOS
26CF C3 2617 JMP RUNTSL ; THEN RUN THAT LINE
26D2 2A 4EAB RETURN: LHLD STKGOS ;OLD STACK POINTER
26D5 7C MOV A,H ; 0 MEANS NOT EXIST
26D6 B5 ORA L
26D7 CA 29C6 JZ QWHAT ; SO, WE SAY: "WHAT?"
26DA F9 SPHL ; ELSE, RESTORE IT
26DB E1 RESTO: POP H
26DC 22 4EAB SHLD STKGOS ; AND THE OLD 'STKGOS'
26DF E1 POP H
26E0 22 4EA9 SHLD CURRNT ; AND THE OLD 'CURRNT'
26E3 D1 POP D
26E4 CD 2B0C CALL POPA
26E7 CD 2298 CALL EXPAND

```

.MAIN. -

```

26EA  F7          RST      RSTFIN
; * *****
; *
; * FOR *** & NEXT ****
; *
; * 'FOR' HAS TWO FORMS: 'FOR VAR=EXP1 TO EXP2 STEP EXP3' AND
; * 'FOR VAR=EXP1 TO EXP2'. THE SECOND FORM MEANS THE SAME THING
; * AS THE FIRST FORM WITH EXP3=1, (I.E. WITH A STEP OF +1) TBI
; * WILL FIND THE VARIABLE VAR. AND SET ITS VALUE TO THE CURRENT
;
; * VALUE OF EXP1. IT ALSO EVALUATES EXP2 AND EXP3 AND SAVES ALL
;
; * THESE TOGETHER WITH THE TEXT POINTER ETC. IN THE 'FOR' SAVE AREA
; * WHICH CONSISTS OF 'LOPVAR', 'LOPINC', 'LOPLMT', 'LOPLN', AND 'LOPPT'.
; * IF THERE IS ALREADY SOMETHING IN THE SAVE AREA (INDICATED BY A
; * NON-ZERO 'LOPVAR'), THEN THE OLD SAVE AREA IS SAVED IN THE STACK
; * BEFORE THE NEW ONE OVERWRITES IT. TBI WILL THEN DIG IN THE STACK
; * AND FIND OUT IF THIS SAME VARIABLE WAS USED IN ANOTHER CURRENTLY
; * ACTIVE 'FOR' LOOP. IF THAT IS THE CASE, THEN THE OLD 'FOR' LOOP IS
; * DEACTIVATED. (PURGED FROM THE STACK..)
; *
; * 'NEXT VAR' SERVES AS THE LOGICAL (NOT NECESSARILY PHYSICAL) END OF
; * THE 'FOR' LOOP. THE CONTROL VARIABLE VAR. IS CHECKED WITH THE
; * 'LOPVAR'. IF THEY ARE NOT THE SAME, TBI DIGS IN THE STACK TO FIND
; * THE RIGHT ONE AND PURGES ALL THOSE THAT DID NOT MATCH. EITHER WAY,
; * TBI THEN ADDS THE 'STEP' TO THAT VARIABLE AND CHECKS THE RESULT WITH
; * THE LIMIT. IF IT IS WITHIN THE LIMIT, CONTROL LOOPS BACK TO THE
; * COMMAND FOLLOWING THE 'FOR'. IF OUTSIDE THE LIMIT, THE SAVE AREA IS
; * PURGED AND EXECUTION CONTINUES.
; *
26EB  CD 2B27     FOR:     CALL    PUSHA   ;SAVE THE OLD SAVE AREA
26EE  CD 299D     CALL    SETVAL  ; SET THE CONTROL VAR.
26F1  2B         DCX     H       ; HL IS ITS ADDRESS
26F2  22 4EAF     SHLD   LOPVAR  ; SAVE THAT
26F5  CF         TSTCC  77H,FR1A   ; TO? ; LOOK FOR WORD "TO"[ RST 1
26F6  77         +      .BYTE  77H
26F7  01         +      .BYTE  FR1A  --1
;
+ ]
26F8  D7         FR1:    RST     RSTEXP  ; EVALUATE THE LIMIT
26F9  22 4EB3     FR1A:   SHLD   LOPLMT  ; SAVE THAT
26FC  21 0001     LXI     H,1
26FF  CF         TSTCC  75H,FR4  ; STEP?[ RST 1
2700  75         +      .BYTE  75H
2701  01         +      .BYTE  FR4   --1
;
+ ]
2702  D7         RST     RSTEXP
2703  22 4EB1     FR4:    SHLD   LOPINC   ; SAVE THAT TOO
2706  2A 4EA9     LHL    CURRNT  ; SAVE CURRENT LINE #
2709  22 4EB5     SHLD   LOPLN   ; AND TEXT POINTER
270C  EB         XCHG

```

.MAIN. -

```

270D 22 4EB7          SHLD  LOPPT
2710 01 000A          LXI   B,10      ; DIG INTO STACK TO
2713 2A 4EAF          LHLD  LOPVAR   ; FIND 'LOPVAR'
2716 EB              XCHG
2717 60              MOV   H,B
2718 68              MOV   L,B      ; HL=0 NOW
2719 39              DAD   SP      ; HERE IS THE STACK
271A 1801            JMPR  FR6
271C 09              DAD   B        ; EACH LEVEL IS 10 DEEP
271D 7E              MOV   A,M      ; GET THAT OLD 'LOPVAR'
271E 23              INX   H
271F B6              ORA   M
2720 2817            JRZ   FR7      ; 0 SAYS NO MORE IN IT
2722 7E              MOV   A,M
2723 2B              DCX   H
2724 BA              CMP   D        ; SAME AS THIS ONE?
2725 20F5            JRNZ  FR5
2727 7E              MOV   A,M      ; THE OTHER HALF?
2728 AB              XRA   E
2729 20F1            JRNZ  FR5
272B EB              XCHG   ; YES, FOUND ONE
272C 67              MOV   H,A
272D 6F              MOV   L,A
272E 39              DAD   SP      ; TRY TO MOVE SP
272F 44              MOV   B,H
2730 4D              MOV   C,L
2731 21 000A          LXI   H,10
2734 19              DAD   D
2735 CD 2AFD          CALL  MVDOWN   ; AND PURGE 10 WORDS
2738 F9              SPHL   ; IN THE STACK
2739 2A 4EB7          LHLD  LOPPT   ; JOB DONE, RESTORE DE
273C EB              XCHG
273D F7              RST   RSTFIN  ; AND CONTINUE
273E CD 2A49          NEXT: CALL  TSTV   ; GET ADDRESS OF VAR.
2741 DA 29C6          JC    QWHAT   ; NO VARIABLE, "WHAT?"
2744 22 4EAD          SHLD  VARNXT  ; YES, SAVE IT
2747 D5              NX1:  PUSH D    ; SAVE TEXT POINTER
2748 EB              XCHG
2749 2A 4EAF          LHLD  LOPVAR  ;GET VAR. IN 'FOR'
274C 7C              MOV   A,H
274D B5              ORA   L        ; 0 SAYS NEVER HAD ONE
274E CA 29C7          JZ    AWHAT   ; SO WE ASK: "WHAT?"
2751 CD 2160          CALL  COMP    ; ELSE WE CHECK THEM
2754 2809            JRZ   NX2      ; OK, THEY AGREE
2756 D1              POP   D        ; NO, LET'S SEE
2757 CD 2B0C          CALL  POPA    ; PURGE CURRENT LOOP
275A 2A 4EAD          LHLD  VARNXT  ; AND POP ONE LEVEL
275D 18E8            JMPR  NX1      ; GO CHECK AGAIN
275F EB              NX2:  XCHG   ; COME HERE WHEN AGREED
2760 DF              RST   RSTLDE  ; DE=VALUE OF VAR.
2761 6F              MOV   L,A
2762 13              INX   D
2763 DF              RST   RSTLDE

```

```

2764 67          MOV     H,A
2765 EB          XCHG
2766 2A 4EB1     LHLD   LOPINC
2769 E5          PUSH   H
276A 7C          MOV     A,H
276B AA          XRA     D      ; S=SIGN OF DIFFER
276C 7A          MOV     A,D      ; A=SIGN OF DE
276D 19          DAD     D      ; ADD ONE STEP
276E FA 2775     JM      NX3      ; CANNOT OVERFLOW
2771 AC          XRA     H      ; MAY OVERFLOW
2772 FA 2798     JM      NX5      ; AND IT DID
2775 EB          NX3:   XCHG
2776 2A 4EAF     LHLD   LOPVAR ; PUT IT BACK
2779 CD 2FDC     CALL  STDEHL
277C 2A 4EB3     LHLD   LOPLMT ; HL=LIMIT
277F F1          POP     PSW    ; OLD HL
2780 07          RLC      ; EXAMINE SIGN BIT
2781 3001        JRNC   NX4      ; IF POS SKIP XCHG
2783 EB          XCHG
2784 CD 215A     NX4:   CALL  CKHLDE ; COMPARE WITH LIMIT
2787 D1          POP     D      ; RESTORE TEXT POINTER
2788 3810        JRC     NX6      ; OUTSIDE LIMIT
278A 2A 4EB5     LHLD   LOPLN  ; WITHIN LIMIT, GO
278D 22 4EA9     SHLD  CURRNT ; BACK TO THE SAVED
2790 2A 4EB7     LHLD   LOPPT  ; 'CURRNT' AND TEXT
2793 EB          XCHG      ; POINTER
2794 CD 2298     CALL  EXPAND
2797 F7          RST     RSTFIN
2798 E1          NX5:   POP     H      ; OVERFLOW, PURGE
                ; RESTO LINKS IN HERE
2799 D1          NXXX:  POP     D      ; GARBAGE IN STACK
279A CD 2B0C     NX6:   CALL  POPA   ; PURGE THIS LOOP
279D F7          RST     RSTFIN

; *
; *****
; *
; * IF *** INPUT *** & LET (& DEFLT) ****
; *
; *
; * 'IF' IS FOLLOWED BY AN EXPR. AS A CONDITION AND ONE OR MORE COMMANDS
; * (INCLUDING OTHER 'IF'S) SEPARATED BY SEMI-COLONS. NOTE THAT THE
; * WORD 'THEN' IS NOT USED. TBI EVALUATES THE EXPR. IF IT IS NON-ZERO,
; * EXECUTION CONTINUES. IF THE EXPR. IS ZERO, THE COMMANDS THAT
; * FOLLOW ARE IGNORED AND EXECUTION CONTINUES AT THE NEXT LINE.
; *
; * 'INPUT' COMMAND IS LIKE THE 'PRINT' COMMAND, AND IS FOLLOWED BY A
; * LIST OF ITEMS. IF THE ITEM IS A STRING IN SINGLE OR DOUBLE QUOTES,
; * OR IS AN UP-ARROW, IT HAS THE SAME EFFECT AS IN 'PRINT'. IF AN ITEM
; * IS A VARIABLE, THIS VARIABLE NAME IS PRINTED OUT FOLLOWED BY A
; * COLON. THEN TBI WAITS FOR AN EXPR. TO BE TYPED IN. THE VARIABLE IS
; * THEN SET TO THE VALUE OF THIS EXPR. IF THE VARIABLE IS PRECEDED BY
; * A STRING (AGAIN IN SINGLE OR DOUBLE QUOTES), THE STRING WILL BE
; * PRINTED FOLLOWED BY A COLON. TBI THEN WAITS FOR INPUT EXPR. AND

```

```

; * SETS THE VARIABLE TO THE VALUE OF THE EXPR.
; *
; * IF THE INPUT EXPR. IS INVALID, TBI WILL PRINT "WHAT?", "HOW?", OR
; * "SORRY" AND REPRINT THE PROMPT AND REDO THE INPUT. THE EXECUTION
; * WILL NOT TERMINATE UNLESS YOU TYPE CONTROL-C. THIS IS HANDLED IN
; * 'INPERR'.
; *
279E 2A 4EAD INPERR: LHL D STKINP ; *** INPERR ***
27A1 F9 SPHL ; RESTORE OLD SP
27A2 E1 POP H ; AND OLD 'CURRNT'
27A3 22 4EA9 SHLD CURRNT
27A6 D1 POP D ; AND OLD TEXT POINTER
27A7 D1 POP D ; REDO INPUT
27A8 CD 22A3 CALL EXPMAN ; EXPAND THAT LINE OUT
27AB INPUT == .
27AB D5 IP1: PUSH D ; SAVE IN CASE OF ERROR
27AC CD 2B5D CALL QTSTG ; IS NEXT ITEM A STRING?
27AF 1820 JMPR IP8 ; NO
27B1 CD 2A49 IP2: CALL TSTV ; YES, BUT FOLLOWED BY A
27B4 3814 JRC IP5 ; M6IVARIABLE? NO.
27B6 CD 27DF IP3: CALL IP12
27B9 11 4EBA LXI D,BUFFER ; POINTS TO BUFFER
27BC D7 RST RSTEXP ; EVALUATE INPUT
27BD D1 POP D ; OK, GET OLD HL
27BE EB XCHG
27BF CD 2FDC CALL STDEHL
27C2 E1 IP4: POP H ; GET OLD 'CURRNT'
27C3 22 4EA9 SHLD CURRNT
27C6 D1 POP D ; AND OLD TEXT POINTER
27C7 CD 22A3 CALL EXPMAN
27CA F1 IP5: POP PSW ; PURGE JUNK IN STACK
27CB CF IP6: TSTCC COMMA,IP7 ; IS NEXT CH. ","? [ RST 1
27CC 2C + .BYTE COMMA
27CD 02 + .BYTE IP7 -. -1
+]
27CE 18DB JMPR INPUT ; YES, MORE ITEMS.
27D0 F7 IP7: RST RSTFIN
27D1 D5 IP8: PUSH D ; SAVE FOR 'PRSTG'
27D2 CD 2A49 CALL TSTV ; MUST BE VARIABLE NOT
27D5 DA 29C6 JC QWHAT ; "WHAT?" IT IS NOT?
27D8 43 IP11: MOV B,E
27D9 D1 POP D
27DA CD 2B75 CALL PRTCHS ; PRINT THOSE AS PROMPT
27DD 18D7 JMPR IP3 ; YES, INPUT VARIABLE
27DF C1 IP12: POP B ; RETURN ADDRESS
27E0 D5 PUSH D ; SAVE TEXT POINTER
27E1 EB XCHG
27E2 2A 4EA9 LHL D CURRNT ; ALSO SAVE 'CURRNT'
27E5 E5 PUSH H
27E6 21 27AB LXI H,IP1 ; A NEGATIVE NUMBER
27E9 22 4EA9 SHLD CURRNT ; AS A FLAG
27EC 21 0000 LXI H,0 ; SAVE SP TOO
27EF 39 DAD SP
    
```

.MAIN. -

```

27F0  22 4EAD          SHLD  STKINP
27F3  D5                PUSH  D      ; OLD HL
27F4  C5                PUSH  B
27F5  3E20             MVI   A, ' '
27F7  C3 2C5F          JMP   GETLN  ; AND GET A LINE
; *****
; *
; ** EXPR **
; *
; * 'EXPR' EVALUATES ARITHMETICAL OR LOGICAL EXPRESSIONS.
; * <EXPR>::=<EXPR1>
; *      <EXPR1><REL.OP><EXPR1>
; * WHERE <REL.OP> IS ONE OF THE OPERATORS IN TAB6 AND THE RESULT OF
; * THESE OPERATIONS IS 1 IF TRUE AND 0 IF FALSE.
; * <EXPR1>::=(+ OR -)<EXPR2>( + OR -<EXPR2>)(...)
; * WHERE () ARE OPTIONAL AND (...) ARE OPTIONAL REPPEATS.
; * <EXPR2>::=<EXPR3>(<* OR /><EXPR3>)(...)
; * <EXPR3>::=<VARIABLE>
; *      <FUNCTION>
; *      (<EXPR>)
; * <EXPR> IS RECURSIVE SO THAT VARIABLE '@' CAN HAVE AN <EXPR> AS
; * INDEX, FUNCTIONS CAN HAVE AN <EXPR> AS ARGUMENTS, AND
; * <EXPR3> CAN BE AN <EXPR> IN PARENTHESES.
27FA  CD 2842          EXPR:  CALL  EXPR1  ; *** EXPR ***
27FD  E5                PUSH  H      ;SAVE <EXPR1> VALUE
27FE  21 2C23          LXI   H,TAB6-1 ; LOOKUP REL.OP.
2801  C3 25E2          JMP   EXEC   ; GO DO IT.
2804  CD 282D          XPR1:  CALL  XPR8  ; REL.OP. ">="
2807  D8                RC        ; NO, RETURN HL=0
2808  6F                MOV     L,A   ; YES, RETURN HL=1
2809  C9                RET
280A  CD 282D          XPR2:  CALL  XPR8  ; REL.OP. "#"
280D  C8                RZ        ; FALSE, RETURN HL=0
280E  6F                MOV     L,A   ; TRUE, RETURN HL=1
280F  C9                RET
2810  CD 282D          XPR3:  CALL  XPR8  ; REL.OP. ">"
2813  C8                RZ        ; FALSE
2814  D8                RC
2815  6F                MOV     L,A   ; TRUE, RETURN HL=1
2816  C9                RET
2817  CD 282D          XPR4:  CALL  XPR8  ; REL.OP "<="
281A  6F                MOV     L,A   ; SET HL=1
281B  C8                RZ        ; REL. TRUE, RETURN
281C  D8                RC
281D  6C                MOV     L,H   ; ELSE SET HL=0
281E  C9                RET
281F  CD 282D          XPR5:  CALL  XPR8  ; REL.OP. "="
2822  C0                RNZ       ; FALSE, RETURN HL=0
2823  6F                MOV     L,A   ; ELSE SET HL=1
2824  C9                RET
2825  CD 282D          XPR6:  CALL  XPR8  ; REL.OP "<"
2828  D0                RNC       ; FALSE, RETURN HL=0
2829  6F                MOV     L,A   ; ELSE SET HL=1

```



.MAIN. -

```

282A C9          RET
282B E1          XPR7: POP      H      ; NOT REL.OP.
282C C9          RET          ;RETURN HL=<EXPR1>
282D 79          XPR8: MOV      A,C    ; SUBROUTINE FOR ALL
282E E1          POP      H      ; REL.OP.'S
282F C1          POP      B
2830 E5          PUSH     H      ; REVERSE TOP OF STACK
2831 C5          PUSH     B
2832 4F          MOV      C,A
2833 CD 2842     CALL     EXPR1   ; GET 2ND <EXPR1>
2836 EB          XCHG
2837 E3          XTHL
2838 CD 215A     CALL     CKHLDE  ; COMPARE 1ST WITH 2ND
283B D1          POP      D      ; RESTORE TEXT POINTER
283C 21 0000     LXI      H,0    ; SET HL=0, A=1
283F 3E01       MVI      A,1
2841 C9          RET
2842 CF          XPR1: TSTC   '-',XP11   ; NEGATIVE SIGN?[      RST      1
2843 2D          +      .BYTE  '-'
2844 05          +      .BYTE  XP11   -.-1
                +]
2845 21 0000     LXI      H,0    ; YES, FAKE "0-"
2848 1821       JMPR   XP16   ; TREAT LIKE SUBTRACT
284A CF          XPR1: TSTC   '+',XP12   ; POSITIVE SIGN? IGNORE[      RST      1
284B 2B          +      .BYTE  '+'
284C 00          +      .BYTE  XP12   -.-1
                +]
284D CD 2874     XPR2: CALL   EXPR2   ; 1ST <EXPR2>
2850 CF          XPR3: TSTC   '+',XP15   ; ADD?[ RST      1
2851 2B          +      .BYTE  '+'
2852 15          +      .BYTE  XP15   -.-1
                +]
2853 E5          PUSH     H      ; YES, SAVE VALUE
2854 CD 2874     CALL     EXPR2   ; GET 2ND <EXPR2>
2857 EB          XPR4: XCHG
2858 E3          XTHL
2859 7C          MOV      A,H    ; COMPARE SIGN
285A AA          XRA      D
285B 7A          MOV      A,D
285C 19          DAD      D
285D D1          POP      D      ; RESTORE TEXT POINTER
285E FA 2850     JM       XP13   ; 1ST 2ND SIGN DIFFER
2861 AC          XRA      H      ;1ST 2ND SIGN EQUAL
2862 F2 2850     JP       XP13   ; SO IS RESULT
2865 C3 2AE6     JMP      QHOW   ; ELSE WE HAVE OVERFLOWN
2868 CF          XPR5: TSTC   '-',XPR9   ; SUBTRACT?[      RST      1
2869 2D          +      .BYTE  '-'
286A 98          +      .BYTE  XPR9   -.-1
                +]
286B E5          XPR6: PUSH     H      ; YES, SAVE 1ST <EXPR2>
286C CD 2874     CALL     EXPR2   ; GET 2ND <EXPR2>
286F CD 298C     CALL     CHGSGN  ; NEGATE
2872 18E3       JMPR   XP14   ; AND ADD THEM

```

```

2874 CD 28D7   EXPR2: CALL   EXPR3   ; GET 1ST <EXPR3>
2877 CF       XP21: TSTCC  62H,XP24 ; MULTIPLY?{   RST   1
2878 62       +       .BYTE  62H
2879 27       +       .BYTE  XP24   --1
          +]

287A E5       PUSH   H           ; YES, SAVE 1ST
287B CD 28D7   CALL   EXPR3   ; AND GET 2ND <EXPR3>
287E 0600     MVI   B,0       ; CLEAR B FOR SIGN
2880 CD 2989   CALL   CHKSGN  ; CHECK SIGN
2883 E3       XTHL                ; 1ST IN HL
2884 CD 2989   CALL   CHKSGN  ; CHECK SIGN OF 1ST
2887 EB       XCHG
2888 E3       XTHL
2889 7C       MOV   A,H       ; IS HL > 255?
288A B7       ORA   A
288B 2806     JRZ   XP22     ; NO
288D 7A       MOV   A,D       ; YES, HOW ABOUT DE
288E B2       ORA   D
288F EB       XCHG                ; PUT SMALLER IN DE
2890 C2 2AE7   JNZ   AHOW    ; ALSO >, WILL OVERFLOW
2893 7D       XP22: MOV   A,L     ; THIS IS DUMB
2894 21 0000   LXI   H,0       ; CLEAR RESULT
2897 B7       ORA   A           ; ADD AND COUNT
2898 2830     NERDXX: JRZ   XP25
289A 19       XP23: DAD   D
289B DA 2AE7   JC     AHOW    ; OVERFLOW
289E 3D       DCR   A
289F 18F7     JMPR  NERDXX  ; FINISHED
28A1 CF       XP24: TSTCC  63H,XPR9  ; DIVIDE?{   RST   1
28A2 63       +       .BYTE  63H
28A3 5F       +       .BYTE  XPR9   --1
          +]

28A4 E5       PUSH   H           ; YES, SAVE 1ST <EXPR3>
28A5 CD 28D7   CALL   EXPR3   ; AND GET 2ND ONE
28A8 0600     MVI   B,0       ; CLEAR B FOR SIGN
28AA CD 2989   CALL   CHKSGN  ; CHECK SIGN OF 2ND
28AD E3       XTHL                ; GET 1ST IN HL
28AE CD 2989   CALL   CHKSGN  ; CHECK SIGN OF 1ST
28B1 EB       XCHG
28B2 E3       XTHL
28B3 EB       XCHG
28B4 7A       MOV   A,D       ; DIVIDE BY 0?
28B5 B3       ORA   E
28B6 CA 2AE7   JZ     AHOW    ; SAY "HOW?"
28B9 C5       PUSH  B           ; ELSE, SAVE SIGN
28BA CD 2974   CALL  DIVIDE   ; USE SUBROUTINE
28BD D1       POP   D           ; SIGN STUFF TO DE
28BE C5       PUSH  B           ; SAVE DIVIDE RESULT
28BF CB7A     BIT   7,D       ; WAS SIGN SET?
28C1 C4 298C   CNZ   CHGSGN  ; YEP - CHANGE
28C4 22 4E78   SHLD  REMAIN  ; STUFF IT
28C7 E1       POP   H           ; RESULT TO HL
28C8 42       MOV   B,D       ; COPY OVER SIGN STUFF
    
```

.MAIN. -

```

28C9 4B          MOV     C,E
28CA D1          XP25:  POP     D           ; GET TEXT POINTER BACK
28CB 7C          MOV     A,H           ; HL MUST BE +
28CC B7          ORA     A
28CD FA 2AE6     JM      QHOW           ; ELSE IT IS OVERFLOW
28D0 78          MOV     A,B
28D1 B7          ORA     A
28D2 FC 298C     CM      CHGSGN        ; CHANGE SIGN IF NEEDED
28D5 18A0        JMPR   XP21           ; LOOK FOR MORE TERMS
28D7 21 2BFA     EXPR3: LXI   H,TAB3-1       ; FIND FUNCTION IN TAB3
28DA C3 25E2     JMP     EXEC           ; AND GO DO IT
28DD CD 2A49     NOTF:  CALL   TSTV        ; NO, NOT A FUNCTION
28E0 380A        JRC    XP32           ; NOR A VARIABLE
28E2 EB          XCHG
28E3 DF          RST     RSTLDE
28E4 F5          PUSH   PSW
28E5 13          INX     D
28E6 DF          RST     RSTLDE
28E7 EB          XCHG
28E8 67          MOV     H,A
28E9 F1          POP     PSW
28EA 6F          MOV     L,A
28EB C9          RET
28EC CD 2ABF     XP32:  CALL   TSTNUM       ; OR IS IT A NUMBER?
28EF 78          MOV     A,B           ; # OF DIGIT
28F0 B7          ORA     A
28F1 C0          RNZ           ; OK
                ; SINGLE CHAR STRING CONSTANT?
28F2 CF          TSTC   '",' ,PARN           ; HAVE WE GOT QUOTES?[ RST 1
28F3 22          +     .BYTE  '",'
28F4 07          +     .BYTE  PARN      -.-1
                +]
28F5 1A          LDAX   D           ; NAILED RSTLDE
28F6 6F          MOV     L,A           ; FAILED TSTNUM SET H TO ZERO
28F7 13          INX     D
28F8 CF          TSTC   '",' ,XPR0         ; ERROR IF NO TRAILING[ RST 1
28F9 22          +     .BYTE  '",'
28FA 09          +     .BYTE  XPR0      -.-1
                +]
28FB C9          RET
                ; *****
                ; *
28FC CF          PARN:  TSTC   '(',XPR0         ; NO DIGIT, MUST BE [ RST 1
28FD 28          +     .BYTE  '('
28FE 05          +     .BYTE  XPR0      -.-1
                +]
28FF D7          PARNP: RST     RSTEXP        ; "(EXPR)"
2900 CF          TSTC   ')',XPR0[         RST 1
2901 29          +     .BYTE  ')'
2902 01          +     .BYTE  XPR0-.-1
                +]
2903 C9          XPR9:  RET

```

.MAIN. -

```

2904 C3 29C6 XPR0: JMP QWHAT ; ELSE SAY: "WHAT?"
2907 EF RND: RST RSTPAR ; *** RND(EXOR) ***
2908 7C MOV A,H ; EXPR MUST BE +
2909 B7 ORA A
290A FA 2AE6 JM QHOW
290D D5 PUSH D ; SAVE BOTH
290E EB XCHG ; DE = RANGE
290F AF XRA A
2910 FF RANGED[INTP%[.IFE .INTP.,[RST 7]]
2911 76 +.BYTE 118+0]
2912 6F MOV L,A
2913 AF XRA A
2914 FF RANGED[INTP%[.IFE .INTP.,[RST 7]]
2915 76 +.BYTE 118+0]
2916 67 MOV H,A
; HL = RANDOM #
2917 C5 PUSH B
2918 7A MOV A,D
2919 B3 ORA E
291A C4 2974 CNZ DIVIDE ; RND(N)=MOD(M,N)+1
291D C1 POP B
291E D1 POP D
291F 23 INX H
2920 C9 RET
2921 EF ABS: RST RSTPAR ; *** ABS(EXPR) ***
2922 1B DCX D
2923 CD 2989 CALL CHKSGN ; CHECK SIGN
2926 13 INX D
2927 C9 RET
2928 2A 4E20 SIZE: LHLD TXTUNF ; *** SIZE ***
292B D5 PUSH D ; GET THE NUMBER OF
292C EB XCHG ; FREE BYTES BETWEEN 'TXTUNF'
292D 21 A70C LXI H,DFTLMT ; AND 'TXTLMT'
2930 A7 ANA A
2931 ED52 DSBC D
2933 D1 POP D
2934 C9 RET
; FUNCTION TO SENSE DIAL VALUE
2935 3E1B GETPOT: MVI A,1BH
2937 CD 2967 CALL CHKRNG ; GET DATA
293A 2F CMA
293B D680 SUI 80H
293D 6F MOV L,A
; FALL INTO ...
; SIGN EXTEND SUBROUTINE
293E 2600 SGNEXT: MVI H,0
2940 7D MOV A,L
2941 A7 ANA A
2942 F0 RP
2943 25 DCR H
2944 C9 RET
; FUNCTION TO SENSE STATE OF TRIGGER
2945 CD 2965 GETTRG: CALL CHKR1

```

.MAIN. -

```

2948 E610          ANI    10H
294A C8           RZ
294B 2C          INR    L
294C C9          RET
; FUNCTIONS TO RETURN JOYSTICK VALUE
; THESE FUNCTIONS RETURN EITHER +1, ,0 OR -1, DEPENDING
; ON JOYSTICK STATE
294D CD 2965     GETJX: CALL   CHKR1  ; PARM IN RANGE?
2950 0F          RRC
2951 0F          RRC
2952 0F          RRC
2953 380E        JRC     GETJY3
2955 0F          RRC
2956 3807        JRC     GETJY1
2958 C9          RET
; ENTRY FOR Y JOYSTICK VALUE
2959 CD 2965     GETJY: CALL   CHKR1
295C 0F          RRC
295D 3002        JRNC   GETJY2
295F 23          GETJY1: INX   H
2960 C9          RET
2961 0F          GETJY2: RRC
2962 D0          RNC
2963 2B          GETJY3: DCX   H
2964 C9          RET
; SUBROUTINE TO GET PARAMETER BETWEEN 1 AND 4
2965 3E0F        CHKR1: MVI   A,0FH
2967 C5          CHKRNG: PUSH  B
2968 F5          PUSH  PSW
2969 EF          RST   RSTPAR
296A F1          POP   PSW
296B 85          ADD   L
296C 4F          MOV   C,A
296D ED78        INP   A
296F C1          POP   B
2970 21 0000     LXI   H,0
2973 C9          RET
; *
; * *****
; * *** DIVIDE *** SUBDE *** CHKSGN *** CHGSGN *** & CKHLDE ***
; *
; * 'DIVIDE' DIVIDES HL BY DE, RESULT IN BC, REMAINDER IN HL
; *
; * 'SUBDE' SUBTRACTS DE FROM HL.
; *
; * 'CHKSGN' CHECKS SIGN OF HL. IF +,NO CHANGE. IF -,CHANGE SIGN AND
; * FLIP SIGN OF B.
; *
; * 'CHKSGN' CHNGES SIGN OF HL AND B UNCONDITIONALLY.
; *
; * 'CKHLDE' CHECKS SIGN OF HL AND DE. IF DIFFERENT, HL AND DE ARE
; * INTERCHANGED. IF SAME SIGN, NOT INTERCHANGED. EITHER CASE, HL DE
; * ARE THEN COMPARED TO SET THE FLAGS.

```

.MAIN. -

```

; *
2974 E5 DIVIDE: PUSH H ;*** DIVIDE ***
2975 6C MOV L,H ;DIVIDE H BY DE
2976 2600 MVI H,0
2978 CD 297F CALL DV1
297B 41 MOV B,C ; ;SAVE RESULT IN B
297C 7D MOV A,L ; (REMAINDER + L)/DE
297D E1 POP H
297E 67 MOV H,A
297F 0EFF DV1: MVI C,-1 ; RESULT IN C
2981 0C DV2: INR C ; DUMB ROUTINE
2982 A7 ANA A
2983 ED52 DSBC D
2985 30FA JRNC DV2
2987 19 DAD D
2988 C9 RET
2989 7C CHKSGN: MOV A,H ; *** CHKSNG ***
298A B7 ORA A ; CHECK SIGN OF HL
298B F0 RP ; IF -, CHANGE SIGN
298C 7C CHGSGN: MOV A,H ; *** CHGSGN ***
298D B5 ORA L
298E C8 RZ
298F 7C MOV A,H
2990 F5 PUSH PSW
2991 2F CMA ; CHANGE SIGN OF HL
2992 67 MOV H,A
2993 7D MOV A,L
2994 2F CMA
2995 6F MOV L,A
2996 23 INX H
2997 F1 POP PSW
; XRA H
; JP QHOW
2998 78 MOV A,B ; AND ALSO FLIP B
2999 EE80 XRI 80H
299B 47 MOV B,A
299C C9 RET
;CKHLDE: MOV A,H ; *** CKHLDE ***
; XRA D ; SAME SIGN?
; JP CK1 ; YES, COMPARE
; XCHG ; NO, XCH AND COMPARE
;CK1: CALL COMP
; RET
;COMP: MOV A,H ; *** COMP ***
; CMP D ; COMPARE HL WITH DE
; RNZ ; RETURN CORRECT C AND
; MOV A,L ; ZFLAGS
; CMP E ; BUT OLD A IS LOST
; RET
; * *****
; *
; * *** SETVAL *** FIN *** ENDCHK *** & ERROR (& FRIENDS) ***
; *

```

```

; * "SETVAL" EXPECTS A VARIABLE, FOLLOWED BY AN EQUAL SIGN AND THEN AN
; * EXPR. IT EVALUATES THE EXPR. AND SETS THE VARIABLE TO THAT VALUE.
; *
; * "FIN" CHECKS THE END OF A COMMAND. IF IT ENDED WITH ";", EXECUTION
; * CONTINUES. IF IT ENDED WITH A CR, IT FINDS THE NEXT LINE AND
; * CONTINUES FROM THERE.
; *
; * "ENDCHK" CHECKS IF A COMMAND IS ENDED WITH CR. THIS IS REQUIRED IN
; * CERTAIN COMMANDS. (GOTO, RETURN, AND STOP ETC.)
; *
; * "ERROR" PRINTS THE STRING POINTED BY DE (AND ENDS WITH CR). IT THEN
; * PRINTS THE LINE POINTED BY 'CURRNT' WITH A "?" INSERTED AT WHERE THE
; * OLD TEXT POINTER (SHOULD BE ON TOP OF THE STACK) POINTS TO.
; * EXECUTION OF TB IS STOPPED AND TBI IS RESTARTED. HOWEVER, IF
; * 'CURRNT'=> ZERO (INDICATING A DIRECT COMMAND), THE DIRECT COMMAND
; * IS NOT PRINTED. AND IF 'CURRNT'=>NEGATIVE # (INDICATING 'INPUT'
; * COMMAND, THE INPUT LINE IS NOT PRINTED AND EXECUTION IS NOT
; * TERMINATED BUT CONTINUED AT 'INPERR'.
; *
; * RELATED TO 'ERROR' ARE THE FOLLOWING: 'QWHAT' SAVES TEXT POINTED IN
; * STACK AND GET MESSAGE "WHAT?" JUST GETS MESSAGE "WHAT?" AND
; * JUMP TO 'ERROR'. 'QSORRY' AND 'ASORRY' DO SAME KIND OF THING.
; * 'QHOW' AND 'AHOW' IN THE ZERO PAGE SECTION ALSO DO THIS.
; *

```

```

299D  CD 22F3  SETVAL: CALL  TSTVFF ; *** SETVAL ***
29A0  CF          TSTC  '=' , QWHAT      ; 2WHAT?" NO VARIABLE[ RST    1
29A1  3D          +      .BYTE  '='
29A2  23          +      .BYTE  QWHAT  --1
+ ]

29A3  E5          SETV1: PUSH  H          ; SAVE ADDRESS OF VAR.
29A4  D7          RST   RSTEXP ; EVALUATE EXPR.
29A5  EB          XCHG
29A6  E3          XTHL
29A7  CD 2FDC    CALL   STDEHL
29AA  D1          POP   D
29AB  C9          RET
29AC  CD 29B1    FINISH: CALL  FIN      ; CHECK END OF COMMAND
29AF  1815       JMPR   QWHAT  ; PRINT "WHAT?" IF WRONG
29B1  CF          FIN:  TSTCC  59, FI1  ; *** FIN ***[ RST    1
29B2  3B          +      .BYTE  59
29B3  04          +      .BYTE  FI1   --1
+ ]

29B4  F1          POP   PSW      ; ";", PURGE RET ADDR.
29B5  C3 262C    JMP   RUNSML ; CONTINUE SAME LINE
29B8  CF          FI1:  TSTCC  CR, FI2  ; NOT ";", IS IT CR?[ RST    1
29B9  0D          +      .BYTE  CR
29BA  61          +      .BYTE  FI2   --1
+ ]

29BB  F1          POP   PSW      ; PURGE RETURN ADDRESS
29BC  3A 4EAA    IMCHK: LDA   CURRNT+1
29BF  07          RLC
29C0  D2 2544    JNC   RSTART
29C3  C3 2605    JMP   RUNNXL ; RUN NEXT LINE

```

.MAIN. -

```

29C6 D5 QWHAT: PUSH D ; *** QWHAT ***
29C7 11 201A AWHAT: LXI D,WHAT ; *** AWHAT ***
29CA CD 2D06 ERROR: CALL CRLF ; *** ERROR ***
29CD CD 2B4F CALL PRTSTG ; PRINT ERROR MESSAGE
29D0 2A 4EA9 LHLD CURRNT ; GET CURRENT LINE #
29D3 E5 PUSH H
29D4 EB XCHG ; CHECK THE VALUE
29D5 DF RST RSTLDE
29D6 67 MOV H,A
29D7 13 INX D
29D8 DF RST RSTLDE
29D9 B4 ORA H
29DA EB XCHG
29DB D1 POP D
29DC CA 2544 JZ TELL ; IF ZERO, JUST RESTART
29DF EB XCHG ; IF NEGATIVE
29E0 DF RST RSTLDE
29E1 EB XCHG
29E2 B7 ORA A
29E3 FA 279E JM INPERR ; REDO INPUT
29E6 CD 2BBC CALL PRTLN ; ELSE PRINT THE LINE
29E9 E1 POP H ; HL=ERROR ADDR
29EA 01 4EBA LXI B,XQTBUF
29ED A7 ANA A
29EE ED42 DSBC B
29F0 19 DAD D
29F1 45 MOV B,L
29F2 CD 2B75 CALL PRTCHS
29F5 CD 20A4 CALL OUTCHQ
29F8 CD 2B4F CALL PRTSTG ; LINE
29FB C3 2544 JMP TELL ; THEN RESTART
29FE D5 QSORRY: PUSH D ; *** QSORRY ***
29FF 11 216B ASORRY: LXI D,SORRY ; *** ASORRY ***
2A02 18C6 JMPR ERROR
; * *****
; *
; * *** FNDLN (& FRIENDS) ***
; *
; * 'FNDLN' FINDS A LINE WITH A GIVEN LINE # (IN HL) IN THE TEXT SAVE
; * AREA. DE IS USED AS THE TEXT POINTER. IF THE LINE IS FOUND, DE
; * WILL POINT TO THE BEGINNING OF THAT LINE (I.E, THE LOW BYTE OF THE
; * LINE #), AND FLAGS ARE NC & Z. IF THAT LINE IS NOT THERE AND A LINE
; * WITH A HIGHER LINE # IS FOUND, DE POINTS TO THERE AND FLAGS ARE NC &
; * NZ. IF WE REACHED THE END OF TEXT SAVE AREA AND CANNOT FIND THE
; * LINE, FLAGS ARE C & NZ. 'FNDLN' WILL INITIALIZE DE TO THE BEGINNING
; * OF THE TEXT SAVE AREA TO START THE SEARCH. SOME OTHER ENTRIES OF
; * THIS ROUTINE WILL NOT INITIALIZE DE AND DO THE SEARCH. 'FNDLP'
; * WILL START WITH DE AND SEARCH FOR THE LINE #. 'FNDNXT' WILL BUMP DE
; * BY 2, FIND A CR AND THEN START SEARCH. 'FNDSKP' USES DE TO FIND A
; * CR, AND THEN STARTS SEARCH.
; *
2A04 7C FNDLN: MOV A,H ;*** FNDLN ***
2A05 B7 ORA A ; CHECK SIGN OF HL

```



.MAIN. -

```

2A06 FA 2AE6 JM QHOW ; IT CANNOT BE -
2A09 11 A000 LXI D,TEXT ; INIT. TEXT POINTER
2A0C 13 FNDLP: INX D ; IS IT EOT MARK?
2A0D DF RST RSTLDE
2A0E 4F MOV C,A
2A0F 1B DCX D
2A10 87 ADD A
2A11 D8 RC
2A12 DF RST RSTLDE ; C,NZ PASSED END
2A13 95 SUB L ; WE DID NOT, GET BYTE 1
2A14 47 MOV B,A ; IS THIS THE LINE?
2A15 13 INX D ; COMPARE LOW ORDER
2A16 79 MOV A,C ; GET BYTE 2
2A17 9C SBB H ; COMPARE HIGH ORDER
2A18 3804 JRC FL1 ; NO, NOT THERE YET
2A1A 1B DCX D ; ELSE WE EITHER FOUND
2A1B B0 ORA B ; IT, OR IT IS NOT THERE
2A1C C9 FI2: RET ; NC,Z: FOUND; NC,NZ: NO
2A1D 13 FNDNXT: INX D ; FIND NEXT LINE
2A1E 13 FL1: INX D ; JUST PASSED BYTE
; FASTER FNDSKP:
2A1F EB FNDSKP: XCHG
2A20 29 DAD H ; CONVERT TO 'NORMAL'
2A21 7E ..FS1: MOV A,M ; GET NEXT BYTE
2A22 07 RLC ; COMBINE WITH FOLLOWING FELLA
2A23 23 INX H
2A24 AE XRA M ; TO MAKE THE REAL DATA
2A25 E6AA ANI 0AAH
2A27 AE XRA M
2A28 23 INX H
2A29 FE0D CPI CR ; HIT A CR YET?
2A2B 20F4 JRNZ ..FS1 ; NO SIR EEE
2A2D 37 STC ; WE GOT IT
2A2E CB1C RARR H ; NORMALIZE OUR POINTER
2A30 CB1D RARR L
2A32 EB XCHG
2A33 18D7 JMPR FNDLP ; REENTER FIND LOOP
; SUBROUTINE TO GRAB AND VERIFY SUBSCRIPT
2A35 FDE1 GETSUB: POP Y ; STICK RETURN INTO IY
2A37 13 INX D ; SKIP DA NAME
2A38 EF RST RSTPAR ; GET THE PARM
2A39 29 DAD H ; CONVERTTETH TO BYTES
2A3A DA 2AE6 JC QHOW ; REJECT ABSURD VALUES
2A3D D5 PUSH D ; SAVE SCAN PTR
2A3E EB XCHG
2A3F CD 2928 CALL SIZE ; CHECK FOR VALID SUBSCRIPT
2A42 CD 2160 CALL COMP
2A45 38B8 JRC ASORRY ; APOLOGIZE FOR RANGE ERR
2A47 FDE9 PCIY ; GO HOME
2A49 E7 TSTV: RST RSTIGN ; *** TSTV ***
2A4A FE25 CPI '%' ; PEEK-POKE?
2A4C 281F JRZ TSTV0
2A4E FE2A CPI '*' ; BACKWARDS ARRAY?

```

.MAIN. -

```

2A50 2810          JRZ    STARR  ; YEP - JUMP TO IT
2A52 D640          SUI    '@'    ; TEST VARIABLES
2A54 D8            RC      C:     ; C: NOT A VARIABLE
2A55 201A          JRNZ   TV1    ; NOT "@" ARRAY
2A57 CD 2A35       CALL   GETSUB
2A5A 2A 4E20       LHLD   TXTUNF
2A5D 2B            DCX    H
2A5E 2B            DCX    H
2A5F 19            DAD    D
2A60 D1            POP    D
2A61 C9            RET
; PROCESS THE BACKWARDS ARRAY
2A62 CD 2A35       STARR:  CALL   GETSUB
2A65 21 A70C       LXI    H,DFTLMT ; SUBTRACT INDEX FROM END
2A68 ED52          DSBC   D
2A6A D1            POP    D
2A6B AF            XRA    A      ; NO CY SHIT
2A6C C9            RET
; %(ADDR) PEEK-POKE CALL
2A6D 13            TSTV0: INX    D
2A6E EF            RST    RSTPAR ; GET ADDR
2A6F AF            XRA    A      ; CLEAR CY
2A70 C9            RET    ; AND GO BACK
2A71 FE1B          TV1:   CPI    27  ; NOT @, IS IT A TO Z
2A73 3F            CMC      ; IF NOT RETURN C FLAG
2A74 D8            RC
2A75 13            INX    D      ; IF A THROUGH Z
; IS SECOND CHARACTER ALSO ALPHA?
2A76 6F            MOV    L,A    ; SAVE FIRST ONE
2A77 1A            LDAX   D      ; ZAPPED RSTLDE
2A78 FE41          CPI    'A'
2A7A 3826          JRC    DEVV4  ; IF NOT IN RANGE A-Z
2A7C FE5B          CPI    'Z'+1
2A7E 3022          JRNC   DEVV4  ; THEN SEARCH
2A80 C5            PUSH   B
2A81 D5            PUSH   D
2A82 67            MOV    H,A    ; SECOND CHAR TO H
2A83 0613          MVI    B,PARNUM ; B - ITERATION CTR
2A85 11 21D7       LXI    D,DEVLST ; DE - SEARCH TABLE
2A88 1A            DEVV1: LDAX   D      ; GET FIRST ENTRY
2A89 13            INX    D
2A8A BD            CMP    L
2A8B 1A            LDAX   D
2A8C 13            INX    D
2A8D 200B          JRNZ   DEVV2
2A8F BC            CMP    H
2A90 2008          JRNZ   DEVV2
; MATCH FOUND - FIGURE OUT LOOKUP INDEX
2A92 78            MOV    A,B
2A93 C61A          ADI    26
2A95 6F            MOV    L,A
2A96 D1            POP    D
2A97 13            INX    D      ; BUMP CHAR PTR

```

.MAIN. -

```

2A98 1807          JMPR    DEVV3
          ; MISMATCH - LOOP BACK IF POSS
2A9A 10EC          DEVV2: DJNZ    DEVV1
          ; NOT POSSIBLE - RETURN NOT A VAR
2A9C D1            POP     D
2A9D C1            POP     B
2A9E 1B            DCX     D          ; BACKUP TO CHAR START
2A9F 37            STC     ; SET CARRY
2AA0 C9            RET
2AA1 C1            DEVV3: POP     B
2AA2 7D            DEVV4: MOV    A,L
2AA3 21 4E20       LXI     H,VARBGN-2
2AA6 07            RLC
2AA7 85            ADD     L
2AA8 6F            MOV    L,A
2AA9 3E00          MVI    A,0
2AAB 8C            ADC     H
2AAC 67            MOV    H,A
2AAD C9            RET
          ; *
          ; * *****
          ; *
          ; * *** TSTCH *** TSTNUM ***
          ; *
          ; * TSTCH IS USED TO TEST THE NEXT NON-BLANK CHARACTER IN THE TEXT
          ; * (POINTED BY DE) AGAINST THE CHARACTER THAT FOLLOWS THE CALL. IF
          ; * THEY DO NOT MATCH, N BYTES OF CODE WILL BE SKIPPED OVER, WHERE N IS
          ; * BETWEEN 0 & 255 AND IS STORED IN THE SECOND BYTE FOLLOWING THE CALL
          ; *
          ; * TSTNUM IS USED TO CHECK WHETHER THE TEXT (POINTED BY DE) IS A
          ; * NUMBER. IF A NUMBER IS FOUND, B WILL BE NON-ZERO AND HL WILL
          ; * CONTAIN THE VALUE (IN BINARY) OF THE NUMBER, ELSE B AND HL ARE 0.
          ; *
2AAE E3            TSTCH: XTHL          ; *** TSTCH ***
2AAF E7            RST     RSTIGN ; IGNORE LEADING BLANKS
2AB0 BE            CMP     M          ; AND TEST THE CHARACTER
2AB1 23            INX     H          ; COMPARE THE BYTE THAT
2AB2 2807          JRZ     TC1       ; FOLLOWS THE CALL INTS.
2AB4 C5            PUSH    B          ; WITH THE TEXT (DE->)
2AB5 4E            MOV    C,M       ; IF NOT =, ADD THE 2ND
2AB6 0600          MVI    B,0       ; BYTE THAT FOLLOWS THE
2AB8 09            DAD     B          ; CALL TO THE OLD PC
2AB9 C1            POP     B          ; I.E., DO A RELATIVE
2ABA 1B            DCX     D          ; JUMP IF NOT =
2ABB 13            TC1:  INX     D          ; IF =, SKIP THOSE BYTES
2ABC 23            INX     H          ; AND CONTINUE
2ABD E3            XTHL
2ABE C9            RET
2ABF 21 0000       TSTNUM: LXI    H,0       ; *** TSTNUM ***
2AC2 44            MOV    B,H       ; TEST IF THE TEXT IS
2AC3 E7            RST     RSTIGN ; A NUMBER
2AC4 FE30          TN1:  CPI    '0'       ; IF NOT, RETURN 0 IN
2AC6 D8            RC          ; B AND HL

```

.MAIN. -

```

2AC7 FE3A      CPI      3AH      ; IF NUMBERS, CONVERT
2AC9 D0        RNC          ; TO BINARY IN HL AND
2ACA 3EF0     MVI      A,0FOH   ; SET B TO # OF DIGITS
2ACC A4        ANA      H      ; IF H>255, THERE IS NO
2ACD 2017     JRNZ     QHOW   ; ROOM FOR NEXT DIGIT
2ACF 04       INR      B      ; B COUNTS # OF DIGITS
2AD0 C5       PUSH     B
2AD1 44       MOV      B,H    ; HL=10*HL+(NEW DIGIT)
2AD2 4D       MOV      C,L
2AD3 29       DAD      H      ; WHERE 10* IS DONE BY
2AD4 29       DAD      H      ; SHIFT AND ADD
2AD5 09       DAD      B
2AD6 29       DAD      H
2AD7 1A       LDAX    D      ; AND (DIGIT) IS FROM
2AD8 13       INX      D      ; STRIPPING THE ASCII
2AD9 E60F     ANI      00FH   ; CODE
2ADB 85       ADD      L
2ADC 6F       MOV      L,A
2ADD 3E00     MVI      A,0
2ADF 8C       ADC      H
2AE0 67       MOV      H,A
2AE1 C1       POP      B
2AE2 1A       LDAX    D      ; DO THIS DIGIT AFTER
2AE3 F2 2AC4  JP      TN1    ; DIGIT. S SAYS OVERFLOW
2AE6 D5       QHOW:    PUSH   D      ; *** ERROR: "HOW?" ***
2AE7 11 2166  AHOW:    LXI      D,HOW
2AEA C3 29CA  JMP      ERROR
; MVUP, MVDOWN, POPA, AND PUSHA
; 'MVUP' MOVES A BLOCK UP FROM WHERE DE-> TO WHERE BC->
; UNTIL DE=HL
;
; 'MVDOWN' MOVES A BLOCK DOWN FROM WHERE DE-> TO WHERE HL->
; UNTIL DE=BC
;
; 'POPA' RESTORES THE 'FOR' LOOP VARIABLE SAVE AREA FROM THE STACK
; 'PUSHA' STACKS THE 'FOR' LOOP VARIABLE SAVE AREA INTO THE STACK
2AED CD 2160  MVUP:    CALL     COMP   ; *** MVUP ***
2AF0 C8       RZ          ; DE=HL, RETURN
2AF1 DF       RST      RSTLDE ; GET ONE BYTE
2AF2 E5       PUSH     H      ; SHOVEL REGS
2AF3 60       MOV      H,B
2AF4 69       MOV      L,C
2AF5 CD 2FE2  CALL     STHL   ; MOVE IT
2AF8 E1       POP      H
2AF9 13       INX      D      ; INCREASE BOTH POINTERS
2AFA 03       INX      B
2AFB 18F0     JMPR    MVUP   ; UNTIL DONE
2AFD 78       MVDOWN:  MOV     A,B   ; *** MVDOWN ***
2AFE 92       SUB      D      ; TEST IF DE = BC
2AFF 2003     JRNZ     MD1    ; NO, GO MOVE
2B01 79       MOV      A,C   ; MAYBE, OTHER BYTE
2B02 93       SUB      E
2B03 C8       RZ          ; YES, RETURN

```

.MAIN. -

```

2B04 1B MD1: DCX D ; ELSE MOVE A BYTE
2B05 2B DCX H ; BUT FIRST DECREASE
2B06 DF RST RSTLDE ; BOTH PTRS AND THEN
2B07 CD 2FE2 CALL STHL ; DO IT
2B0A 18F1 JMPR MVDOWN ; LOOP BACK
2B0C C1 POPA: POP B ; BC = RETURN ADDR.
2B0D E1 POP H ; RESTORE LOPVAR, BUT
2B0E 22 4EAF SHLD LOPVAR ; =0 MEANS NO MORE
2B11 7C MOV A,H
2B12 B5 ORA L
2B13 2810 JRZ PP1 ; YEP, GO RETURN
2B15 E1 POP H ; NO, RESTORE OTHERS
2B16 22 4EB1 SHLD LOPINC
2B19 E1 POP H
2B1A 22 4EB3 SHLD LOPLMT
2B1D E1 POP H
2B1E 22 4EB5 SHLD LOPLN
2B21 E1 POP H
2B22 22 4EB7 SHLD LOPPT
2B25 C5 PP1: PUSH B ; BC = RETURN ADDR.
2B26 C9 RET
2B27 21 B096 PUSHA: LXI H,-STKLMT ; *** PUSHA ***
2B2A C1 POP B ; BC = RETURN ADDR.
2B2B 39 DAD SP ; IS STACK NEAR THE TOP?
2B2C D2 29FE JNC QSORRY ; YES - SORRY FOR THAT
2B2F 2A 4EAF LHLD LOPVAR ; ELSE SAVE LOOP VAR.S
2B32 7C MOV A,H ; BUT IF LOPVAR IS 0
2B33 B5 ORA L ; THAT WILL BE ALL
2B34 2813 JRZ PU1
2B36 2A 4EB7 LHLD LOPPT ; ELSE, MORE TO SAVE
2B39 E5 PUSH H
2B3A 2A 4EB5 LHLD LOPLN
2B3D E5 PUSH H
2B3E 2A 4EB3 LHLD LOPLMT
2B41 E5 PUSH H
2B42 2A 4EB1 LHLD LOPINC
2B45 E5 PUSH H
2B46 2A 4EAF LHLD LOPVAR
2B49 E5 PU1: PUSH H
2B4A C5 PUSH B ; BC = RETURN ADDR.
2B4B C9 RET
; PRTSTG, QTSTG, PRTNUM, PRTLN
; 'PRTSTG' PRINTS A STRING POINTED AT BY DE. IT STOPS
; PRINTING AND RETURNS TO CALLER WHEN EITHER A CR IS
; PRINTED OR WHEN THE NEXT BYTE IS ZERO. REG. A AND B
; ARE CHANGED. REG. DE POINTS TO WHAT FOLLOWS THE CR
; OR TO THE ZERO
; 'QTSTG' LOOKS FOR SINGLE QUOTE, OR DOUBLE QUOTE. IF
; EITHER IT PRINTS THE STRING UNTIL MATCHING UNQUOTE
; AND RETURNS 2 BYTES LATE.
; 'PRTNUM' PRINTS THE NUMBER HL. LEADING BLANKS ARE ADDED IF
; NEEDED TO PAD THE NUMBER OF SPACES TO THE NUMBER IN C.
; HOWEVER, IF THE NUMBER OF DIGITS IS LARGER THAN THE # IN C,

```

.MAIN. -

```

; ALL DIGITS ARE PRINTED ANYWAY. NEGATIVE SIGN IS ALSO PRINTED
; AND COUNTED IN. POSITIVE SIGN IS NOT.
; 'PRTLN' FINDS A SAVED LINE, PRINTS THE LINE # AND A SPACE.
2B4C   CD 2BBC   PRTLNS: CALL  PRTLN
2B4F   97       PRTSTG: SUB  A      ; *** PRTSTG ***
2B50   47       PS1:    MOV   B,A
2B51   DF       PS2:    RST   RSTLDE ; GET A CHARACTER
2B52   13       INX   D      ; BUMP POINTER
2B53   B8       CMP   B      ; SAME AS OLD A
2B54   C8       RZ     ; YES, RETURN
2B55   CD 4E9B   CALL  OUTCH  ; ELSE PRINT IT
2B58   FE0D     CPI   CR      ; WAS IT A CR?
2B5A   20F5     JRNZ  PS2     ; NO - NEXT
2B5C   C9       RET     ; YES-RETURN
2B5D   DF       QTSTG: RST   RSTLDE
2B5E   13       INX   D      ; BUMP PAST
2B5F   FE22     CPI   '"'
2B61   2806     JRZ   QT1     ; IF SINGLE QUOTE-PRINTIT
2B63   FE27     CPI   27H    ; OR IF DOUBLE
2B65   2802     JRZ   QT1     ; LIKEWISE
2B67   1B       DCX   D
2B68   C9       RET
2B69   CD 2B50   QT1:    CALL  PS1     ; PRINT UNTIL ANOTHER
2B6C   FE0D     QT2:    CPI   CR      ; WAS LAST ONE A CR?
2B6E   E1       POP   H      ; RETURN ADDRESS
2B6F   CA 29BC   JZ     IMCHEK  ; WAS CR, END OF THIS
2B72   23       INX   H      ; SKIP 2 BYTES, THEN RET
2B73   23       INX   H
2B74   E9       PCHL
2B75   7B       PRTCHS: MOV  A,E
2B76   B8       CMP   B
2B77   C8       RZ
2B78   DF       RST   RSTLDE
2B79   CD 4E9B   CALL  OUTCH
2B7C   13       INX   D
2B7D   18F6     JMPR  PRTCHS
2B7F           PRTNUM  ==    .      ; *** PRTNUM ***
2B7F   0600     PN3:    MVI  B,0      ; B=SIGN
2B81   CD 2989   CALL  CHKSGN  ; CHECK SIGN
2B84   F2 2B8A   JP    PN4     ; NO SIGN
2B87   062D     MVI  B,'-'    ; B=SIGN
2B89   0D       DCR   C      ; '-' TAKES SPACE
2B8A   D5       PN4:    PUSH  D
2B8B   11 000A   LXI  D,10
2B8E   D5       PUSH  D
2B8F   0D       DCR   C
2B90   C5       PUSH  B
2B91   CD 2974   PN5:    CALL  DIVIDE  ; DIV HL BY 10
2B94   78       MOV   A,B      ; RESULT 0
2B95   B1       ORA   C
2B96   2807     JRZ   PN6     ; YES, WE GOT ALL
2B98   E3       XTHL  ; NO SAVE REMAINDER
2B99   2D       DCR   L      ; AND COUNT SPACE

```

.MAIN. -

```

2B9A E5          PUSH   H          ; HL IS OLD BC
2B9B 60          MOV    H,B          ; MOVE RESULT TO BC
2B9C 69          MOV    L,C
2B9D 18F2        JMPR   PN5          ; AND DIV BY 10
2B9F C1          PN6:   POP    B          ; WE GOT ALL DIGITS IN
2BA0 0D          PN7:   DCR    C          ; THE STACK
2BA1 CB79        BIT    7,C          ; IF SPACE COUNT NEG
2BA3 2005        JRNZ  PN8          ; NO LEADING BLANKS
2BA5 CD 2BC7    CALL  SPOUTCH      ; SPACE OUTCH
2BA8 18F6        JMPR   PN7          ; MORE?
2BAA 78          PN8:   MOV    A,B          ; PRINT SIGN
2BAB B7          ORA    A
2BAC C4 4E9B     CNZ    OUTCH      ; MAYBE - OR NULL
2BAF 5D          MOV    E,L          ; LAST REMAINDER IN E
2BB0 7B          PN9:   MOV    A,E          ; CHECK DIGIT IN E
2BB1 FE0A       CPI    10         ; 10 IS FLAG FOR NO MORE
2BB3 D1          POP    D
2BB4 C8          RZ          ; IF SO RETURN
2BB5 C630       ADI    '0'
2BB7 CD 4E9B     CALL  OUTCH      ; AND PRINT THE DIGIT
2BBA 18F4        JMPR   PN9          ; GO BACK FOR MORE
2BBC DF          PRTLN:  RST    RSTLDE     ; *** PRTLN ***
2BBD 6F          MOV    L,A          ; LOW ORDER LINE #
2BBE 13          INX    D
2BBF DF          RST    RSTLDE     ; HIGH ORDER
2BC0 67          MOV    H,A
2BC1 13          INX    D
2BC2 0E02       MVI    C,2         ; PRINT 2 DIGIT LINE #
2BC4 CD 2B7F    CALL  PRTRNUM
2BC7 3E20       SPOUTCH: MVI   A,' '      ; FOLLOWED BY BLANK
2BC9 C3 4E9B     JMP    OUTCH
2BCC 5456       TAB2:   ITEM   'TV',PUTCD      ; DIRECT-STATEMENT[ .ASCII 'TV'
2BCE A4          +     DEFF  PUTCD [ .BYTE (PUTCD >8)!80H
2BCF E5          +     .BYTE  PUTCD  &0FFH
+
2BD0 4D55       ITEM   'MU',PUTMU[ .ASCII 'MU'
2BD2 A4          +     DEFF  PUTMU[ .BYTE (PUTMU>8)!80H
2BD3 DC          +     .BYTE  PUTMU&0FFH
+
2BD4 26         ITEM   '&',PUTIO[ .ASCII '&'
2BD5 A4          +     DEFF  PUTIO[ .BYTE (PUTIO>8)!80H
2BD6 C5          +     .BYTE  PUTIO&0FFH
+
2BD7 43414C4C   ITEM   'CALL',DOCALL [ .ASCII 'CALL'
2BDB A4          +     DEFF  DOCALL [ .BYTE (DOCALL >8)!80H
2BDC F1          +     .BYTE  DOCALL  &0FFH
+
2BDD 2E         ITEM   '.',REM[ .ASCII '.'
2BDE A6          +     DEFF  REM[ .BYTE (REM>8)!80H
2BDF 05          +     .BYTE  REM&0FFH
+
2BE0 3A         .BYTE  ':'
2BE1 74         TOKEN  74H,TOUTPU[ .BYTE 74H

```

.MAIN. -

```

2BE2  A0      +      DEFF TOUTPU[ .BYTE (TOUTPU>8)!80H
2BE3  2E      +      .BYTE TOUTPU&0FFH
      ]
2BE4  3A      .BYTE ':'
2BE5  73      TOKEN 73H,TINPUT[ .BYTE 73H
2BE6  A0      +      DEFF TINPUT[ .BYTE (TINPUT>8)!80H
2BE7  77      +      .BYTE TINPUT&0FFH
      ]
2BE8  3A      .BYTE ':'
2BE9  6A      TOKEN 6AH,TLOAD[ .BYTE 6AH
2BEA  A0      +      DEFF TLOAD[ .BYTE (TLOAD>8)!80H
2BEB  86      +      .BYTE TLOAD&0FFH
      ]
2BEC  3A      .BYTE ':'
2BED  68      TOKEN 68H,TVLIST[ .BYTE 68H
2BEE  A0      +      DEFF TVLIST[ .BYTE (TVLIST>8)!80H
2BEF  71      +      .BYTE TVLIST&0FFH
      ]
2BF0  60      TOKEN 60H,SILENCE[ .BYTE 60H
2BF1  A2      +      DEFF SILENCE[ .BYTE (SILENCE>8)!80H
2BF2  8E      +      .BYTE SILENCE&0FFH
      ]
2BF3  53544F50 ITEM 'STOP',STOP[ .ASCII 'STOP'
2BF7  A5      +      DEFF STOP[ .BYTE (STOP>8)!80H
2BF8  44      +      .BYTE STOP&0FFH
      ]
2BF9  A9      DEFF FINISH[ .BYTE (FINISH>8)!80H
2BFA  AC      +      .BYTE FINISH&0FFH
      ]
2BFB  76      TAB3: TOKEN 76H,RND ; FUNCTIONS[ .BYTE 76H
2BFC  A9      +      DEFF RND [ .BYTE (RND >8)!80H
2BFD  07      +      .BYTE RND &0FFH
      ]
2BFE  4B4E      ITEM 'KN',GETPOT[ .ASCII 'KN'
2C00  A9      +      DEFF GETPOT[ .BYTE (GETPOT>8)!80H
2C01  35      +      .BYTE GETPOT&0FFH
      ]
2C02  5452      ITEM 'TR',GETTRG[ .ASCII 'TR'
2C04  A9      +      DEFF GETTRG[ .BYTE (GETTRG>8)!80H
2C05  45      +      .BYTE GETTRG&0FFH
      ]
2C06  4A58      ITEM 'JX',GETJX[ .ASCII 'JX'
2C08  A9      +      DEFF GETJX[ .BYTE (GETJX>8)!80H
2C09  4D      +      .BYTE GETJX&0FFH
      ]
2C0A  4A59      ITEM 'JY',GETJY[ .ASCII 'JY'
2C0C  A9      +      DEFF GETJY[ .BYTE (GETJY>8)!80H
2C0D  59      +      .BYTE GETJY&0FFH
      ]
2C0E  4B50      ITEM 'KP',GETKB[ .ASCII 'KP'
2C10  A4      +      DEFF GETKB[ .BYTE (GETKB>8)!80H
2C11  BA      +      .BYTE GETKB&0FFH
      ]

```



.MAIN. -

```

2C12  5058          ITEM  'PX',PIXFUN[  .ASCII  'PX'
2C14  A2            +      DEFF PIXFUN[  .BYTE  (PIXFUN>8)!80H
2C15  D0            +      .BYTE  PIXFUN&0FFH
                +]
2C16  26           ITEM  '&',IOFUN[  .ASCII  '&'
2C17  A4            +      DEFF IOFUN[  .BYTE  (IOFUN>8)!80H
2C18  D4            +      .BYTE  IOFUN&0FFH
                +]
2C19  414253       ITEM  'ABS',ABS[  .ASCII  'ABS'
2C1C  A9            +      DEFF ABS[  .BYTE  (ABS>8)!80H
2C1D  21            +      .BYTE  ABS&0FFH
                +]
2C1E  535A         ITEM  'SZ',SIZE[  .ASCII  'SZ'
2C20  A9            +      DEFF SIZE[  .BYTE  (SIZE>8)!80H
2C21  28            +      .BYTE  SIZE&0FFH
                +]
2C22  A8           DEFF NOTF[  .BYTE  (NOTF>8)!80H
2C23  DD            +      .BYTE  NOTF&0FFH
                +]
2C24  3E3D         TAB6:  ITEM  '>=',XPR1  ; RELATION OPS[ .ASCII  '>='
2C26  A8            +      DEFF XPR1  [  .BYTE  (XPR1  >8)!80H
2C27  04            +      .BYTE  XPR1  &0FFH
                +]
2C28  23           ITEM  '#',XPR2[  .ASCII  '#'
2C29  A8            +      DEFF XPR2[  .BYTE  (XPR2>8)!80H
2C2A  0A            +      .BYTE  XPR2&0FFH
                +]
2C2B  3E           ITEM  '>',XPR3[  .ASCII  '>'
2C2C  A8            +      DEFF XPR3[  .BYTE  (XPR3>8)!80H
2C2D  10            +      .BYTE  XPR3&0FFH
                +]
2C2E  3D           ITEM  '=',XPR5[  .ASCII  '='
2C2F  A8            +      DEFF XPR5[  .BYTE  (XPR5>8)!80H
2C30  1F            +      .BYTE  XPR5&0FFH
                +]
2C31  3C3D         ITEM  '<=',XPR4[  .ASCII  '<='
2C33  A8            +      DEFF XPR4[  .BYTE  (XPR4>8)!80H
2C34  17            +      .BYTE  XPR4&0FFH
                +]
2C35  3C           ITEM  '<',XPR6[  .ASCII  '<'
2C36  A8            +      DEFF XPR6[  .BYTE  (XPR6>8)!80H
2C37  25            +      .BYTE  XPR6&0FFH
                +]
2C38  A8           DEFF XPR7[  .BYTE  (XPR7>8)!80H
2C39  2B            +      .BYTE  XPR7&0FFH
                +]
2C3A          RANEND == .
2C3A  3A 4E86       GLED:  LDA  EDFLG
2C3D  A7           ANA  A
2C3E  2814         JRZ  GLEDA
2C40  11 4EBA      LXI  D,BUFFER
2C43  CD 2ABF      CALL TSTNUM
2C46  CD 2A04      CALL FNDLN

```

.MAIN. -

```

2C49 3E3F          MVI    A,'?'
2C4B C0           RNZ
2C4C 13           INX    D
2C4D CD 2C59      CALL   GLEDB
2C50 AF          XRA    A
2C51 32 4E86      STA    EDFLG
2C54 ED5B 4E87   GLEDA: LDED  EDPTR
2C58 DF          RST    RSTLDE
2C59 13           GLEDB: INX    D
2C5A ED53 4E87   SDED  EDPTR
2C5E C9           RET
2C5F 11 4EBA      GETLN: LXI    D,BUFFER
2C62 32 4E86      STA    EDFLG
2C65 CD 4E9B      GL1:  CALL   OUTCH  ; PROMPT OR ECHO
2C68 C5           GL2:  PUSH   B
2C69 E5           PUSH   H
2C6A D5           PUSH   D
                ; PLACE UP CURSOR BLOCK
2C6B 0EAA         MVI    C,0AAH
2C6D CD 2DFA      CALL   CURSE
                ; RETURN CHAR FROM NEXT LINE #
2C70 21 4E8D      GL2A: LXI    H,NLLNCT
2C73 7E           MOV    A,M      ; SENSE FLAG
2C74 A7           ANA    A
2C75 2838         JRZ    GL2C
2C77 35           DCR    M
                ; FIRST TIME THRU?
2C78 FE05         CPI    5
2C7A 200C         JRNZ  GL2B  ; JUMP IF NOT
                ; GET PREVIOUS LINE # AND BUMP IT
2C7C 2A 4E8F      LHLD  OLDLN
2C7F 11 000A      LXI    D,10
2C82 19           DAD   D
2C83 CBBC         RES    7,H      ; ALLOW NEG
2C85 22 4E8B      GL2J: SHLD  NLLNLN ; MOVE TO WORKING RAM CELL
                ; COMPUTE DIVISION SUBTRACTOR
2C88 FF           GL2B: INDEXW 1[INTP%[.IFE .INTP.,[RST 7]]
2C89 5B           +.BYTE 90+1]
2C8A 2F4B         .WORD  TBLDIV-2
2C8C 2A 4E8B      LHLD  NLLNLN
2C8F 0600         MVI    B,0
2C91 A7           GL2E: ANA    A
2C92 ED52         DSBC  D
2C94 FA 2C9A      JM    GL2F
2C97 04           INR    B
2C98 18F7         JMPR  GL2E
2C9A 19           GL2F: DAD   D
2C9B 22 4E8B      SHLD  NLLNLN
2C9E 21 4E8E      LXI    H,NLLNZS
2CA1 78           MOV    A,B
2CA2 A7           ANA    A
2CA3 2005         JRNZ  GL2G
2CA5 7E           MOV    A,M

```

.MAIN. -

```

2CA6  A7          ANA    A
2CA7  28C7        JRZ    GL2A  ; YES - JUMP BACK
2CA9  AF          XRA    A
2CAA  C630        GL2G:  ADI    '0'  ; MAKE ASCII
2CAC  77          MOV    M,A  ; SET NONZERO FLAG
2CAD  180B        JMPR   GL2D
      ; NOTHIN FANCY
2CAF  CD 4E98    GL2C:  CALL   CHKIO  ; GET NORMAL CHARACTER
2CB2  D1          POP    D
2CB3  12          STAX   D      ; STUFF CHAR AS DELIMITER
2CB4  D5          PUSH   D
2CB5  FE66        CPI    EDKEY
2CB7  CC 2C3A    CZ      GLED
2CBA  D1          GL2D:  POP    D
2CBB  E1          POP    H
2CBC  C1          POP    B
2CBD  12          GL3:  STAX   D
2CBE  FE1F        CPI    RUBOUT
2CC0  2031        JRNZ   GL4
2CC2  7B          MOV    A,E
2CC3  FEBA        CPI    BUFFER&0FFH
2CC5  28A1        JRZ    GL2
2CC7  1B          DCX    D
2CC8  1A          LDAX   D
2CC9  FE68        CPI    68H  ; TOKEN TO RUB OUT?
2CCB  3007        JRNC   TOKIN
2CCD  CD 2EBE    CALL   PNOTE
2CD0  3E1F        MVI    A,RUBOUT
2CD2  1891        JMPR   GL1
2CD4  D5          TOKIN:  PUSH   D
2CD5  CD 2F2A    CALL   TOKEPT
2CD8  7E          TOKER:  MOV    A,M
2CD9  E5          PUSH   H
2CDA  E67F        ANI    7FH
2CDC  CD 2EBE    CALL   PNOTE
2CDF  3E1F        MVI    A,RUBOUT
2CE1  CD 2D13    CALL   VDM
2CE4  E1          POP    H
2CE5  7E          MOV    A,M
2CE6  23          INX    H
2CE7  07          RLC
2CE8  30EE        JRNC   TOKER
2CEA  3E1F        TOKEQ:  MVI    A,RUBOUT
2CEC  CD 4E9B    CALL   OUTCH  ; ECHO ONE RUBOUT CHAR
2CEF  D1          POP    D
2CF0  C3 2C68    GL9:  JMP    GL2
2CF3  EE0D        GL4:  XRI    CR
2CF5  280A        JRZ    GL5
2CF7  7B          MOV    A,E
2CF8  FE22        CPI    BUFEND&0FFH
2CFA  28F4        JRZ    GL9
2CFC  1A          LDAX   D
2CFD  13          INX    D

```

.MAIN. -

```

2CFE  C3 2C65          JMP     GL1
2D01  13              GL5:  INX     D
2D02  13              INX     D
2D03  3D              DCR     A
2D04  12              STAX    D
2D05  1B              DCX     D
2D06  3E0D           CRLF:  MVI     A,CR
2D08  C3 4E9B         JMP     OUTCH
; SUBROUTINE TO SIMULATE A CHARACTER DISPLAY IN
; THE ARCADE FRAME BUFFER.  THE SIMULATED VDM HAS
; DIMENSIONS 26 CHARS BY 11 LINES.  THE CHARACTER GRAPHICS ARE 5 X 7
; IN A 6 X 8 FRAME.  ALTERNATE FONT IS USED TO GET THIS.
; THE 64 UPPER CASE ASCII CHARACTERS ARE DISPLAYED BY THIS
; HANDLER.  THE ASCII CONTROL CHARACTERS CARRIAGE RETURN AND
; RUBOUT ARE ALSO PROCESSED BY THIS HANDLER.  CR CAUSES
; THE DISPLAY TO GO TO THE NEXT LINE OF THE DISPLAY, WITH
; SCROLL UP IF NECESSARY.  RUBOUT CAUSES THE CURSOR TO MOVE
; BACKWARDS ONE CHARACTER POSITION.
; CHARACTER TO DISPLAY IS IN A.  THE ALTERNATE REGISTER SET
; IS USED.
2D0B  F5             XOUTCH: PUSH  PSW
2D0C  D9             EXX
2D0D  CD 2D13        CALL   VDM
2D10  F1             POP   PSW
2D11  D9             EXX
2D12  C9             RET
; SOME FUNNY GUYS ENTER HERE
2D13  FE0D           VDM:  CPI     CR
2D15  282E           JRZ   VDM0CR
2D17  FE1F           CPI     RUBOUT ; TRANSLATE TRASH TO ?
2D19  280F           JRZ   VDM1
2D1B  3804           JRC   FILT1
2D1D  FE78           CPI     78H
2D1F  3802           JRC   FILT2
2D21  3E3F           FILT1: MVI   A,'?'
2D23  FE68           FILT2: CPI   68H ; TOKEN TO PRINT?
2D25  304F           JRNC  TOKEP ; JUMP IF SO
; PLAY NOTE FOR THIS CHAR
2D27  CD 2EBE        CALL   PNOTE
; NON NEW LINE CHAR - UNWRITE OLD CURSOR
2D2A  CD 2D8B        VDM1:  CALL  UCURSE
2D2D  CD 2449        CALL  LDVDMC
2D30  FE1F           CPI     RUBOUT ; WAS THAT RUBOUT?
2D32  201F           JRNZ  VDM3 ; JUMP IF NOT
; RUBOUT ENTERED - SO RUB OUT
2D34  7D             MOV    A,L ; GET X
2D35  A7             ANA   A ; IS X = 0?
2D36  2805           JRZ   VDM2 ; YES - JUMP
2D38  D606           SUI   6 ; NO - BACKUP X
2D3A  6F             MOV   L,A
2D3B  1831           JMPR  VDMDN1 ; AND JOIN STORE BACK
2D3D  2E96           VDM2:  MVI   L,150
2D3F  7C             MOV   A,H

```

.MAIN. -

```

2D40  D608          SUI    8
2D42  67            MOV    H,A
2D43  1829          JMPR   VMDMN1
      ; NEW LINE CHAR - DID WE JUST WRAP AROUND
2D45  3A 4E7C      VDM0CR: LDA   VDMNLF ; CHECK OLD GLORY
2D48  A7            ANA    A
2D49  2026          JRNZ   VMDMDON ; YES - SKIP DIDDLING
2D4B  CD 2D8B      CALL   UCURSE ; NO - UNWRITE CURSOR
2D4E  CD 2D8F      CALL   NEWLIN ; GO TO NEXT LINE
2D51  181E          JMPR   VMDMDON ; AND QUIT
      ; NORMAL CHARACTER ENTERED - DISPLAY IT
2D53  54            VDM3:  MOV    D,H    ; COORDINATES TO DE
2D54  5D            MOV    E,L
2D55  F680          ORI    80H    ; ALT FONT THE CHAR
2D57  0E18          MVI    C,011000B ; OR WRITE THE CHAR
2D59  DD21 4EA0    LXI    X,ALTFON ; USING ALTERNATE CHAR FONT
2D5D  FF            CHRDIS ; IT    1[INTP%[.IFE .INTP.,[RST 7]]
2D5E  32            +.BYTE 50+0]
2D5F  7D            MOV    A,L    ; ADVANCE X POINTER
2D60  C606          ADI    6
2D62  6F            MOV    L,A
2D63  FE9C          CPI    156   ; END OF LINE?
2D65  2007          JRNZ   VMDMN1 ; NO - JUMP
2D67  CD 2D8F      CALL   NEWLIN ; YES - NEW    1 LINE
2D6A  3E01          MVI    A,1    ; AND SET NEW LINE FORCED FLAG
2D6C  1804          JMPR   VMDMN2
2D6E  CD 2463      VMDMN1: CALL  STVDMC
2D71  AF            VMDMDON: XRA   A    ; CLEAR NEW LINE FORCED FLAG
2D72  32 4E7C      VMDMN2: STA   VDMNLF
2D75  C9            RET
      ; ROUTINE TO DISPLAY A TOKEN IN FULL FORM
2D76  CD 2F2A      TOKEP: CALL  TOKEPT
2D79  7E            TOKEP1: MOV   A,M
2D7A  E67F          ANI    7FH
2D7C  E5            PUSH   H
2D7D  CD 2D13      CALL   VDM
2D80  E1            POP    H
2D81  7E            MOV    A,M
2D82  23            INX    H
2D83  07            RLC
2D84  30F3          JRNC   TOKEP1
2D86  3E20          TOKEP2: MVI   A,' ' ; PUT SPACE AFTER TOKEN
2D88  C3 2D13      JMP    VDM    ; AND GO HOME
      ; SUBROUTINE TO UNWRITE THE CURSOR
2D8B  0E00          UCURSE: MVI   C,0
2D8D  186B          JMPR   CURSE
      ; SUBROUTINE TO DISPLAY NEW LINE
2D8F  CD 2449      NEWLIN: CALL  LDVDMC
      ; IS SCROLL UP NEEDED?
2D92  2E00          MVI    L,0
2D94  7C            MOV    A,H
2D95  FE50          CPI    80
2D97  203E          JRNZ   NEWL1 ; JUMP IF NOT NEEDED

```

.MAIN. -

```

; SCROLL UP IS NEEDED
2D99   CD 2463           CALL   STVDMC
; WHAT MODE SHALL WE USE?
2D9C   3A 4E7A         LDA    SCRMOD
2D9F   3D              DCR    A
2DA0   C8              RZ
2DA1   3D              DCR    A
2DA2   21 4C80         LXI    H,4C80H
2DA5   2848            JRZ    CLRFRZ
2DA7   3D              DCR    A
2DA8   2838            JRZ    CLRENT
2DAA   3D              DCR    A
2DAB   2830            JRZ    CLRFRZ
2DAD   21 4DC0         LXI    H,4DC0H
2DB0   7E              SCRL9: MOV   A,M
2DB1   E655            ANI    01010101B
2DB3   77              MOV   M,A
2DB4   23              INX    H
2DB5   7D              MOV   A,L
2DB6   FE20            CPI    20H
2DB8   20F6            JRNZ   SCRL9
2DBA   0604            MVI    B,4
2DBC   C5              SCRLP: PUSH  B
2DBD   21 4000         LXI    H,NORMEM
2DC0   11 4050         LXI    D,NORMEM+80
2DC3   01 C00E         LXI    B,0C00EH
2DC6   1A              SCRUP: LDAX  D
2DC7   AE              XRA    M
2DC8   E6AA            ANI    10101010B
2DCA   AE              XRA    M
2DCB   77              MOV   M,A
2DCC   23              INX    H
2DCD   13              INX    D
2DCE   10F6            DJNZ   SCRUP
2DD0   0D              DCR    C
2DD1   20F3            JRNZ   SCRUP
2DD3   C1              POP    B
2DD4   10E6            DJNZ   SCRLP
2DD6   C9              RET

;
2DD7   C608            NEWL1: ADI    8
2DD9   67              MOV   H,A
2DDA   C3 2463         JMP   STVDMC
; CLEAR COMMAND
2DDD   CLRFRZ:
2DDD   CD 2E94         CALL   KEYSN
2DE0   28FB            JRZ    CLRFRZ
; RESET VDM GOODIES
2DE2   D5              CLRENT: PUSH  D
2DE3   FF              MOVE   1[INTP%[.IFE .INTP.,[RST 7]]
2DE4   5F              +.BYTE 94+1]
2DE5   4E5C            .WORD  VDMX
2DE7   0004            .WORD  4

```

.MAIN. -

```

2DE9  202A          .WORD  INIDEV+6
2DEB  D1            POP      D
2DEC  21 4000      LXI      H,4000H
2DEF  7E            CLRLP:  MOV     A,M
2DF0  E655          ANI      01010101B
2DF2  77            MOV     M,A
2DF3  23            INX     H
2DF4  7C            MOV     A,H
2DF5  FE4E          CPI     4EH
2DF7  20F6          JRNZ   CLRLP
2DF9  C9            RET

; SUBROUTINE TO PAINT CURSOR
; C = DATA TO PAINT 00 OR AA
2DFA  F5            CURSE:  PUSH   PSW
2DFB  CD 2449      CALL   LDVDMC
2DFE  EB            PCURS1: XCHG
2DF7  AF            XRA     A
2E00  FF            RELAB1[INTP%[.IFE .INTP.,[RST 7]]
2E01  3A            +.BYTE  58+0]
2E02  D30C          OUT     MAGIC
2E04  EB            XCHG
2E05  79            MOV     A,C
2E06  01 0806      LXI     B,0806H
2E09  CD 2365      CALL   BOXPUT
2E0C  F1            POP     PSW
2E0D  C9            RET

; NEW KEYBOARD HANDLER
; WITH SHIFT KEY ROLLOVER
2E0E  CD 2E94      XCHKIO: CALL  KEYSCN ; MAKE SURE PREVIOUS KEY RELEASED
2E11  20FB          JRNZ   XCHKIO
; AWAIT DEBOUNCE TIMER COUNTDOWN
2E13  21 4E7D      CHKIO0: LXI   H,KEYTMR
2E16  3606          MVI   M,6 ; SET IT
2E18  7E            LOOPER: MOV  A,M
2E19  A7            ANA   A
2E1A  20FC          JRNZ   LOOPER
; SAVE BACKGROUND COLOR
2E1C  3A 4E56      LDA   DEVCL0
2E1F  F5            PUSH  PSW
; ASSUME FIRST LEVEL KEYCODE
2E20  21 2F5E      LXI   H,FIRSTL
2E23  E5            GETK1: PUSH  H ; SAVE TABLE PTR
; SCAN ONLY FOR SHIFT KEYS
2E24  21 2FB1      LXI   H,KTBL4
2E27  11 FFEB      LXI   D,-21 ; ** SIZE OF LOOKUP TABLE
2E2A  01 0414      LXI   B,0414H
2E2D  ED78          GETK2: INP   A ; INPUT FROM PORT
2E2F  E620          ANI   20H ; SHIFT KEY DOWN?
2E31  2007          JRNZ  GETK3 ; JUMP IF YEP
2E33  19            DAD   D ; ELSE TO NEXT TABLE
2E34  0C            INR   C ; AND PORT
2E35  10F6          DJNZ  GETK2
; NO SHIFT KEY IS DOWN - USE WHATEVER WE HAD BEFORE

```

.MAIN. -

```

2E37 E1          POP      H
2E38 1806       JMPR    GETK5
          ; A SHIFT KEY IS DOWN - SAME OLD STORY?
2E3A D1          GETK3:  POP    D          ; DISCARD OLD BELIEFS
2E3B 7E          MOV     A,M        ; SET NEW COLOR
2E3C 32 4E56     STA     DEVCL0
2E3F 23          GETK4:  INX    H          ; SKIP COLOR BYTE
          ; NOW SCAN FOR ANY 'NORMAL' KEY DEPRESSION
2E40 CD 2E94     GETK5:  CALL   KEYSCN
2E43 28DE       JRZ     GETK1    ; JUMP IF NO KEY DOWN
          ; WE GOT ONE - CONVERT TO ASCII
2E45 3D          DCR     A          ; BY TABLE LOOKUP
2E46 4F          MOV     C,A
2E47 0600       MVI     B,0
2E49 09          DAD     B
2E4A F1          POP     PSW        ; RESTORE COLOR
2E4B 32 4E56     STA     DEVCL0
2E4E 7E          MOV     A,M        ; GET CODE
2E4F A7          ANA     A          ; A HLT PERCHANCE?
2E50 283C       JRZ     INIJMP    ; YEP - RESET
2E52 FE01       CPI     1          ; AN ERROR?
2E54 28B8       JRZ     XCHKIO    ; YEP - GO DOIT AGAIN
          ; GOOD KEY...
2E56           CHKIO2:
2E56 07          RLC
2E57 DC 2E69     CC          WCLICK
2E5A 7E          MOV     A,M
2E5B E67F       ANI     7FH
2E5D FE67       CPI     NLLN
2E5F C0          RNZ
2E60 21 0005     LXI     H,5
2E63 22 4E8D     SHLD   NLLNCT  ; SET FLAG AND ZERO SUPPRESS
2E66 3E0D       MVI     A,CR    ; PASS BACK CR AS FIRST CHAR
2E68 C9          RET
          ;
          ; NEW CLICK ROUTINE
2E69           WCLICK:
2E69 3EFD       MVI     A,G0
2E6B 32 4E81     STA     MUZTON
2E6E 3A 4E5A     LDA     DEVTEM
2E71 3D          DCR     A
2E72 F8          RM
2E73 3E01       MVI     A,1
2E75 32 4E7F     STA     NEWTMR
2E78 C9          RET
          ; SUBROUTINE TO CHECK FOR HLT KEY WHILE PGM RUNNING
2E79 C5          WHATSU:  PUSH   B
2E7A D5          PUSH   D
2E7B CD 2E94     CALL   KEYSCN  ; GET KEY CODE
2E7E D602       SUI     2          ; FREEZE?
2E80 2805       JRZ     FRZKEY
2E82 3D          DCR     A
2E83 2809       JRZ     INIJMP

```



.MAIN. -

```

2E85 180A          JMPR   FRZGBK ; ELSE GO BACK TO CALLER
2E87 CD 2E94      FRZKEY: CALL  KEYSCN ; SCAN FOR NONZERO KEY TO REL
2E8A 28FB          JRZ    FRZKEY
2E8C FE03          CPI     3 ; HLT NAILED?
2E8E CA 2541      INIJMP: JZ    INIT
2E91 D1            FRZGBK: POP  D
2E92 C1           POP  B
2E93 C9           RET

; SUBROUTINE TO SCAN TINY BASIC KEYBOARD
2E94 01 0414      KEYSCN: LXI  B,0414H ; B = CNT, C = PORT #
2E97 11 4E83      LXI  D,KEYTRK ; DE = KEYBOARD MEMORY
2E9A AF          XRA   A
2E9B FF          RANGED[INTP%[.IFE .INTP.,[RST 7]]
2E9C 76          +.BYTE 118+0]
2E9D ED78        KYSCN1: INP  A ; LOOK AT COLUMN
2E9F E61F        ANI  1FH ; ISOLATE THE RELEVANT
2EA1 2006        JRNZ  KYSCN2 ; JUMP IF BITS HIGH
2EA3 0C          INR  C ; BUMP PORT #
2EA4 10F7        DJNZ  KYSCN1
2EA6 AF          XRA   A ; SET ZERO STATUS
2EA7 12          STAX  D ; NOTHIN - SAY ZIP
2EA8 C9          RET

; DEPRESSION FOUND - JUMP UP AND DOWN
2EA9 05          KYSCN2: DCR  B
2EAA 0E00        MVI  C,0 ; COME UP WITH BIT #
2EAC 0F          KYSCN4: RRC  ; SHIFT BIT OVER
2EAD 3803        JRC   KYSCN3 ; JUMP IF THE ONE
2EAF 0C          INR  C ; ELSE COUNT UP
2EB0 18FA        JMPR  KYSCN4 ; AND TRY AGAIN

; FOUND BIT - ASSEMBLE KEYCODE
2EB2 79          KYSCN3: MOV  A,C ; BIT # TO A
2EB3 07          RLC  ; * 4
2EB4 07          RLC
2EB5 B0          ORA  B ; COMBINE WITH COL #
2EB6 3C          INR  A
2EB7 47          MOV  B,A
2EB8 1A          LDAX D
2EB9 A8          XRA  B
2EBA 78          MOV  A,B
2EBB C8          RZ   ; QUIT IF THE SAME
2EBC 12          STAX D ; ELSE UPDATE TRACKER
2EBD C9          RET

; SUBROUTINE TO PLAY A NOTE
2EBE E5          PNOTE: PUSH  H
2EBF D5          PUSH  D
2EC0 F5          PUSH  PSW
2EC1 67          MOV  H,A

; WAIT FOR PREVIOUS PARAMETERS TO BE EATEN
2EC2 3A 4E7F      PRWAIT: LDA  NEWTMR
2EC5 A7          ANA  A
2EC6 20FA        JRNZ  PRWAIT ; LOOP
2EC8 7C          MOV  A,H
2EC9 EE20        XRI  ' '

```

.MAIN. -

```

2ECB 282B          JRZ  TSTOR
2ECD 7C           MOV  A,H
2ECE FE63         CPI  63H
2ED0 2837         JRZ  PNOTDV
2ED2 FE62         CPI  62H
2ED4 2836         JRZ  PNOTML
2ED6 FE2B         CPI  '+'
2ED8 2839         JRZ  PNOTPL
2EDA FE2D         CPI  '-'
2EDC 2838         JRZ  PNOTMN
2EDE FE0D         CPI  CR
2EE0 281F         JRZ  PNOTCL
2EE2 D630         SUI  '0'
2EE4 2834         JRZ  PNOTZ
2EE6 3D           DCR  A
2EE7 FE07         CPI  7
2EE9 3805         JRC  ANSW
2EEB 3E6C         MVI  A,6CH
2EED 94           SUB  H
2EEE 1808         JMPR TSTOR
2EF0 21 4E82      ANSW: LXI  H,SHARPF
2EF3 86           ADD  M
2EF4 FF           INDEXB 1[INTP%[.IFE .INTP.,[RST 7]]
2EF5 5D           +.BYTE 92+1]
2EF6 2F38         .WORD DICKY
2EF8 32 4E81      TSTOR: STA  MUZTON
2EFB 3A 4E5A      LDA  DEVTEM
2EFE 32 4E7F      STA  NEWTMR
2F01 AF           PNOTCL: XRA  A
2F02 32 4E82      PSHARP: STA  SHARPF
2F05 F1           LINKB: POP  PSW
2F06 D1           POP  D
2F07 E1           POP  H
2F08 C9           RET
2F09 3E8F         PNOTDV: MVI  A,OA1
2F0B 11           .BYTE 11H
2F0C 3E23         PNOTML: MVI  A,OA3
2F0E 32 4E80      STA  MUZMO
2F11 18F2         JMPR  LINKB
2F13 3E07         PNOTPL: MVI  A,7
2F15 11           .BYTE 11H
2F16 3E0E         PNOTMN: MVI  A,14
2F18 18E8         JMPR  PSHARP
2F1A 21 4E7E      PNOTZ: LXI  H,MUZTMR
2F1D 3A 4E5A      LDA  DEVTEM
2F20 A7           ANA  A
2F21 FA 2F05      JM   LINKB
2F24 F3           DI
2F25 86           ADD  M
2F26 77           MOV  M,A
2F27 FB           EI
2F28 18DB         JMPR  LINKB
; SUBROUTINE TO POINT AT A TOKEN

```

.MAIN. -

```

2F2A 21 2199      TOKEPT: LXI   H,TOKTXT      ; POINT AT TEXT LIST
2F2D D668          SUI     68H
2F2F C8           JOKEP1: RZ           ; QUIT IF POINTING AT EM
2F30 CB7E        JOKEP2: BIT     7,M      ; MOVE PAST NEXT WORD
2F32 23          INX     H
2F33 28FB        JRZ     JOKEP2
2F35 3D          DCR     A
2F36 18F7        JMPR    JOKEP1      ; LOOP BACK AND CHECK
                ; DICKS MUSIC SYSTEM NOTE LOOKUP TABLE
2F38 5E544A463E37 DICKY: .BYTE  C2,D2,E2,F2,G2,A2,B2
2F3F 594F46423B34 .BYTE  CS2,DS2,F2,FS2,GS2,AS2,C3
2F46 64594F4A423B .BYTE  B1,CS2,DS2,E2,FS2,GS2,AS2
2F4D 0001        TBLDIV: .WORD  1
2F4F 000A        .WORD  10
2F51 0064        .WORD  100
2F53 03E8        .WORD  1000
2F55 2710        .WORD  10000
2F57 1A          IGNBLK: LDAX   D          ; *** IGNBLK ***
2F58 FE20        CPI     ' '          ; IGNORE BLANKS
2F5A C0          RNZ           ; IN TEXT (WHERE DE->)
2F5B 13          INX     D          ; AND RETURN THE FIRST
2F5C 18F9        JMPR    IGNBLK      ; NON-BLANK CHAR. IN A
                ; TABLE OF FIRST LEVEL KEYCODES
2F5E          FIRSTL:
2F5E 8D          .BYTE  CR+80H
2F5F 66          .BYTE  EDKEY
2F60 00          .BYTE  0
2F61 E3          .BYTE  63H+80H
2F62 37          .BYTE  '7'
2F63 38          .BYTE  '8'
2F64 39          .BYTE  '9'
2F65 E2          .BYTE  62H+80H
2F66 34          .BYTE  '4'
2F67 35          .BYTE  '5'
2F68 36          .BYTE  '6'
2F69 AD          .BYTE  '-' +80H
2F6A 31          .BYTE  '1'
2F6B 32          .BYTE  '2'
2F6C 33          .BYTE  '3'
2F6D AB          .BYTE  '+' +80H
2F6E 20          .BYTE  ' '
2F6F B0          .BYTE  '0' +80H
2F70 9F          .BYTE  RUBOUT+80H
2F71 3D          .BYTE  '='
                ; FIRST SHIFT KEY
2F72          KTBL1:
2F72 A7          .BYTE  0A7H      ; FIRST SHIFT KEY COLOR
2F73 8D          .BYTE  CR+80H
2F74 66          .BYTE  EDKEY
2F75 00          .BYTE  0
2F76 01          .BYTE  1
2F77 41          .BYTE  'A'
2F78 44          .BYTE  'D'

```

.MAIN. -

```

2F79 47          .BYTE  'G'
2F7A 4A          .BYTE  'J'
2F7B 4D          .BYTE  'M'
2F7C 50          .BYTE  'P'
2F7D 53          .BYTE  'S'
2F7E 56          .BYTE  'V'
2F7F 59          .BYTE  'Y'
2F80 5F          .BYTE  5FH
2F81 5E          .BYTE  5EH
2F82 26          .BYTE  '&'
2F83 24          .BYTE  '$'
2F84 3C          .BYTE  '<'
2F85 28          .BYTE  '('
2F86 23          .BYTE  '#'
          ; SECOND SHIFT KEY
2F87          KTBL2:
2F87 5F          .BYTE  05FH ; SECOND SHIFT KEY COLOR
2F88 8D          .BYTE  CR+80H
2F89 2F          .BYTE  2FH
2F8A 00          .BYTE  0
2F8B 5B          .BYTE  5BH
2F8C 42          .BYTE  'B'
2F8D 45          .BYTE  'E'
2F8E 48          .BYTE  'H'
2F8F 4B          .BYTE  'K'
2F90 4E          .BYTE  'N'
2F91 51          .BYTE  'Q'
2F92 54          .BYTE  'T'
2F93 57          .BYTE  'W'
2F94 5A          .BYTE  'Z'
2F95 27          .BYTE  27H
2F96 2E          .BYTE  '.'
2F97 40          .BYTE  '@'
2F98 2C          .BYTE  ','
2F99 22          .BYTE  22H
2F9A 3B          .BYTE  ';'
2F9B 25          .BYTE  '%'
          ; TABLE THE THIRD
2F9C          KTBL3:
2F9C 0F          .BYTE  0FH ; THIRD SHIFT KEY COLOR
2F9D 8D          .BYTE  CR+80H
2F9E 5C          .BYTE  5CH
2F9F 00          .BYTE  0
2FA0 5D          .BYTE  5DH
2FA1 43          .BYTE  'C'
2FA2 46          .BYTE  'F'
2FA3 49          .BYTE  'I'
2FA4 4C          .BYTE  'L'
2FA5 4F          .BYTE  'O'
2FA6 52          .BYTE  'R'
2FA7 55          .BYTE  'U'
2FA8 58          .BYTE  'X'
2FA9 21          .BYTE  '!'

```

.MAIN. -

```

2FAA 61          .BYTE 61H
2FAB 60          .BYTE 60H
2FAC 2A          .BYTE '*'
2FAD 3F          .BYTE '?'
2FAE 3E          .BYTE '>'
2FAF 29          .BYTE ')'
2FB0 3A          .BYTE ':'
          ; TOKEN KEY
2FB1          KTBL4:
2FB1 77          .BYTE 77H      ; WORDS KEY COLOR
2FB2 67          .BYTE NLLN
2FB3 66          .BYTE EDKEY
2FB4 6A          .BYTE 6AH
2FB5 68          .BYTE 68H
2FB6 72          .BYTE 72H
2FB7 77          .BYTE 77H
2FB8 75          .BYTE 75H
2FB9 6B          .BYTE 6BH
2FBA 6F          .BYTE 6FH
2FBB 70          .BYTE 70H
2FBC 76          .BYTE 76H
2FBD 6D          .BYTE 6DH
2FBE 69          .BYTE 69H
2FBF 6C          .BYTE 6CH
2FC0 71          .BYTE 71H
2FC1 6E          .BYTE 6EH
2FC2 66          .BYTE EDKEY
2FC3 73          .BYTE 73H
2FC4 01          .BYTE 1
2FC5 74          .BYTE 74H
          ; SUBROUTINE TO LDAX  D FROM SCREEN TEXT MEMORY IF NECESSARY
2FC6 CB7A      LDE:  BIT 7,D
2FC8 2810      JRZ  LDE1
2FCA EB        XCHG
2FCB 29        DAD  H
2FCC 7E        MOV  A,M
2FCD 07        RLC
2FCE 23        INX  H
2FCF AE        XRA  M
2FD0 E6AA      ANI  10101010B
2FD2 AE        XRA  M
2FD3 37        STC
2FD4 CB1C      RARR  H
2FD6 CB1D      RARR  L
2FD8 EB        XCHG
2FD9 C9        RET
2FDA 1A      LDE1: LDAX  D
2FDB C9        RET
          ; DOUBLE STORE INTO HL
2FDC          STDEHL:
2FDC 7B        MOV  A,E
2FDD CD 2FE2   CALL  STHL
2FE0 23        INX  H

```

.MAIN. -

```
2FE1 7A          MOV      A,D
          ; THEN FALL INTO ...
          ; SUBROUTINE TO STORE MOV      M,A
2FE2 CB7C      STHL:  BIT      7,H
2FE4 2818      JRZ      STHL1
2FE6 C5        PUSH     B
2FE7 4F        MOV      C,A
2FE8 29        DAD      H
2FE9 0F        RRC
2FEA AE        XRA      M
2FEB E655      ANI      01010101B
2FED AE        XRA      M
2FEE 77        MOV      M,A
2FEF 23        INX      H
2FF0 79        MOV      A,C
2FF1 AE        XRA      M
2FF2 E655      ANI      01010101B
2FF4 AE        XRA      M
2FF5 77        MOV      M,A
2FF6 37        STC
2FF7 CB1C      RARR     H
2FF9 CB1D      RARR     L
2FFB 79        MOV      A,C
2FFC C1        POP      B
2FFD C9        RET
2FFE 77      STHL1:  MOV      M,A
2FFF C9        RET
          .END
```

.MAIN. -

+++++ SYMBOL TABLE +++++

0A7	47	B300:D8	U	11F	4X	B200:F0	U	3%F	1B	B300:E8	U	5F	1	B100:EC	M	51F	N	B300:E0	U	5M91YA	2622:23	U	
A0		00E1		A1		0070		A2		0037		A3		001B		A4		000D		A5		0006	
ALKEYS	0214		I	AS0		00D4		AS1		006A		AS2		0034		AS3		001A		B0		00C8	
B1		0064		B2		0031		B3		0018		BEGRAM		4FCE	I	BITSPL		00A0	I	BOTRAM		A000	
BOTROM	2000			BOTSCR		4E20		BYTEPL		0028	I	C1		00BD		C2		005E		C3		002E	
C4		0017		C5		000B		C6		0005		C7		0002		CBA		0009	I	CBB		0007	I
CBC	0006		I	CBD		0005	I	CBE		0004	I	CBFLAG		0008	I	CBH		000B	I	CBIXH		0003	I
CBIXL	0002		I	CBIYH		0001	I	CBIYL		0000	I	CBL		000A	I	CHDOWN		0001	I	CHLEFT		0002	I
CHRIGH	0003		I	CHTRIG		0004	I	CHUP		0000	I	COL0L		0004	I	COLOR		0000	I	COL1L		0005	I
COL1R	0001		I	COL2L		0006	I	COL2R		0002	I	COL3L		0007	I	COL3R		0003	I	COLBX		000B	I
COLLST	4FE8		I	COMMA		002C		CONCM		0008	I	CR		000D		CS1		00B2		CS2		0059	
CS3		002C		CS4		0015		CS5		000A		CT0		4FD5	I	CT1		4FD6	I	CT2		4FD7	I
CT3		4FD8	I	CT4		4FD9	I	CT5		4FDA	I	CT6		4FDB	I	CT7		4FDC	I	CTIMER		0203	I
CUNT	4FDD		I	D1		00A8		D2		0054		D3		0029		D4		0014		DEVCL0		4E56	
DEVCL1	4E58			DEVMO		4E62		DEVNM		4E70		DEVNV		4E76		DEVOA		4E64		DEVOB		4E66	
DEVOC	4E68			DEVTEM		4E5A		DEVVA		4E72		DEVVAR		4E56		DEVVB		4E74		DEVVC		4E6E	
DEVVD	4E6A			DEVVR		4E6C		DFTLMT		A70C		DS1		009F		DS2		004F		DS3		0027	
DS4		0013		DS5		0009		DS6		0004		DURAT		4FEA	I	E1		0096		E2		004A	
E3		0025		E4		0012		EDFLG		D5C9:C1	EU	EDKEY		0066		ENDSCR		4FF4	I	F1		008D	
F2		0046		F3		0022		F4		0011		F5		0008		FIRSTC		2000	I	FNTSML		020D	I
FNTSYS	0206		I	FS1		0085		FS2		0042		FS3		0020		FS4		0010		FTBASE		0000	I
FTBYTE	0003		I	FTFSX		0001	I	FTFSY		0002	I	FTPTH		0006	I	FTPTL		0005	I	FTYSIZ		0004	I
G0		00FD		G1		007E		G2		003E		G3		001F		G4		000F		G5		0007	
G6		0003		G7		0001		G8		0000		GAMSTB		4FF8	I	GS0		00EE		GS1		0077	
GS2		003B		GS3		001D		GS4		000E		GSBEND		0007	I	GSBSCR		0001	I	GSBTIM		0000	I
GTMIN5	4FEE		I	GTSECS		4FED	I	HLTPOR		0015		HORAF		000F	I	HORCB		0009	I	HY7 2L	B200:C9	U	
INFBK	000D		I	INLIN		000F	I	INMOD		000E	I	INTST		0008	I	KEY0		0014	I	KEY1		0015	I
KEY2		0016	I	KEY3		0017	I	KEYSEX		4FE3	I	KEYTMR		4E7D		KEYTRK		4E83		LINEND		4E84	
MAGIC	000C		I	MENUST		0218	I	MRFLOP		0006	I	MRLOCK		4FF7	I	MROR		0004	I	MRRROT		0002	I
MRSHTF	0003		I	MRXOR		0005	I	MRXPND		0003	I	MUZMO		4E80		MUZPC		4FCE	I	MUZSP		4FD0	I
MUZTMR	4E7E			MUZTON		4E81		MXSCR		021E	I	NBV 0L	B200:F8	U	NEWTMR		4E7F		NLLN		0067		
NOGAME	0235		I	NOPLAY		0228	I	NORMEM		4000	I	NUMPLY		4FF3	I	OA1		008F		OA2		0047	
OA3		0023		OA4		0011		OA5		0008		OB0		00FE		OC0		00F1		OD1		00D6	
OE1		00BF		OF1		00B4		OG1		00A0		OLDXY		4E60		ON 5	B200:C8	U	OPOT0		4FDF	I	
OPOT1	4FE0		I	OPOT2		4FE1	I	OPOT3		4FE2	I	OSW0		4FE4	I	OSW1		4FE5	I	OSW2		4FE6	I
OSW3		4FE7	I	POT0		001C	I	POT1		001D	I	POT2		001E	I	POT3		001F	I	PRIOR		4FF9	I
PSWCY	0000		I	PSWPV		0002	I	PSWSGN		0007	I	PSWZRO		0006	I	PVOLAB		4FD2	I	PVOLMC		4FD3	I
RANSHT	4FEF		I	REMAIN		4E78		RSTEXP		0002		RSTFIN		0006		RSTIGN		0004		RSTLDE		0003	
RSTPAR	0005			RUBOUT		001F		SCREEN		0000	I	SCRMOD		4E7A		SCT0		0001	I	SCT1		0002	I
SCT2		0003	I	SCT3		0004	I	SCT4		0005	I	SCT5		0006	I	SCT6		0007	I	SCT7		0008	I
SEMI4S	4FDE		I	SENFLG		4FFA	I	SF0		0009	I	SF1		000A	I	SF2		000B	I	SF3		000C	I
SF4		000D	I	SF5		000E	I	SF6		000F	I	SF7		0010	I	SHARPF		4E82		SJ0		0015	I
SJ1		0017	I	SJ2		0019	I	SJ3		001B	I	SKYD		0013	I	SKYU		0012	I	SNDBX		0018	I
SNUL	0000		I	SP0		001C	I	SP1		001D	I	SP2		001E	I	SP3		001F	I	SSEC		0011	I
ST0		0014	I	ST1		0016	I	ST2		0018	I	ST3		001A	I	STIMER		0200	I	SW0		0010	I
SW1		0011	I	SW2		0012	I	SW3		0013	I	SYSRAM		4FCE	I	TIMOUT		4FEC	I	TMR60		4FEB	I
TONEA	0011		I	TONEB		0012	I	TONEC		0013	I	TONMO		0010	I	TOPSCR		4FEA		TXTUNF		4E20	
UMARGT	4FFB		I	URNAL		0FFF	I	USERTB		4FFD	I	VARBGN		4E22		VBBLNK		0006	I	VBCHK		0004	I
VBCH		0003	I	VBCL		0002	I	VBCLAT		0003	I	VBCLMT		0000	I	VBCREV		0001	I	VBDCH		0001	I
VBDCL	0000		I	VBDXH		0004	I	VBDXL		0003	I	VBDYH		0009	I	VBDYL		0008	I	VBMR		0000	I
VBOAH	000E		I	VBOAL		000D	I	VBSACT		0007	I	VBSTAT		0001	I	VBTIMB		0002	I	VBXCHK		0007	I
VBXH		0006	I	VBXL		0005	I	VBYCHK		000C	I	VBYH		000B	I	VBYL		000A	I	VDMNLF		4E7C	
VDMX	4E5C			VDMY		4E5E		VERAF		000E	I	VERBL		000A	I	VIBRA		0014	I	VOICES		4FD4	I

