```
                        TITLE 'INTERACT ROM & OPERATING SYSTEM 2.0 LISTING'
0000                    ORG 0000H
                        ;*****************************************************
                        ;*****************  E Q U A T E S  *******************
                        ;*****************************************************
                        ;
                        ;*********************** CONSTANTS, ROM ENTRY POINTS, AND
                                                 MEMORY MAP ADDRESSES
                        ;
0000            ZEROS   EQU     0000H   ;DATA USED TO CLEAR OUT DOUBLE REGISTER SETS
0008            BS      EQU     08H     ;ASCII VALUE FOR BACKSPACE CHAR
000D            CR      EQU     0DH     ;ASCII VALUE FOR CARRIAGE RETURN
00F0            HIDE1   EQU     00F0H   ;ADR OF ROM'S FIRST HIDDEN INSTRUCTION
0131            HIDE2   EQU     0131H   ;ADR OF ROM'S SECOND HIDDEN INSTRUCTION
0395            HIDE3   EQU     0395H   ;ADR OF ROM'S THIRD HIDDEN INSTRUCTION
0642            LNSTRT  EQU     0642H   ;SCREEN COORDS FOR TEXT OUTPUT AT BOTTOM LEFT
1000            CLRGA   EQU     1000H   ;ADR OF COLOR REGISTER A
1800            CLRGB   EQU     1800H   ;ADR OF COLOR REGISTER B
2003            SDRG1   EQU     2003H   ;ADR OF TOP OF SOUND REGISTER 1
2801            NRMVOL  EQU     2801H   ;ADR IN SOUND REG 2 THAT SETS NORMAL VOLUME
2802            SNDABL  EQU     2802H   ;SOUND REG 2 ADR - ENABLES & INHIBITS SOUND
2803            SDRG2   EQU     2803H   ;ADR OF TOP OF SOUND REGISTER 2
3000            CLKAD   EQU     3000H   ;ADR OF MISCELLANEOUS OUTPUT REGISTER
3600            KYBDBT  EQU     3600H   ;ADR OF THE BOTTOM OF THE KEYBOARD REGISTERS
3606            KYBDTP  EQU     3606H   ;ADR OF THE TOP OF THE KEYBOARD REGISTERS
3607            JOYSTK  EQU     3607H   ;ADR WHERE JOYSTICK DATA IS READ
4000            SCRNTP  EQU     4000H   ;ADR OF TOP OF THE SCREEN OR START OF RAM
48E0            LSTLIN  EQU     48E0H   ;ADR OF LAST LINE ON SCREEN (BELOW VIEW)
```

```
                        ;/********** FIRST ROM VARIABLE STORAGE LOCATIONS ****
                        ;
4C00            GAMEST  EQU     4C00H   ;ADR WHERE ALL GAMES & PRGMS START
5FC0            STKTOP  EQU     5FC0H   ;ROM STK PTR TOP & COPY OF LAST DATA TO CLKAD
5FC1            JOYVAL  EQU     5FC1H   ;ADR OF STORAGE OF BOTH JOYSTICK VALUES
5FC2            K1RG0   EQU     5FC2H   ;KEY TABLE 1, 1ST PASS SETS LOCATION IF HIT...
5FC8            K1RG6   EQU     5FC8H   ;KEY TABLE 1-...2ND PASS CHECKS FOR SET.
5FC9            K2RG0   EQU     5FC9H   ;KEY TABLE 2, INHIBITS KEY SENSE IF SET...
5FCF            K2RG6   EQU     5FCFH   ;KEY TABLE 2-...DECODER SETS, STAYS TIL FREED)
5FD0            KYSTAT  EQU     5FD0H   ;ADR OF KEY HIT OR NOT STATUS
5FD1            LSTCHR  EQU     5FD1H   ;ADR OF STORAGE OF LAST CHAR HIT
5FD2            SHIFT   EQU     5FD2H   ;SHIFT STATUS REGISTER,(RE)SET BY SHIFT LOCK
5FD3            RDSTAT  EQU     5FD3H   ;STORAGE ADR OF READING TAPE STATUS
5FD4            TPSTRT  EQU     5FD4H   ;STORAGE ADR OF TAPE DATA STARTING ADR
5FD6            TPBYTS  EQU     5FD6H   ;STORAGE ADR OF # OF BYTES INCOMING FROM TAPE
5FD8            HDRFLG  EQU     5FD8H   ;READ TAPE STORAGE OF END FLAG FOR RECORD
5FD9            BULKFL  EQU     5FD9H   ;READ TAPE BULK MEMORY FILL DATA BYTE STORAGE
5FDA            TAPWRD  EQU     5FDAH   ;MNEMONIC DERIVED FROM REVERSI
5FDB            CLRS02  EQU     5FDBH   ;ADR OF STORAGE OF COLORS 0 & 2
5FDC            CLRS13  EQU     5FDCH   ;ADR OF STORAGE OF COLORS 1 & 3
5FDD            USRRT   EQU     5FDDH   ;USER SET ADR FOR TAPE DATA TO START LOADING TO
5FDF            UBYTES  EQU     5FDFH   ;USER SET # OF BYTES TO BE READ IN FROM TAPE
5FE1            ALTSET  EQU     5FE1H   ;ADR CHECKED FOR ALT CHAR SET USAGE
5FE3            ALTSIZ  EQU     5FE3H   ;ADR CHECKED FOR ALT CHAR SIZE USAGE
5FE5            IMGADR  EQU     5FE5H   ;ADR ROM STORES SCREEN IMAGE ADR
5FE7            SCNSTR  EQU     5FE7H   ;ADR ROM STORES ADR OF TOP OF SCREEN
5FE9            HEIGHT  EQU     5FE9H   ;ADR ROM STORES HEIGHT OF OUTPUT IMAGES
5FEA            WIDTH   EQU     5FEAH   ;ADR ROM STORES WIDTH OF OUTPUT IMAGES
5FEB            PCHLAD  EQU     5FEBH   ;ROM ROUTINE LOADS HL FROM HERE BEFORE PCHL
5FED            ALTCLR  EQU     5FEDH   ;STORAGE OF ALTERNATE OUTPUT COLOR MASK
5FEE            CLRMSK  EQU     5FEEH   ;STORAGE OF OUTPUT TEXT COLOR MASK
5FEF            RTCLCK  EQU     5FEFH   ;ADR OF REAL TIME CLOCK UPDATED BY RST 7
5FF1            LHJOY   EQU     5FF1H   ;ADR OF LEFT HAND JOYSTICK DATA
5FF2            RHJOY   EQU     5FF2H   ;ADR OF RIGHT HAND JOYSTICK DATA
5FF3            USRRST  EQU     5FF3H   ;ADR OF STORAGE OF ADR FOR USER RESTART CODE
5FF5            A2DFLG  EQU     5FF5H   ;FLAG ADR FOR TURNING ON/OFF A/D PROCESSING
5FF6            A2DCHO  EQU     5FF6H   ;A/D CHANNEL 0 DATA, NEXT 7 LOCS ARE OTHER CH'S
5FF7            LFIRE   EQU     5FF7H   ;A/D CHANNEL 1 DATA, LEFT HAND FIRE BUTTON
5FF8            LPOT    EQU     5FF8H   ;A/D CHANNEL 2 DATA, LEFT HAND POT KNOB
5FFA            RFIRE   EQU     5FFAH   ;A/D CHANNEL 4 DATA, RIGHT HAND FIRE BUTTON
5FFB            RPOT    EQU     5FFBH   ;A/D CHANNEL 5 DATA, RIGHT HAND POT KNOB
5FFF            PRTFLG  EQU     5FFFH   ;IOS VARIABLE- FLAG FOR USER ROUTINE IN OUTCHR
7FFF            ENDRAM  EQU     7FFFH   ;ADR OF LAST BYTE OF RAM IN 16K INTERACTS
                        ;**********************************************
```

```
                        ;#########################################################################
0000  3E40      RESTO:  MVI A,  40H     ;BINARY 0100 0000, SET BIT 6 IN THE A REG
0002  32022B            STA     SNDABL  ;SET THE SOUND SYSTEM ENABLE/INHIBIT BIT
0005  C30C00            JMP     SKIP1   ;SKIP OVER RESTART 1
                        ;#########################################################################
0008  F3        REST1:  DI              ;RESTART 1 (U/S BREAKPOINT CMD)
0009  C3040F            JMP     AFBKPT  ;JUMP TO 5TH BYTE OF SECOND ROM
                        ;#########################################################################
000C  3E38      SKIP1:  MVI A,  38H     ;BINARY 0011 1000 (VREF)
000E  320030            STA     CLKAD   ;MISCELLANEOUS OUTPUT REGISTER
0011  AF                XRA A           ;A=0
0012  320010            STA     CLRGA   ;(COLOR REGISTERS TO ZERO
0015  32001E            STA     CLRGB   ;(
0018  063E              MVI B,  3EH     ;COUNTER
001A  21C05F            LXI H,  STKTOP  ;ROM STACK BOTTOM IN HL
001D  F9                SPHL            ;INIT STACK POINTER WITH HL
001E  77        VARCLR: MOV M,A         ;CLEAR STORAGE AREA LOOP )
001F  23                INX H           ;                        )
0020  05                DCR B           ;                        )
0021  C21E00            JNZ     VARCLR  ;                        )
0024  3A0008            LDA     ROM2    ;GET 1ST BYTE OF SECOND ROM
0027  B7                ORA A           ;CHECK IT
0028  CA0008            JZ      ROM2    ;IF BYTE WAS ZERO, JUMP THERE
002B  3A0030    BAKJMP: LDA     CLKAD   ;GRAB CURRENT VALUE OF I/O REGISTER
002E  77                MOV M,A         ;STORE IT @ THE LOC @ THE TOP OF THE STACK
002F  3E80              MVI A,  80H     ;A=80H
0031  32D85F            STA     CLRS02  ;SET TAPE OUTPUT PHASE BIT
0034  C38C01            JMP     SKIP7   ;JUMP OVER RESTART 7 CODE
                        ;#########################################################################
```

```
                        ;/##########################################################################
0037  FF                RST 7           ;THIS CODE IS EXECUTED 60 TIMES PER SECOND
                        ;               ;BY HARDWARE INTERRUPTS. IT HANDLES ALL KEYBOARD
                        ;               ;INPUTS AND ANALOG TO DIGITAL CONVERSIONS.
                        ;               ;
0038  E5      REST7:    PUSH H          ;
0039  D5                PUSH D          ;
003A  C5                PUSH B          ;
003B  F5                PUSH PSW        ;
003C  3AF55F            LDA     A2DFLG  ;GRAB A/D PROCESSING ON/OFF FLAG
003F  B7                ORA A           ;CHECK IT
0040  C226A00           JNZ     UDRTCK  ;IF A2DFLG IS SET, DONT PROCESS A/D DATA
0043  21C05F            LXI H,  STKTOP  ;ADR OF STORAGE OF LAST I/O TO CLKAD
0046  7E                MOV A,M         ;GRAB LAST DATA TO CLKAD (YYZZZAAA)
0047  1F                RAR             ;(ROTATE A/D CHANNEL # DOWN TO THE BOTTOM
0048  1F                RAR             ;(YY = MULTIPLEXER/SELECTOR, ZZZ = A/D CHANNEL #,
0049  1F                RAR             ;(AAA = SOUNDS DATA
004A  E607              ANI     07H     ;MASK OFF LAST ACCESSED CHANNEL #
004C  F5                PUSH PSW        ;SAVE CHANNEL # ON STACK
004D  11F65F            LXI D,  A2DCHD  ;ADR OF START OF STORAGE OF A/D DATA
0050  83                ADD E           ;ADD BASE ADR TO CHANNEL # TO GET ACTUAL...
0051  5F                MOV E,A         ;...STORAGE ADR FOR THE CHANNEL'S DATA
0052  3A0030            LDA     CLKAD   ;READ NEW A/D DATA
0055  12                STAX D          ;STORE A COPY OF IT IN RAM
0056  F1                POP PSW         ;BRING BACK A/D CHANNEL #
0057  3D                DCR A           ;DECREMENT IT TO THE NEXT CHANNEL
0058  C25D00            JNZ     LCHANL  ;IF NOT ON LAST CHANNEL, JUMP
005B  3E06              MVI A,  06H     ;ELSE RESET CHANNEL # TO THE MAXIMUM
005D  87        LCHANL: ADD A           ;(SHIFT NEW CHANNEL # TO ITS' PROPER LOCATION
005E  87                ADD A           ;(
005F  87                ADD A           ;(
0060  AE                XRA M           ;)MASK NEW CHANNEL # INTO DATA FOR IOREG
0061  E6F8              ANI     0F8H    ;)
0063  AE                XRA M           ;)
0064  F640              ORI     40H     ;RESET COUNTER CMD TO MULTIPLEXER
0066  77                MOV M,A         ;STORE NEW DATA VALUE BACK AT STKTOP
0067  320030            STA     CLKAD   ;& AT THE A/D REGISTER
006A  2AEF5F    UDRTCK: LHLD    RTCLCK  ;GET LAST R/T CLOCK VALUE
006D  23                INX H           ;ADD ONE TO IT
006E  22EF5F            SHLD    RTCLCK  ;STORE NEW VALUE OF R/T CLOCK
0071  110638            LXI D,  KYBDTP  ;HIGHEST KEYBOARD REGISTER ADR
0074  21C85F            LXI H,  K1RG6   ;KEY TABLE 1 REGISTER 6 DATA ADR
0077  01CF5F            LXI B,  K2RG6   ;KEY TABLE 2 REGISTER 6 DATA ADR
007A  1A        SCANKB: LDAX D          ;GET MEMORY FROM KEYBOARD REGISTER
007B  2F                CMA             ;KEYBOARD DATA IS ONE'S COMPLEMENT
007C  A6                ANA M           ;MASK WITH KEY TABLE 1 DATA
007D  C28900            JNZ     KFOUND  ;IF KEY PRESSED, & IS SAME AS 1 PRESSED @ INTERRUPT, JUM
0080  2B        CNTSCN: DCX H           ;(DECREMENT TABLE POINTERS
0081  0B                DCX B           ;(
0082  1B                DCR E           ;DECREMENT KEYBOARD REGISTER #
0083  F27A00            JP      SCANKB  ;IF KEYBOARD REGISTER # IS <=3800, JUMP
0086  C32401            JMP     NOKEYS  ;NO VALID KEY PRESSED, SO SKIP ASCII DECODE
                        ;###############################################################################
```

```
                        ;*****************************************************************
0089  F5          KFOUND: PUSH PSW            ;SAVE KEY REG DATA
008A  0A                  LDAX B              ;GET KEY TABLE 2 DATA
008B  2F                  CMA                 ;COMPARE TABLE 2 DATA W/KEY REGISTER DATA
008C  E3                  XTHL                ;(IF THEY ARE THE SAME, JUMP (INVALID KEY)
008D  A4                  ANA H               ;(IE PRIOR DECODED KEY WHICH HAS BEEN FREED)
008E  E1                  POP H               ;(
008F  CA8000              JZ        CNTSCN    ;(
0092  110207              LXI D,    0702H     ;D=REG COUNTER, E=VALIDITY COUNT(THIS ROUTINE
0095  F5                  PUSH PSW            ;SAVE KEYBOARD REG DATA          (CHECKS THE
0096  2AC95F            LXI H,    K2RGD     ;BOTTOM OF TABLE 2 AREA          (VALIDITY OF
0099  7E          KCHK1:  MOV A,M             ;GET TABLE 2 DATA                (TABLE 2 DATA.
009A  87          KCHK2:  ADD A               ;SHIFT LEFT 1 BIT                (IF MORE THAN 1
009B  D29F00              JNC       KCHK3     ;IF NC, THEN NO BIT YET          (REG HAS DATA
009E  1D                  DCR E               ;BIT FOUND, UNBUMP COUNT         (OR IF 1 REG
009F  C2A000      KCHK3:  JNZ       KCHK2     ;DATA(OR COUNT).NE.0,JMP         (HAS MORE THAN
00A2  23                  INX H               ;BUMP TABLE 2 POINTER            (1 BIT ON, THEN
00A3  15                  DCR D               ;UNBUMP REGISTER COUNTER         (TABLE 2 DATA
00A4  C29900              JNZ       KCHK1     ;IF NOT ALL CHK.D, LOOP          (IS INVALID &
00A7  F1                  POP PSW             ;GET KEY REG DATA                (SHOULD BE
00A8  1D                  DCR E               ;CHECK VALIDITY COUNT            (IGNORED.
00A9  FA2401              JM        NOKEYS    ;IF INVALID, SKIP KEY STUFF      ;
00AC  1608                MVI D,    08H       ;COUNTER FOR MAX BITS TO ROTATE TO GET SET BIT
00AE  0F          BTCT1:  RRC                 ;ROTATE REG DATA RIGHT, BIT 0=BIT 7=CARRY FLAG
00AF  D2B500              JNC       BTCT2     ;IF BIT NOT FOUND, JUMP
00B2  5A                  MOV E,D             ;IF FOUND, SAVE BIT COUNTER TO E(D7=1,D6=2,ETC.)
00B3  3E80                MVI A,    80H       ;SET ONLY BIT 7 IN THE A REG
00B5  15          BTCT2:  DCR D               ;UNBUMP BIT COUNTER
00B6  C2AE00              JNZ       BTCT1     ;IF COUNTER NOT TO 0 YET, KEEP CHECKING
00B9  57                  MOV D,A             ;ROTATED KEY DATA SAVED TO D
00BA  0A                  LDAX B              ;GET TABLE 2 DATA
00BB  B2                  ORA D               ;PRESERVE OTHER INHIBIT BITS PRESENT
00BC  02                  STAX B              ;STORED INTO TABLE 2(STOPS DECODE ON NEXT PASS)
00BD  79                  MOV A,C             ;LOW OF TABLE 2 ADR
00BE  D6C9                SUI       0C9H      ;ADJUST TO KEY REGISTER # (3800=0,3801=1,ETC)
00C0  87                  ADD A               ;(MULTIPLY REGISTER # X 8 THEN ADD BIT POSITION
00C1  87                  ADD A               ;(...OF CHAR. (D7=1,D6=2,...)
00C2  87                  ADD A               ;(
00C3  83                  ADD E               ;(
00C4  3D                  DCR A               ;MINUS 1 = ASCII - 23H (UPPER CASE)
00C5  FE02                CPI       02H       ;WAS THE KEY HIT THE SHIFT LOCK KEY?
00C7  21D25F              LXI H,    SHIFT     ;SHIFT STATUS REGISTER ADR
00CA  C20300              JNZ       NLOCK     ;IF NOT SHIFT LOCK KEY, JUMP
00CD  7E                  MOV A,M             ;IF IT WAS, FLIP THE SHIFT STATUS REGISTER
00CE  2F                  CMA                 ;(OFFH = SHIFT LOCKED, 00H = UNLOCKED)
00CF  77                  MOV M,A             ;
00D0  C32401              JMP       NOKEYS    ;SHIFT LOCK IS PROCESSED ALONE, SKIP THE REST
                        ;*****************************************************************
```

```
                        ;//##################################################################
00D3  FE07      NLOCK:  CPI     07H     ;JUMP IF CHAR IS FROM KEY REG 3800, SPECIAL CHAR
00D5  DAE900            JC      SPCHR   ;IF NOT FROM ADR 3800, JUMP
00D8  57                MOV     D,A     ;SAVE CHAR IN D
00D9  3A0038            LDA     KYBDBT  ;GET VALUE OF 1ST KEYBOARD REGISTER
00DC  2F                CMA             ;KEYBOARD DATA IS ONE'S COMPLEMENT
00DD  E680              ANI     80H     ;GET 'SHIFT' BIT IF SET
00DF  B6                ORA     M       ;M=SHIFT STATUS. IF SHIFT STATUS SET RESTORE...
00E0  7A                MOV     A,D     ;...SAVED CHAR IN D
00E1  CAF500            JZ      NOSHFT  ;IF SHIFT KEY WAS NOT HIT, JUMP
00E4  FE17              CPI     17H     ;IS KEY .LE. '9', IF NOT, JUMP
00E6  D2F000            JNC     HIDE1   ;
00E9  217501    SPCHR:  LXI     H, CHRTBL-3 ;HL GET ADR OF SPECIAL CHAR TABLE ADR -3
00EC  85                ADD     L       ;ADD THE TABLE ADR TO THE INDEX VALUE
00ED  6F                MOV     L,A     ;RESTORE NEW ADR OF CHAR IN TABLE
00EE  7E                MOV     A,M     ;GRAB CHAR FROM CHAR TABLE
00EF  11C623            LXI     D,  23C6H ;GARBAGE INSTR TO JUMP OVER HIDDEN INSTR
                        ;-----------------------------------------------------------------
                        ;HIDE1: ADI     23H     ;HIDDEN INSTRUCTION
                        ;-----------------------------------------------------------------
00F2  C30301            JMP     NOTUC   ;CHARS IN CHRTBL ARE NOT PROCESSED AS CAPS
                        ;##################################################################
00F5  C623      NOSHFT: ADI     23H     ;CONVERT CHAR TO ASCII
00F7  FE41              CPI     41H     ;CAPITAL 'A'?
00F9  DA0301            JC      NOTUC   ;LESS THAN 'A', JUMP
00FC  FE5B              CPI     5BH     ;CAPITAL 'Z'+1?
00FE  D20301            JNC     NOTUC   ;GREATER THAN OR EQUAL TO 'Z'+1, JUMP
0101  C620              ADI     20H     ;OFFSET BETWEEN UPPER CASE & LOWER CASE
0103  57        NOTUC: MOV     D,A     ;SAVE CHAR IN D
0104  3A0038            LDA     KYBDBT  ;GET VALUE OF 1ST KEYBOARD REGISTER
0107  E6C0              ANI     0C0H    ;GET 'CONTROL' &/OR 'SHIFT' BITS
0109  EE80              XRI     80H     ;FLIP 'SHIFT' BIT
010B  7A                MOV     A,D     ;RESTORE CHAR FROM D
010C  C21601            JNZ     STRCHR  ;IF THE CONTROL BIT IS NOT SET, JUMP
010F  FE60              CPI     60H     ;ELSE CHECK FOR A VALID CONTROL CHARACTER
0111  DA1601            JC      STRCHR  ;IF CHAR IS NOT VALID, JUMP
0114  D660              SUI     60H     ;IF IT IS, THEN CONVERT IT TO IT'S PROPER VALUE
0116  32D15F    STRCHR: STA     LSTCHR  ;STORE AWAY LAST CHAR HIT IN RAM
0119  21D05F            LXI     H,  KYSTAT ;HL = ADR OF KEY HIT OR NOT STATUS
011C  7E                MOV     A,M     ;GRAB STATUS
011D  B7                ORA     A       ;CHECK IT
011E  CA2301            JZ      NOSET   ;IF ZERO, DON'T SET BIT 7 OF MEMORY LOCATION
0121  3680              MVI     M,  80H ;ELSE SET BIT 7 OF THE KEY STATUS LOCATION
0123  34                INR     M       ;BUMP STATUS FLAG IN MEMORY
0124  010038  NOSET:  LXI     B,  KYBDBT ;ADR OF LOWEST #'ED KEYBOARD REGISTER
0127  11C25F    NOKEYS: LXI     D,  K1RGO ;KEY TABLE 1 REG 0 STORAGE ADR
012A  21C95F            LXI     H,  K2RGO ;KEY TABLE 2 REG 0 STORAGE ADR
012D  0A                LDAX    B       ;GET CHAR FROM 1ST KEYBOARD REGISTER (3800)
012E  F6C0              ORI     0C0H    ;IGNORE SHIFT & CONTROL BITS (HANDLED BEFORE)
0130  FE0A              CPI     0AH     ;GARBAGE INSTRUCTION TO SKIP HIDDEN INSTRUCTION
                        ;-----------------------------------------------------------------
                        ;HIDE2: LDAX    B       ;HIDDEN INSTRUCTION-GETS CHAR FROM KEY REGS 3801+
                        ;-----------------------------------------------------------------
```

```
                        ;/
0132  2F                CMA                 ;KEYBOARD DATA IS ONE'S COMPLEMENT
0133  12                STAX D              ;STORE IN TABLE 1 LOCATION
0134  A6                ANA M               ;MASK WITH KEY TABLE 2 DATA
0135  77                MOV M,A             ;STORE NEW TABLE 2 DATA
0136  03                INX B               ;INCREMENT KEYBOARD REGISTER POINTER
0137  13                INX D               ;INCREMENT KEY TABLE 1 POINTER
0138  23                INX H               ;
0139  3E07              MVI A,   07H        ;CHECK TO SEE IF WHOLE KEYBOARD AREA...
013B  B9                CMP C               ;HAS BEEN SCANNED YET, IF NOT, JUMP BACK
013C  C23101            JNZ      HIDE2      ;JUMPS TO HIDDEN INSTRUCTION
013F  215F15            LXI H,   JOYVAL     ;POINT @ LOC FOR STORAGE OF BOTH JOYSTICK VALUES
0142  46                MOV B,M             ;GRAB LAST PASS DATA
0143  3A0738            LDA      JOYSTK     ;CHECK THE JOYSTICK REGISTER ON THIS PASS
0146  2F                CMA                 ;JOYSTICK DATA IS ONE'S COMPLEMENT
0147  77                MOV M,A             ;STORE THE DATA FOR THIS PASS
0148  A0                ANA B               ;MASK THE 2 BYTES TOGETHER
0149  47                MOV B,A             ;SAVE JOYSTICK DATA
014A  1F                RAR                 ;(SHIFT RIGHT JOYSTICK DATA TO BOTTOM OF REGISTER
014B  1F                RAR                 ;(
014C  1F                RAR                 ;(
014D  1F                RAR                 ;(
014E  E60F              ANI      0FH        ;MASK OFF BOTTOM FOUR BITS (RIGHT JOYSTICK)
0150  215F15            LXI H,   RHJOY      ;HL GET STORAGE ADR FOR RIGHT JOYSTICK
0153  77                MOV M,A             ;MOVE RIGHT JOYSTICK DATA TO CORE
0154  2B                DCX H               ;POINT HL AT LEFT JOYSTICK STORAGE ADR
0155  78                MOV A,B             ;RESTORE JOYSTICK DATA
0156  E60F              ANI      0FH        ;MASK OFF LEFT JOYSTICK DATA
0158  77                MOV M,A             ;MOVE LEFT JOYSTICK DATA TO CORE
0159  3AF55F            LDA      A2DFLG     ;GRAB A/D PROCESSING ON/OFF FLAG
015C  B7                ORA A               ;CHECK IT
015D  C26A01            JNZ      NOA2D      ;IF FLAG IS SET, DON'T PROCESS A/D DATA
0160  215C5F            LXI H,   STKTOP     ;ADR OF STORAGE OF LAST A/D DATA
0163  7E                MOV A,M             ;GRAB A/D DATA VALUE
0164  EFCO              XRI      0C0H       ;TOGGLE TOP 2 BITS FROM RESET 2 START A/D CONV
0166  77                MOV M,A             ;RESTORE DATA INTO STORAGE LOCATION...
0167  320030            STA      CLKAD      ;...& AT A/D REGISTER
016A  2AF35F  NOA2D:     LHLD     USRRST     ;GET USER'S RESTART ADR IF THERE IS ONE
016D  7D                MOV A,L             ;)-CHECK IT FOR ZERO
016E  B4                ORA H               ;)/
016F  E5                PUSH H              ;PUT ADR ON STACK
0170  C0                RNZ                 ;SNEAKY WAY TO GO TO USER'S RESTART CODE
0171  E1                POP H               ;IF NO RESTART CODE, DESTROY ZEROS ON STACK
0172  F1  ENDRST:      POP PSW             ;ENTRY POINT TO END RESTART 7 NORMALLY
0173  C1                POP B               ;
0174  D1                POP D               ;
0175  E1                POP H               ;
0176  FB                EI                  ;ENABLE KEYBOARD INTERRUPTS
0177  C9                RET
                        ;##########################################################################
```

```
                          ;/##########################################################################
0178   08        CHRTBL:  DB        BS         ;'BS'=BACK SPACE
0179   09                 DB        09H        ;'HT'=HORIZONTAL TAB
017A   0D                 DB        CR         ;'CR'=CARRIAGE RETURN
017B   20                 DB        20H        ;' '=SPACE
017C   2A                 DB        2AH        ;'*'=ASTERISK
017D   5E                 DB        5EH        ;'^'=EXPONENTIATION SIGN
017E   2C                 DB        2CH        ;','=COMMA
017F   5F                 DB        5FH        ;'_'=UNDERLINE
0180   2E                 DB        2EH        ;'.'=PERIOD
0181   2F                 DB        2FH        ;'/'=FRONT SLASH
0182   3C                 DB        3CH        ;'<'=LESS THAN
0183   3E                 DB        3EH        ;'>'=GREATER THAN
0184   22                 DB        22H        ;' '=DOUBLE QUOTE
0185   27                 DB        27H        ;'''=SINGLE QUOTE
0186   24                 DB        24H        ;'$'=DOLLAR SIGN
0187   25                 DB        25H        ;'%'=PERCENT SIGN
0188   21                 DB        21H        ;'!'=EXCLAMATION POINT
0189   3A                 DB        3AH        ;':'=COLON
018A   28                 DB        28H        ;'('=OPEN PARENTHESIS
018B   29                 DB        29H        ;')'=CLOSE PARENTHESIS
                          ;##########################################################################
```

```
                        ;/##########################################################################
018C  FB        SKIP?:  EI              ;ENABLE KEYBOARD INTERRUPTS
018D  CD730C            CALL    CLRSCR  ;CLEAR SCREEN
0190  01E501            LXI B,  RMSG1   ;ADR OF 'DEPRESS', 'L TO LOAD'
0193  CD0301            CALL    OUTPUT  ;OUTPUT MESSAGE TO SCREEN
0196  3A004C  STRTCK:  LDA     GAMEST  ;PICK UP FIRST BYTE THAT IS ALWAYS CHECKED
0199  FE31              CPI     31H     ;IS IT A 'LXI SP' INSTRUCTION?
019B  F5                PUSH PSW        ;SAVE BYTE & STATUS ON STACK
019C  23                INX B           ;POINTS BC PAIR TO RMSG2
019D  010602            LXI B,  RMSG2   ;ADR OF 'R TO RESTART' (WASTED INSTRUCTION!!!)
01A0  CC0301            CZ      OUTPUT  ;OUTPUT MESSAGE TO SCREEN IF 4C00 = 'LXI SP'
01A3  011702            LXI B,  CTAB1   ;ADR OF COLOR TABLE
01A6  CD3606            CALL    SYSCLR  ;SET SYSTEM COLORS
01A9  CDE007            CALL    KYWAIT  ;WAIT FOR KEY INPUT
01AC  47                MOV B,A         ;SAVE CHAR HIT IN B REGISTER
01AD  F1                POP PSW         ;RESTORE STATUS OF 4C00
01AE  78                MOV A,B         ;RESTORE CHAR HIT
01AF  C2E701            JNZ     NOLXI   ;IF GAMEST DOESN'T HAVE A LXI SP, JUMP
01B2  FE72              CPI     'R'+20H ;CHECK FOR KEY R PRESSED FOR RESTART OPTION
01B4  CA004C            JZ      GAMEST  ;R WAS PRESSED, JUMP TO START OF GAME
01B7  FE6C      NOLXI:  CPI     'L'+20H ;CHECK FOR KEY L PRESSED FOR LOAD TAPE OPTION
01B9  C29601            JNZ     STRTCK  ;IF NEITHER OPTION SELECTED, JUMP BACK
01BC  0601              MVI B,  01H     ;INITIALIZATION FOR READ TAPE ROUTINE
01BE  0E01              MVI C,  01H     ;(
01C0  110000            LXI D,  0000H   ;(
01C3  CD1C02            CALL    READER  ;TAPE READER
01C6  B7                ORA A           ;CHECK READ TAPE STATUS PARAMETER
01C7  C20000            JNZ     RESTO   ;IF A<>0, RESTART
01CA  3A004C            LDA     GAMEST  ;GRAB FIRST BYTE AGAIN TO SEE IF TAPE PUT...
01CD  FE31              CPI     31H     ;...S A 'LXI SP' INSTRUCTION THERE.
01CF  CA004C            JZ      GAMEST  ;YES, JUMP THERE
01D2  C7                RST 0           ;ADR FOR RESTART
                        ;/##########################################################################
01D3  0A        OUTPUT: LDAX B          ;GET TEXT COLOR OR END FLAG
01D4  FEFF              CPI     0FFH    ;END OF MESSAGE YET?
01D6  C8                RZ              ;YES, RETURN
01D7  C02F05            CALL    ACOLOR  ;SET OUTPUT COLOR
01DA  03                INX B           ;INCREMENT ADR
01DB  0A                LDAX B          ;GET Y COORDINATE
01DC  5F                MOV E,A         ;INTO E REGISTER
01DD  03                INX B           ;INCREMENT ADR
01DE  0A                LDAX B          ;GET X COORDINATE
01DF  57                MOV D,A         ;INTO D REGISTER
01E0  03                INX B           ;INCREMENT ADR
01E1  CD4F05            CALL    OUTMSG  ;OUTPUT MESSAGE TO SCREEN (TO FIRST ZERO BYTE)
01E4  03                INX B           ;INCREMENT ADR
01E5  C30301            JMP     OUTPUT  ;CYCLE BACK TO SEE IF MESSAGE CONTINUES
                        ;/##########################################################################
```

```
                        ;#########################################################################
01E8  031223    RMSG1:  DB   03H,12H,23H ;TEXT COLOR, Y COORDINATE, X COORDINATE
01EB  44455052          DB   'DEPRESS',0 ;TEXT OF MESSAGE, END OF LINE FLAG
01EF  45535300
01F3  033A14            DB   03H,3AH,14H ;TEXT COLOR, Y COORDINATE, X COORDINATE
01F6  4C20544F          DB   'L TO LOAD TAPE',0,0FFH ;TEXT, END OF LINE, & END OF MSG FLAGS
01FA  204C4F41
01FE  44205441
0202  504500FF
0206  033A14    RMSG2:  DB   03H,3AH,14H ;TEXT COLOR, Y COORDINATE, X COORDINATE
0209  5220544F          DB   'R TO RESTART',0,0FFH ;TEXT, END OF LINE & END OF MSG FLAGS
020D  20524553
0211  5441 5254
0215  00FF
                        ;#########################################################################
0217  04        CTAB1:  DB   04H     ;BACKGROUND COLOR (4=BLUE)
0218  01                DB   01H     ;COLOR 1 (1=RED)
0219  02                DB   02H     ;COLOR 2 (2=GREEN)
021A  07                DB   07H     ;COLOR 3 (7=WHITE)
021B  00                DB   00H     ;END OF COLOR TABLE FLAG
                        ;#########################################################################
```

```
                        ;/##########################################################################
021C  F3        READER: DI              ;DISABLE INTERRUPTS FOR READING TAPES
021D  AF                XRA A           ;A=0
021E  32D35F            STA     RDSTAT  ;INITIALIZE FOR A GOOD READING STATUS
0221  D5                PUSH D          ;(SAVE USER PASSED INFO. DE = ZEROS,...
0222  C5                PUSH B          ;(...BC = 0101 (USUALLY)
0223  CDCA02            CALL    TAPSND  ;ACTIVATE TAPE READING SOUNDS?
0226  CDDD02            CALL    TAPEON  ;TURN ON THE TAPE MOTOR
0229  011D00            LXI B,  001DH   ;TIME DELAY FOR MOTOR TO SPIN UP
022C  C0F607            CALL    DELAY   ;WAIT LOOP
022F  016401            LXI B,  0164H   ;SET MINIMUM VALUE FOR VALID TAPE LEADER
0232  1E00              MVI E,  00H     ;SET PARAMETER FOR LEADRD ROUTINE
0234  CDB103            CALL    LEADRD  ;WAIT FOR A VALID LENGTH TAPE LEADER
0237  C1                POP B           ;RESTORE PASSED BC VALUE
0238  C5          READ1: PUSH B          ;RESAVE PASSED BC VALUE
0239  010500            LXI B,  0005H   ;# OF BYTES IN TAPE HEADER TO READ INTO CORE
023C  11D45F            LXI D,  TPSTRT  ;DE GET ADR FOR STORAGE OF TAPE HEADER INFO
023F  CD1A03            CALL    READIN  ;READ HEADER BLOCK IN FROM TAPE
0242  C1                POP B           ;RESTORE PASSED BC VALUE
0243  B7                ORA A           ;CHECK STATUS OF HEADER BLOCK READ
0244  CA5D02            JZ      OKHDR   ;IF STATUS WAS GOOD, JUMP
0247  32D35F            STA     RDSTAT  ;IF BAD HEADER READ STATUS, STORE IT AWAY
024A  78                MOV A,B         ;PUT PASSED B VALUE IN THE ACCUMULATOR
024B  B7                ORA A           ;CHECK IT'S VALUE
024C  C23802            JNZ     READ1   ;IF NON-ZERO, GO READ IN ANOTHER HEADER BLOCK
024F  D1          STOPRD: POP D          ;RESTORE PASSED DE VALUE
0250  7A                MOV A,D         ;MOVE D VALUE INTO THE ACCUMULATOR
0251  B7                ORA A           ;CHECK IT'S VALUE
0252  C25802            JNZ     NOTOFF  ;IF NON-ZERO, THEN DO NOT TURN OFF TAPE MOTOR
0255  CDE302            CALL    TAPOFF  ;TURN OFF THE TAPE MOTOR
0258  3AD35F      NOTOFF: LDA     RDSTAT  ;GRAB FINAL TAPE READING STATUS FLAG
025B  FB                EI              ;ENABLE KEYBOARD INTERRUPTS
025C  C9                RET
                        ;##########################################################################
025D  C5          OKHDR: PUSH B         ;SAVE ORIGINAL PASSED BC VALUE
025E  2AD05F            LHLD    USTART  ;GET USER ALTERNATE LOAD ADR
0261  7C                MOV A,H         ;(CHECK IT
0262  B5                ORA L           ;(
0263  CA6F02            JZ      NOALT   ;IF ZERO, THEN NO ALTERNATE LOAD POINT WAS SET
0266  22D45F            SHLD    TPSTRT  ;STORE # TAPE DATA STARTING ADR LOCATION
0269  2AD25F            LHLD    UBYTES  ;GET USER SET # OF BYTES TO BE LOADED IN FROM TAPE
026C  22D65F            SHLD    TPBYTS  ;STORE AWAY # OF BYTES INCOMING FROM TAPE LOC
026F  3AD85F      NOALT: LDA    HDRFLG  ;GRAB TAPE RECORD DATA TYPE IDENTIFIER FLAG
0272  FEFD              CPI     0FDH    ;CHECK FOR END-OF-FILE FLAG
0274  C27B02            JNZ     CKBULK  ;IF NOT END-OF-FILE FLAG, THEN JUMP
0277  C1                POP B           ;RESTORE PASSED BC VALUE
0278  C34F02            JMP     STOPRD  ;GO TO EXIT THE TAPE READING CODE
                        ;##########################################################################
```

```
                        ;/##################################################################
027B  FEFE      CKBULK: CPI     0FEH    ;CHECK FOR BULK-MEMORY-DATA FILL FLAG
027D  C2AE02            JNZ     FFREC   ;IF IT'S NOT BULK-MEMORY-DATA, IT'S RAW DATA
0280  21D95F            LXI H,  BULKFL  ;ADR OF BULK-MEMORY-DATA FILLER BYTE
0283  010100            LXI B,  0001H   ;# OF BYTES TO READ IN FROM TAPE
0286  CD1A03            CALL    READIN  ;READ IN BULK-MEMORY-DATA BYTE
0289  C1              POP B           ;RESTORE PASSED BC VALUE
028A  B7              ORA A           ;CHECK BULK-MEMORY DATA BYTE READ STATUS
028B  CA9902            JZ      BULKIN  ;IF STATUS IS OK, GO TO INSERT BULK DATA
028E  32D35F            STA     RDSTAT  ;ELSE STORE BAD STATUS
0291  78              MOV A,B         ;PUT PASSED B VALUE INTO THE ACCUMULATOR
0292  B7              ORA A           ;CHECK IT'S VALUE
0293  C23802            JNZ     READ1   ;IF NON-ZERO, GO READ IN A NEW HEADER BLOCK
0296  C34F02            JMP     STOPRD  ;ELSE, GO EXIT THE TAPE READING CODE
                        ;##################################################################
0299  2AD65F    BULKIN: LHLD    TPBYTS  ;GET THE # OF BYTES INCOMING FROM TAPE
029C  EB                XCHG            ;# OF INCOMING BYTES TO DE
029D  2AD45F            LHLD    TPSTRT  ;LOAD FROM TAPE DATA STARTING ADR LOCATION
02A0  3AD95F    BULK1:  LDA     BULKFL  ;GET BULK-MEMORY-DATA BYTE
02A3  77                MOV M,A         ;STORE BYTE IN CORE
02A4  23                INX H           ;BUMP STORAGE ADR
02A5  1B                DCX D           ;UNBUMP BYTE COUNTER
02A6  7A                MOV A,D         ;CCHECK BYTE COUNTER VALUE
02A7  B3                ORA E           ;C
02A8  C2A002            JNZ     BULK1   ;IF COUNTER IS NOT ZERO, KEEP FILLING MEMORY
02AB  C33802            JMP     READ1   ;ELSE GO READ IN A NEW HEADER BLOCK
                        ;##################################################################
02AE  2AD45F    FFREC:  LHLD    TPSTRT  ;LOAD FROM TAPE DATA STARTING ADR LOCATION
02B1  EB                XCHG            ;START ADR TO DE
02B2  2AD65F            LHLD    TPBYTS  ;GET THE # OF BYTES INCOMING FROM TAPE
02B5  44                MOV B,H         ;(# OF BYTES TO READ TO BC
02B6  4D                MOV C,L         ;C
02B7  CD0003            CALL    RDBLOK  ;READ IN THE DATA BLOCK FROM TAPE
02BA  C1              POP B           ;RESTORE PASSED BC VALUE
02BB  B7              ORA A           ;CHECK RAW DATA BLOCK READ STATUS
02BC  CA3802            JZ      READ1   ;IF STATUS IS OK, GO READ IN A NEW HEADER BLOCK
02BF  32D35F            STA     RDSTAT  ;ELSE STORE BAD READ STATUS
02C2  78              MOV A,B         ;PUT PASSED B VALUE INTO THE ACCUMULATOR
02C3  B7              ORA A           ;CHECK IT'S VALUE
02C4  C23802            JNZ     READ1   ;IF NON-ZERO, GO READ IN A NEW HEADER BLOCK
02C7  C34F02            JMP     STOPRD  ;ELSE GO EXIT THE TAPE READING CODE
                        ;##################################################################
02CA  0F        TAPSND: RRC             ;
02CB  2F                CMA             ;
02CC  E640              ANI     40H     ;
02CE  320228            STA     SNDABL  ;SOUND SYSTEM ENABLE/INHIBIT BIT
0201  3AC05F            LDA     STKTOP  ;GRAB LAST DATA TO CLKAD
0204  F607              ORI     07H     ;TAPE SOUNDS TO AUDIO
0206  32C05F            STA     STKTOP  ;STORE COPY OF DATA TO CLKAD AWAY
0209  3C0030            STA     CLKAD   ;OPEN AUDIO CHANNEL
02DC  C9                RET             ;
                        ;##################################################################
```

```
                        ;##########################################################################
02DD  F5        TAPEON: PUSH PSW        ;ENTRY TO TURN ON TAPE MOTOR
02DE  3EFF              MVI A,  0FFH    ;
02E0  C3E502            JMP     MTRCHG  ;
                        ;##########################################################################
02E3  F5        TAPOFF: PUSH PSW        ;ENTRY TO TURN OFF TAPE MOTOR
02E4  AF                XRA A           ;A=0
                        ;--------------------------------------------------------------------------
02E5  E5        MTRCHG: PUSH H          ;
02E6  21DB5F            LXI H,  CLRS02  ;
02E9  AE                XRA M           ;
02EA  E640              ANI     40H     ;
02EC  AE                XRA M           ;
02ED  77                MOV M,A         ;
02EE  320010            STA     CLRGA   ;
02F1  21C05F            LXI H,  STKTOP  ;
02F4  7E                MOV A,M         ;
02F5  E607              ANI     07H     ;
02F7  F638              ORI     38H     ;
02F9  77                MOV M,A         ;
02FA  320030            STA     CLKAD   ;
02FD  E1                POP H           ;
02FE  F1                POP PSW         ;
02FF  C9                RET             ;
                        ;##########################################################################
0300  E5        RDBLOK: PUSH H          ;SAVE USED REGISTER SET
0301  CDDD02            CALL    TAPEON  ;TURN ON THE TAPE MOTOR
0304  AF                XRA A           ;A=0
0305  32DB5F            STA     RDSTAT  ;CLEAR READ TAPE STATUS FLAG
0308  CD1A03    RDLOOP: CALL    READIN  ;READ IN ONE RECORD FROM TAPE
030B  21DB5F            LXI H,  RDSTAT  ;HL GET ADR OF KEY HIT OR NOT STATUS ADR
030E  B6                ORA M           ;COMBINE RECORD READ STATUS WITH PRIOR STATUS
030F  77                MOV M,A         ;& STORE AWAY NEW STATUS VALUE
0310  78                MOV A,B         ;HIGH BYTE OF # OF BYTES TO READ IN A
0311  B1                ORA C           ;CHECK AGAINST LOW BYTE
0312  C20803            JNZ     RDLOOP  ;IF MORE BYTES TO READ, LOOP BACK
0315  E1                POP H           ;RESTORE USED REGISTER SET
0316  3ADB5F            LDA     RDSTAT  ;GRAB TAPE READING STATUS FLAG
0319  C9                RET             ;
                        ;##########################################################################
```

```
                        ;/##########################################################################
031A  E5        READIN: PUSH H          ;BASED:DE=ADR TO START LOAD @, HL=???
031B  C5                PUSH B          ;SAVE # OF BYTES TO READ FROM TAPE
031C  CD4E03            CALL    RECNT   ;READ TAPE RECORD BYTE COUNTER
031F  C1                POP B           ;RESTORE # OF BYTES TO READ IN
0320  67                MOV H,A         ;RECORD BYTE COUNT TO H (OR ERROR CODE IF BAD)
0321  D23603            JNC     VLDCNT  ;IF NO ERROR FOUND IN BYTE COUNT READ, JUMP
0324  6F        RERROR: MOV L,A         ;READ ERROR ENTRY POINT, ERROR CODE TO L
0325  7B                MOV A,E         ;LOW BYTE OF STORAGE ADR TO A
0326  84                ADD H           ;ADD ERROR CODE TO LOW BYTE OF STORAGE ADR
0327  5F                MOV E,A         ;OFFSET STORAGE ADR RESTORED TO DE
0328  7A                MOV A,D         ;(IF PRIOR ADDITION GENERATED A CARRY, THEN...
0329  CE00              ACI     00H     ;(...ADJUST THE HIGH BYTE OF THE STORAGE ADR...
032B  57                MOV D,A         ;(...ACCORDINGLY
032C  79                MOV A,C         ;LOW BYTE OF # OF TAPE BYTES TO A
032D  94                SUB H           ;DEDUCT ERROR CODE FROM LOW BYTE OF # OF BYTES
032E  4F                MOV C,A         ;RESTORE ADJUSTED LOW BYTE COUNTER
032F  7A                MOV A,D         ;(IF PRIOR SUBTRACTION GAVE A CARRY, THEN...?
0330  DE00              SBI     00H     ;(...ADJUST THE HIGH BYTE OF THE BYTE...
0332  47                MOV B,A         ;(...COUNTER ACCORDINGLY
0333  7D                MOV A,L         ;RESTORE ERROR CODE TO THE ACCUMULATOR
0334  E1                POP H           ;RESTORE USED REGISTER PAIR
0335  C9                RET             ;
                        ;##########################################################################
0336  CD9A03    VLDCNT: CALL    BLDBYT  ;READ & BUILD A MEMORY BYTE
0339  DA2403            JC      RERROR  ;IF CARRY COMES BACK, THEN ERROR OCCURRED
033C  12                STAX D          ;STORE READ BYTE INTO CORE
033D  13                INX D           ;BUMP STORAGE ADR
033E  0B                DCX B           ;DECREMENT TOTAL BYTES TO READ COUNTER
033F  25                DCR H           ;DECREMENT RECORD BYTE COUNTER
0340  CA4C03            JZ      RECDON  ;IF RECORD ALL READ, JUMP
0343  78                MOV A,B         ;(ELSE, CHECK TOTAL # OF BYTES TO READ VALUE
0344  B1                ORA C           ;(
0345  C23603            JNZ     VLDCNT  ;IF NOT ZERO, GO GET REST OF BYTES FROM TAPE
0348  3E04              MVI A,  04H     ;ELSE SET ERROR CODE = 4
034A  E1                POP H           ;RESTORE USED REGISTER PAIR
034B  C9                RET             ;
                        ;##########################################################################
034C  E1        RECDON: POP H           ;RESTORE USED REGISTER PAIR
034D  AF                XRA A           ;A=0, VALID RETURN FOR A GOOD READ (Z,NC FLAGS)
034E  C9                RET             ;
                        ;##########################################################################
```

```
                        ;/##########################################################################
035E  CDD003    RECNT:  CALL    RD1BIT  ;GET BIT (BYTE) FROM TAPE
0361  FE15             CPI     15H     ;CHECK FOR VALUES .GT. CUTOFF VALUE FOR 'ONE'
0363  DA4F03           JC      RECNT   ;IF .LT. 'ONE', GO BACK FOR GOOD VALUE
0366  CDD003  NOLDR:  CALL    RD1BIT  ;GET BIT (BYTE) FROM TAPE
0369  FE21             CPI     21H     ;CHECK FOR VALUES .GT. CUTOFF VALUE FOR LEADER
036B  D24F03           JNC     RECNT   ;IF .GT. LEADER VALUE, GO BACK FOR GOOD VALUE
036E  FE15             CPI     15H     ;CHECK FOR VALUES .GT. CUTOFF VALUE FOR 'ONE'
0370  D25703           JNC     NOLDR   ;IF .GT. VALUE OF 'ONE', JUMP BACK
0373  C5              PUSH B          ;# OF BYTES TO READ INTO CORE ON STACK
0374  FE0D            CPI     0DH     ;CHECK BOUNDARY BETWEEN 'ZERO' & 'ONE'
0376  3F              CMC             ;NOW, 'ZERO' HAS NO CARRY 'ONE' HAS CARRY
0377  0E80            MVI C,  80H     ;SET UP FLAG BIT FOR BYTE READ & PACK ROUTINE
0379  DA8303           JC      BLD1    ;IF BYTE INPUT = 1, JUMP (WILL HIT RET & EXIT)
037C  CDD003    BLD2:   CALL    RD1BIT  ;
037F  FE0D              CPI     0DH     ;
0381  D29503            JNC     HIDE3   ;
0384  CDD003            CALL    RD1BIT  ;
0387  FE21              CPI     21H     ;
0389  D29503            JNC     HIDE3   ;
038C  FE15              CPI     15H     ;
038E  DA9503            JC      HIDE3   ;
0391  00                NOP             ;
0392  3E01             MVI A,  01H     ;ERROR CODE, A = 1
0394  013E02           LXI B,  023EH   ;GARBAGE INSTRUCTION TO HIDE HIDDEN INSTRUCTION
                        ;-------------------------------------------------------------------------
                        ;HIDE3: MVI A,  02H     ;HIDDEN INSTRUCTION, ERROR CODE, A = 2
0397  C1                POP B           ;RESTORE # OF BYTES TO READ INTO CORE
0398  37                STC             ;CARRY = 1 FOR AN ERROR RETURN
0399  C9                RET
                        ;/##########################################################################
039A  C5        BLDBYT: PUSH B          ;SAVE USED REGISTER SET (???)
039B  0E80              MVI C,  80H     ;SET UP FLAG BIT FOR BYTE READ & PACK ROUTINE
039D  CDD003    BLD1:   CALL    RD1BIT  ;GET BIT (AS BYTE) FROM TAPE
03A0  FE15              CPI     15H     ;CHECK FOR VALUES .GT. 'ONE' RANGE
03A2  D29503            JNC     HIDE3   ;IF .GT. 'ONE', ERROR FOUND
03A5  FE0D      BLD2:   CPI     0DH     ;CHECK CUTOFF VALUE BETWEEN 'ZERO' & 'ONE'
03A7  3F                CMC             ;NOW 'ZERO' HAS NO CARRY, 'ONE' HAS CARRY
03A8  79        BLD3:   MOV A,C         ;BRING IN FLAG BYTE HOLDING REGISTER
03A9  1F                RAR             ;WHEN LOW BIT OF C IS SET, SET CARRY & EXIT
03AA  4F                MOV C,A         ;SAVE FLAG BYTE HOLDING REGISTER
03AB  D29D03            JNC     BLD1    ;IF FLAG BIT IN C HASN'T REACHED THE CARRY, JUMP
03AE  B7                ORA A           ;CLEAR CARRY FOR A VALID READ (A = TAPE BYTE)
03AF  C1                POP B           ;RESTORE USED REGISTER SET (???)
03B0  C9                RET
                        ;/##########################################################################
```

```
                        ;##########################################################################
03B1  C5        LEADR:  PUSH B          ;SAVE MINIMUM VALID LEADER BYTE COUNT
03B2  CD0003    NXTLDR: CALL    RD1BIT  ;PICK UP BYTE FROM TAPE
03B5  FE21              CPI     21H     ;COMPARE W/MAX LEADER BYTE VALUE
03B7  D2CC03            JNC     NOGOOD  ;IF BYTE .GE. MAX, DON'T USE IT
03BA  FE15              CPI     15H     ;COMPARE W/MIN LEADER BYTE VALUE
03BC  DACC03            JC      NOGOOD  ;IF BYTE .LT. MIN, DON'T USE IT
03BF  0B                DCX B           ;GOT VALID BYTE, TAKE 1 FROM VALID BYTE COUNTER
03C0  78                MOV A,B         ;CHECK MINIMUM VALID CONSECUTIVE LEADER...
03C1  B1                ORA C           ;...BYTE COUNTER. IF THE MINIMUM IS NOT YET...
03C2  C2B203            JNZ     NXTLDR  ;...REACHED, LOOP UNTIL IT IS
03C5  7B                MOV A,E         ;PASSED PARAMETER VALUE TO A
03C6  B7                ORA A           ;CHECK IT
03C7  C4CA02            CNZ     TAPSUB  ;IF NON-ZERO, CALL THE SUBROUTINE
03CA  C1                POP B           ;
03CB  C9                RET             ;
                        ;##########################################################################
03CC  C1        NOGOOD: POP B           ;RESTORE MINIMUM VALID LEADER BYTE COUNT
03CD  C3B103            JMP     LEADRD  ;& TRY AGAIN TO READ A VALID LENGTH LEADER
                        ;##########################################################################
03D0  E5        RD1BIT: PUSH H          ;SAVE USED REGISTER PAIR
03D1  2AD0A5            LHLD    TAPWRD  ;GRAB LAST DATA FROM TAPE INTO L
03D4  3AD030    NOCHG:  LDA     CLKAD   ;GET CURRENT DATA COUNT VALUE FROM TAPE
03D7  AD                XRA L           ;IF THE HIGH BIT OF THE 2 VALUES ARE =,...
03D8  F2D403            JP      NOCHG   ;THEN THE STATUS HERE WILL BE POSITIVE, LOOP BACK
03DB  AD                XRA L           ;RESTORE VALUE OF INPUT COUNT VALUE IN A
03DC  32D0A5            STA     TAPWRD  ;STORE THIS VALUE AS THE NEW LAST VALUE IN
03DF  95                SUB L           ;GET THE COUNT DIFFERENCE SINCE THE LAST CHANGE
03E0  E1                POP H           ;RESTORE USED REGISTER PAIR
03E1  E67F              ANI     7FH     ;MAX VALUE TO PASS BACK IN THE A REG = 07FH
03E3  C9                RET             ;
                        ;##########################################################################
```

```
                        ;##########################################################################
03E4  C5        BYT2TP: PUSH B          ;SAVE BYTE TO GO TO TAPE ON STACK (IN C)
03E5  0608              MVI B, 08H      ;LOOP COUNTER FOR # OF BIT/BYTES TO GO OUT
03E7  79        NEXTBT: MOV A,C         ;OUTGOING BYTE TO A
03E8  0F                RRC             ;PUSH A BIT OFF BOTTOM OF BYTE INTO CARRY FLAG
03E9  4F                MOV C,A         ;SAVE REMAINING BITS IN C
03EA  9F                SBB A           ;A=0 IF NO CARRY, =0FFH W/CARRY
03EB  E608              ANI 08H         ;A=0 IF BOTTOM BIT WAS 0, A=8 IF BIT WAS 1
03ED  C608              ADI 08H         ;A=8 IF BOTTOM BIT WAS 0, A=10H IF BIT WAS 1
03EF  CDF803            CALL OUTFRQ     ;PUT OUTPUT FREQUENCY ONTO TAPE
03F2  05                DCR B           ;UNBUMP BIT/BYTE COUNTER
03F3  C2E703            JNZ NEXTBT      ;IF BYTE NOT FULLY WRITTEN, CYCLE UNTIL DONE
03F6  C1                POP B           ;RESTORE BYTE THAT WENT TO TAPE
03F7  C9                RET
                        ;##########################################################################
03F8  D602      OUTFRQ: SUI 02H         ;
03FA  CDFF03            CALL OUTCG      ;CONSTANT LENGTH DATA FOR BETWEEN BITS OF DATA
03FD  3E02              MVI A, 02H      ;OUTPUT STATE CHANGE TO TAPE
03FF  E5        OUTCG:  PUSH H          ;GET LAST DATA TO TAPE COUNTER & TAPE OUTPUT DATA BIT
0400  2ADA5F            LHLD TAPWRD     ;ADD OUTPUT DURATION TO LAST COUNTER VALUE
0403  85                ADD L           ;MAX COUNTER VALUE IS 7FH
0404  E67F              ANI 7FH         ;RESTORE COUNTER TO WAIT FOR TO L
0406  6F                MOV L,A         ;GET COUNTER VALUE FROM HARDWARE
0407  3A0030  WAITCT: LDA CLKAD        ;MAX COUNTER VALUE IS 7FH
040A  E67F              ANI 7FH         ;CHECK AGAINST STOP VALUE FOR OUTPUT BYTE
040C  BD                CMP L           ;IF STOP VALUE NOT REACHED YET, WAIT FOR IT
040D  C20704            JNZ WAITCT      ;GET CLRS02 DATA INTO A
0410  7C                MOV A,H         ;FLIP TAPE OUTPUT BIT
0411  EE80              XRI 80H         ;PUT DATA TO H/W, THEREBY TO TAPE
0413  32D010            STA CLRGA       ;RESTORE H/W DATA W/TOP BIT FLIPPED TO H
0416  67                MOV H,A         ;PUT DATA FOR THIS PASS AWAY IN CORE
0417  22DA5F            SHLD TAPWRD     ;RESTORE USED REGISTER SET
041A  E1                POP H
041B  C9                RET
                        ;##########################################################################
041C  3A0030            LDA CLKAD       ;
041F  E67F              ANI 7FH         ;
0421  32DA5F            STA TAPWRD      ;
0424  78        WGAP:   MOV A,B         ;CHECK VALUE OF COUNTER # OF LEADER BYTES...
0425  B1                ORA C           ;...TO WRITE TO TAPE
0426  C8                RZ              ;IF COUNTER ZERO, RETURN
0427  3E1B              MVI A, 1BH      ;LEADER FREQUENCY OUTPUT BYTE VALUE
0429  CDF803            CALL OUTFRQ     ;PUT OUTPUT FREQUENCY ON TAPE
042C  0B                DCX B           ;UNBUMP COUNTER FOR # OF LEADER BYTES
042D  C32404            JMP WGAP        ;CYCLE BACK TO SEE IF COMPLETED
                        ;##########################################################################
0430  CD3304    ENDFIL: CALL TOGGLE     ;
0433  3E08      TOGGLE: MVI A, 08H      ;SET OUTPUT FREQ. = TO ZERO VALUE
0435  CDF803            CALL OUTFRQ     ;PUT OUTPUT FREQUENCY ON TAPE
0438  010100            LXI D, 0001H    ;COUNTER FOR # OF LEADER BYTES TO PUT ON TAPE
043B  C32404            JMP WGAP        ;GO PUT LEADER BYTE ONTO TAPE
                        ;##########################################################################
```

```
                        ;#################################################################
043E  79        CHROUT: MOV A,C         ;MOVE CHARACTER TO THE A REGISTER
043F  B7        ACHROT: ORA A           ;CHECK CHAR
0440  C8                RZ              ;IF CHAR = 0, RETURN
0441  E5                PUSH H          ;ELSE SAVE HL-UNKNOWN CONTENTS
0442  D5                PUSH D          ;SAVE PRINT POSITION (X,Y)=DE
0443  57                MOV D,A         ;PUT CHAR INTO D REGISTER
0444  2AE15F            LHLD    ALTSET  ;GET ANY USER ALTERNATE CHAR SET ADR
0447  7C                MOV A,H         ;CHECK ADR OF ALTERNATE CHAR ADR STORAGE
0448  B5                ORA L           ;
0449  C24F04            JNZ     NOMSKS  ;IF SET BY USER, DON'T USE ROM CHAR MASKS
044C  21A606            LXI H,  ASCHRS  ;ADR OF TABLE OF ASCII CHAR MASKS
044F  7A        NOMSKS: MOV A,D         ;RESTORE CHAR INTO A REGISTER
0450  FE61              CPI     61H     ;LOWER CASE ASCII 'A'?
0452  DA5704            JC      CTLCA   ;LESS THAN THAT, JUMP
0455  D643              SUI     43H     ;ELSE, LOWER CASE 'A' = 1EH
0457  C36F04            JMP     GTIMGE  ;GO GET CHAR IMAGE
                        ;#################################################################
```

```
                        ;/##########################################################################
045A  FE1E      LTLCA:  CPI     1EH             ;
045C  3D                DCR A                   ;
045D  DA6F04            JC      GTIMGE          ;
0460  D61D              SUI     1DH             ;
0462  57                MOV D,A                 ;SAVE MODIFIED CHAR VALUE IN D
0463  2AE35F            LHLD    ALTSIZ          ;??? ALTERNATE CHAR SIZE SET BY USER ???
0466  7C                MOV A,H                 ;
0467  B5                ORA L                   ;ZERO FLAG IS SET IF NO ALTERNATE CHAR SIZE SET
0468  7A                MOV A,D                 ;RESTORE MODIFIED CHAR VALUE TO A
0469  C26F04            JNZ     GTIMGE          ;IF SO, DON'T USE ROM CHAR SIZE VALUES
046C  216E06            LXI H,  CHRDEF          ;ADR OF CHAR SIZE STORAGE
046F  5E        GTIMGE: MOV E,M                 ;BYTE OF CHAR MASK OR CHAR HEIGHT IN E
0470  23                INX H                   ;
0471  56                MOV D,M                 ;NEXT BYTE OF CHAR MASK OR CHAR-WIDTH IN D
0472  23                INX H                   ;
0473  E5                PUSH H                  ;ADR OF 1ST ALT CHAR ON STACK
0474  D5                PUSH D                  ;CHAR SIZE DIMENSIONS ON STACK
0475  CD1805            CALL    EXA2HL          ;FIND E * A = HL
0478  D1                POP D                   ;RESTORE CHAR DIMENSIONS
0479  7A                MOV A,D                 ;CHAR WIDTH TO THE A REG
047A  C607              ADI     07H             ;ADD 7 TO THE WIDTH OF THE CHAR TO FIND...
047C  1F                RAR                     ;(...THE # OF BYTES WIDE THE OUTPUT IMAGE...
047D  1F                RAR                     ;(...IS SO THAT AN IMAGE MAY BE GENERATED
047E  1F                RAR                     ;(TWO BITS FOR EACH SCREEN PIXEL, 4 PIXELS/BYTE)
047F  E61F              ANI     1FH             ;A IS NOW THE # OF BYTES WIDE CHAR IMAGE IS
0481  EB                XCHG                    ;CHR SIZE DIMS TO HL, PRIOR PRODUCT TO DE
0482  E3                XTHL                    ;SWAP CHR SIZE DIMS & 1ST ALT CHAR ADR ON STACK
0483  CD1D05            CALL    DEXAPH          ;FIND (DE*A)+HL=HL
0486  22E55F            SHLD    INGADR          ;CHAR IMAGE ADR NOW KNOWN & STORED AWAY
0489  E1                POP H                   ;CHAR SIZE DIMENSIONS OFF STACK
048A  D1                POP D                   ;OUTPUT PRINT COORDINATES OFF STACK...
048B  D5                PUSH D                  ;...THEN RESAVED ON STACK
048C  C5                PUSH B                  ;
048D  010000            LXI B,  ZEROS           ;DATA FOR SCREEN OUTPUTTING ROUTINE
0490  CD9D04            CALL    DRWIMG          ;GO DRAW CHAR IMAGE ON SCREEN
0493  C1        POPPR:  POP B                   ;ENTRY POINT USED BY 2ND ROM TO RESTORE REGS
0494  D1                POP D                   ;
0495  E1                POP H                   ;
0496  C9                RET                     ;
                        ;##########################################################################
```

```
                        ;#########################################################
0497  CDE705    RPLOT:  CALL    SCRADR  ;PLOTS OUT PICTURE ON SCREEN
049A  CDB305            CALL    IMGLOC  ;GET ADR OF IMAGE & IMAGE DEFINITION
049D  E5        DRWIMG: PUSH H          ;SAVE OUTPUT IMAGE DIMENSIONS ON STACK
049E  210B04            LXI H,  JMP38
04A1  22EB5F            SHLD    PCHLAD
04A4  E1                POP H           ;RESTORE OUTPUT IMAGE DIMENSIONS
04A5  7C        OUTIMG: MOV A,H         ;(CHECK X DIMENSION
04A6  B7                ORA A           ;(
04A7  C8                RZ              ;(IF ZERO, RETURN
04A8  7D                MOV A,L         ;)CHECK Y DIMENSION
04A9  B7                ORA A           ;)
04AA  C8                RZ              ;)IF ZERO, RETURN
04AB  E5                PUSH B  EXX     ;SAVE ZEROS ON STACK
04AC  D5                PUSH D          ;OUTPUT COORDS ON STACK
04AD  E5                PUSH H          ;IMAGE DIMENSIONS ON STACK
04AE  22E95F            SHLD    HEIGHT  ;SAVE OUTPUT IMAGE DIMENSIONS IN RAM
0481  E5                PUSH H          ;IMAGE DIMENSIONS ON STACK AGAIN
04B2  C5        NXTPXL: PUSH B
04B3  D5                PUSH D
04B4  E5                PUSH H
04B5  2AEB5F            LHLD    PCHLAD
04B8  CD0A04            CALL    PCHL1
04BB  E1                POP H
04BC  D1                POP D
04BD  C1                POP B
04BE  0C                INR C
04BF  1C                INR E
04C0  2D                DCR L
04C1  C2B204            JNZ     NXTPXL
04C4  E3                XTHL            EXX
04C5  7D                MOV A,L
04C6  E3                XTHL
04C7  6F                MOV L,A
04C8  79                MOV A,C
04C9  95                SUB L
04CA  4F                MOV C,A
04CB  7B                MOV A,E
04CC  95                SUB L
04CD  5F                MOV E,A
04CE  04                INR B
04CF  14                INR D
04D0  25                DCR H
04D1  C2B204            JNZ     NXTPXL
0484  E1        POPSTC  POP H
0485  E1                POP H
0486  D1                POP D
0487  C1                POP B
0488  37                STC
04B9  C9                RET
                        ;#########################################################
04DA  E9        PCHL1:  PCHL
                        ;#########################################################
```

coord x + 1
longueur - 1 Hauteur - 1

coord y + 1
longueur - 1

```
                        ;#########################################################################
04DB CDED04     JMP38:  CALL    IMGPTR  ;FIND POINTER INTO OUTPUT IMAGE
04DE C8                 RZ              ;IF IMAGE COMPLETE, RETURN
04DF 3AEE5F     JMP39:  LDA     CLRMSK  ;GRAB CURRENT OUTPUT COLOR MASK
04E2 4F         SETD0:  MOV C,A         ;COLOR MASK TO C
04E3 CD2D05             CALL    BYTMSK  ;GET SCREEN ADR & PXLMSK FOR BYTE OUTPUT
04E6 47         SET3    MOV B,A         ;SAVE PIXEL MASK INTO B
04E7 79         SET4    MOV A,C         ;GET OUTPUT COLOR IN A
04E8 AE         SETPXL: XRA M           ;
04E9 A0                 ANA B           ;
04EA AE                 XRA M           ;
04EB 77                 MOV M,A         ;
04EC C9                 RET             ;
                        ;#########################################################################
04ED 3AEA5E     IMGPTR: LDA     WIDTH   ;GRAB IMAGE WIDTH DIMENSION
04F0 C607               ADI     07H     ;(TO FIND THE # OF WHOLE BYTES WIDE THE IMAGE
04F2 1F                 RAR             ;(...IS, ROUND UP THE TOTAL
04F3 1F                 RAR             ;(
04F4 1F                 RAR             ;(
04F5 E61F               ANI     1FH     ;(A NOW = # OF BYTES WIDE THE SCREEN IMAGE IS
04F7 D5                 PUSH D          ;SAVE OUTPUT COORDS
04F8 59                 MOV E,C         ;
04F9 1600               MVI D,  00H     ;
04FB 2AE55F             LHLD    IMGADR  ;
04FE CD1D05             CALL    DEXAPH  ;FIND (DE*A)+HL=HL
0501 78                 MOV A,B         ;
0502 1F                 RAR             ;
0503 1F                 RAR             ;
0504 1F                 RAR             ;
0505 E61F               ANI     1FH     ;
0507 5F                 MOV E,A         ;
0508 1600               MVI D,  00H     ;
050A 19                 DAD D           ;
050B 78                 MOV A,B         ;
050C E607               ANI     07H     ;
050E 5F                 MOV E,A         ;
050F 7E                 MOV A,M         ;
0510 87         JMP41:  ADD A           ;
0511 1D                 DCR E           ;
0512 F21005             JP      JMP41   ;
0515 9F                 SBB A           ;
0516 D1                 POP D           ;RESTORE OUTPUT COORDS
0517 C9                 RET             ;HL=PTR INTO IMAGE, A=0 IF IMAGE DONE, 0FFH IF NOT
                        ;#########################################################################
```

```
                         ;/##############################################################################
0518   1600      EXA2HL: MVI D,     00H      ;ENTER HERE TO FIND E*A=HL
051A   210000            LXI H,     ZEROS    ;CLEAR RESULT REGISTER SET IF DESIRED
051D   D5        DEXAPH: PUSH D              ;ENTRY TO FIND (DE*A)+HL=HL
051E   B7                ORA A               ;CLEAR CARRY FLAG INITIALLY
051F   1F        NXTBIT: RAR                 ;IF THE A REG IS ODD, SET THE CARRY FLAG
0520   D22405            JNC        EVEN     ;IF IT WAS EVEN, JUMP
0523   19                DAD D               ;DBL PRECISION ADD TEMP RESULT TO SUM
0524   EB        EVEN:   XCHG                ;(
0525   29                DAD H               ;(MULTIPLY DE TIMES 2
0526   EB                XCHG                ;(
0527   B7                ORA A               ;CHECK MULTIPLIER FOR ZERO
0528   C21F05            JNZ        NXTBIT   ;IF NOT ZERO, KEEP MULTIPLYING
052B   D1                POP D               ;ELSE, RETURN W/FINAL RESULT IN HL
052C   C9                RET                 ;EXA2HL KILLS D, BOTH ROUTINES KILL A
                         ;###########################################################################
                         ;                                  GETS ADR OF SCREEN BYTE & PIXEL MASK
                         ;                                  ...FOR THE COLOR TO OUTPUT
X 052D 6B        BYTMSK: MOV L,E             ;Y SCREEN LOCATION INTO THE L REGISTER
052E   2600              MVI H,     00H      ;CLEAR H REGISTER
0530   29                DAD H               ;Y LOCATION X 2
0531   29                DAD H               ;Y LOCATION X 4
0532   29                DAD H               ;Y LOCATION X 8
0533   29                DAD H               ;Y LOCATION X 16 = 10H
0534   29                DAD H  -> DAD H     ;Y LOCATION X 32 = 20H = BYTE WIDTH OF SCREEN
0535   7A                MOV A,D             ;X SCREEN LOCATION INTO THE A REGISTER
0536   1F                RAR                 ;(SHIFT X LOC. 2 BITS TO THE RIGHT
0537   1F                RAR                 ;(
0538   E63F              ANI        3FH      ;MASK OFF BOTTOM 6 BITS
053A   B5                ORA L               ;
053B   6F                MOV L,A             ;
053C   7A                MOV A,D             ;X SCREEN COORD INTO A
053D   EB                XCHG                ;SCREEN OFFSET VALUE TO E
053E   E603              ANI        03H      ;WHICH PAIR OF BITS TO ADDRESS? TOLD BY X COORD
0540   214B05            LXI H,     PXLMSK   ;ADR OF PIXEL MASK
0543   85                ADD L               ;ADD VALUE CORRESPONDING TO PIXEL IN BYTE
0544   6F                MOV L,A             ;(IE. 00112233)
0545   7E                MOV A,M             ;GET PIXEL MASK VALUE
0546   2AE75F            LHLD       SCNSTR   ;GRAB TOP OF SCREEN ADR
0549   19                DAD D               ;ADD OFFSET TO TOP OF SCREEN ADR
054A   C9                RET                 ;HL=ADR IN SCREENTO OUTPUT, A=PIXEL MASK,DE=OFFSET
                         ;###########################################################################
054B   03        PXLMSK: DB         03H      ;BINARY 0000 0011
054C   0C                DB         0CH      ;BINARY 0000 1100
054D   30                DB         30H      ;BINARY 0011 0000
054E   C0                DB         0C0H     ;BINARY 1100 0000
                         ;###########################################################################
```

```
                        ;###############################################################
054F CDF705     OUTMSG: CALL    SCRADR  ;LOAD TOP OF SCREEN ADR @ SCNSTR
0552 0A         NXCHAR: LDAX B          ;GET CHAR FROM MESSAGE
0553 CD5C05             CALL    AMXOUT  ;PUTS OUT CHAR IN A REGISTER
0556 D0                 RNC             ;
0557 03                 INX B           ;INCREMENT CHAR POINTER ADR
0558 C35205             JMP     NXCHAR  ;GO GET NEXT CHAR FROM STRING
                        ;###############################################################
0558 79         CMXOUT: MOV A,C         ;ENTRY TO OUTPUT CHAR IN THE C REG
055C CD3F04     AMXOUT: CALL    ACHROT  ;OUTPUT CHAR IN THE A REGISTER
055F D0                 RNC             ;
0560 3AEA5F             LDA     WIDTH   ;GRAB CHAR WIDTH DIMENSION
0563 3C                 INR A           ;BUMP IT FOR SPACING
0564 82                 ADD D           ;ADD IT TO CURRENT X PRINT POSITION
0565 FE67               CPI     67H     ;CHECK NEW VALUE VS EDGE OF SCREEN VALUE
0567 57                 MOV D,A         ;RESTORE NEW X PRINT POSITION
0568 D8                 RC              ;IF STILL IN VISIBLE SCREEN AREA, RETURN
0569 3AE95F             LDA     HEIGHT  ;IF NOT, GRAB CHAR HEIGHT VALUE
056C 3C                 INR A           ;BUMP IT FOR SPACING
056D 83                 ADD E           ;ADD RESULT TO CURRENT Y PRINT POSITION
056E 5F                 MOV E,A         ;RESTORE NEW Y PRINT POSITION
056F 1606               MVI D,  06H     ;SET NEW DEFAULT START LOC @ LEFT OF SCREEN
0571 37                 STC             ;SET THE CARRY FOR A VALID RETURN
0572 C9                 RET             ;
                        ;###############################################################
0573 0E00       CLRSCR: MVI C,  00H     ;OUTPUT COLOR TO GO TO SCREEN
0575 E5                 PUSH H          ;SAVE USED REGISTERS
0576 D5                 PUSH D          ;
0577 79                 MOV A,C         ;SCREEN COLOR TO THE A REGISTER
0578 CDE805             CALL    MSKSET  ;SET COLOR MASK #
057B 4F                 MOV C,A         ;COLOR MASK # TO THE C REGISTER
057C 210060             LXI H,  SCRNTP  ;ADR OF TOP OF SCREEN
057F 11A009             LXI D,  09A0H   ;# OF SCREEN BYTES IN DE
0582 71         CLRBYT: MOV M,C         ;COLOR MASK TO CORE
0583 23                 INX H           ;BUMP SCREEN POINTER
0584 1B                 DCX D           ;DECREMENT SCREEN BYTE COUNTER
0585 7A                 MOV A,D         ;CHECK BYTE COUNTER FOR ZERO
0586 B3                 ORA E           ;
0587 C28205             JNZ     CLRBYT  ;IF COUNTER NOT ZERO, CYCLE BACK
058A D1                 POP D           ;RESTORE USED REGISTERS
058B E1                 POP H           ;
058C C9                 RET             ;
                        ;###############################################################
```

```
                        ;/##########################################################################
058D  CDE705            CALL    SCRADR  ;LOAD TOP OF SCREEN ADR INTO ROM VARIABLE
0590  210205            LXI H,  JMP44   ;
0593  22E85F            SHLD    PCHLAD  ;
0596  CDB305            CALL    IMGLOC  ;GET ADR OF IMAGE & IMAGE DEFINITION
0599  CDE805            CALL    MSKSET  ;CALCULATE ALT COLOR MASK VALUE
059C  32ED5F            STA     ALTCLR  ;STORE AWAY ALTERNATE COLOR VALUE
059F  C3A504            JMP     DUTIMG  ;GO TO OUTPUT IMAGE ONTO THE SCREEN
                        ;##########################################################################
05A2  CDE705    RFILL:  CALL    SCRADR  ;GET & STORE TOP OF SCREEN ADR @ VAR LOC
05A5  21DF9F            LXI H,  JMP39   ;
05A8  22E85F            SHLD    PCHLAD  ;
05AB  60                MOV H,B         ;PUT HIGH BYTE OF ADR OF BLOCK DESCR. IN H
05AC  69                MOV L,C         ;PUT LOW BYTE OF ADR OF BLOCK DESCR. IN L
05AD  CDBE05            CALL    IMGDEF  ;GET IMAGE SIZE, COLOR, & SCREEN LOCATION
05B0  C3A504            JMP     DUTIMG  ;GO TO OUTPUT IMAGE ONTO THE SCREEN
                        ;##########################################################################
05B3  60        IMGLOC: MOV H,B         ;ADR OF CONTROL TABLE TO HL
05B4  69                MOV L,C         ;
05B5  5E                MOV E,M         ;LSB OF IMAGE BIT PATTERN ADR TO E
05B6  23                INX H           ;
05B7  56                MOV D,M         ;MSB OF IMAGE BIT PATTERN ADR TO D
05B8  23                INX H           ;
05B9  EB                XCHG            ;SWAP IMAGE BIT PATTERN ADR TO HL
05BA  22E55F            SHLD    IMGADR  ;STORE AWAY IMAGE ADR
05BD  EB                XCHG            ;SWAP TABLE ADR BACK TO HL
05BE  5E        IMGDEF: MOV E,M         ;GET IMAGE X WIDTH IN E
05BF  23                INX H           ;NEXT DATA ADR
05C0  56                MOV D,M         ;GET IMAGE Y HEIGHT IN D
05C1  23                INX H           ;NEXT DATA ADR
05C2  D5                PUSH D          ;SAVE IMAGE SIZE ON STACK
05C3  7E                MOV A,M         ;GET IMAGE COLOR IN A
05C4  23                INX H           ;NEXT DATA ADR
05C5  CD2F06            CALL    ACOLOR  ;SET OUTPUT COLOR
05C8  5E        fm01    MOV E,M         ;GET IMAGE X COORDINATE IN E
05C9  23                INX H           ;NEXT DATA ADR
05CA  56                MOV D,M         ;GET IMAGE Y COORDINATE IN D
05CB  23                INX H           ;NEXT DATA ADR
05CC  7E                MOV A,M         ;GET ALTERNATE COLOR VALUE IN A
05CD  E1                POP H           ;RESTORE IMAGE DIMENSIONS IN HL
05CE  010000            LXI B,  ZEROS   ;CLEAR BC
05D1  C9                RET             ;
                        ;##########################################################################
```

```
                        ;##########################################################################
05D2  CDED04    JMP44:  CALL    IMGPTR  ;CALCULATE IMAGE POINTER
05D5  C8                RZ              ;IF IMAGE COMPLETE, RETURN
05D6  CD2D05            CALL    BYTMSK  ;GET SCREEN ADR & PIXEL MASK FOR BYTE OUTPUT
05D9  47                MOV B,A         ;PIXEL MASK VALUE SAVED IN B
05DA  A6                ANA M           ;AND TOGETHER SCREEN BYTE & MASK INTO A
05DB  4F                MOV C,A         ;SAVE MASKED VALUE IN C
05DC  3AEE5F            LDA     CLRMSK  ;GET CURRENT OUTPUT COLOR MASK
05DF  A0                ANA B           ;
05E0  B9                CMP C           ;
05E1  C0                RNZ             ;
05E2  3AED5F            LDA     ALTCLR  ;
05E5  C3E804            JMP     SETPXL  ;
                        ;##########################################################################
05E8  E603      MSKSET: ANI     03H     ;MASK OFF BOTTOM 2 BITS OF SELECTED COLOR
05EA  C5                PUSH B          ;SAVE USED REGISTER PAIR
05EB  4F                MOV C,A         ;TWO BYTE MASKED VALUE TO THE C REGISTER
05EC  87                ADD A           ;MASK X 2
05ED  87                ADD A           ;MASK X 4
05EE  81                ADD C           ;MASK X 5
05EF  87                ADD A           ;MASK X 10 = 0AH
05F0  87                ADD A           ;MASK X 20 = 14H
05F1  81                ADD C           ;MASK X 21 = 15H
05F2  87                ADD A           ;MASK X 42 = 2AH
05F3  87                ADD A           ;MASK X 84 = 54H
05F4  81                ADD C           ;MASK X 85 = 55H = 01010101B
05F5  C1                POP B           ;RESTORE BC REGISTER PAIR
05F6  C9                RET             ;COLOR MASK RETURNS IN THE A REGISTER
                        ;##########################################################################
05F7  E5        SCRADR: PUSH H          ;SAVE HL REGISTER PAIR
05F8  21D040            LXI H,  SCRNTP  ;ADR OF TOP OF SCREEN
05FB  22E75F            SHLD    SCNSTR  ;STORED AWAY
05FE  E1                POP H           ;RESTORE HL REGISTER PAIR
05FF  C9                RET             ;
                        ;##########################################################################
```

```
                        ;################################################################
0600  E5        PLOT:   PUSH H           ;
0601  D5                PUSH D           ;X LOC PASSED IN D, Y LOC PASSED IN E
0602  C5                PUSH B           ;OUTPUT COLOR PASSED IN C
0603  CDE705            CALL    SCRADR   ;LOAD & STORE SCREEN TOP ADR
0606  79                MOV A,C          ;PASS IN DESIRED COLOR IN THE C REGISTER
0607  CDE805            CALL    MSKSET   ;SET COLOR MASK INTO THE A-REGISTER
060A  CDE204            CALL    SETDOT   ;PLACE PIXEL INTO SCREEN
060D  C1                POP B            ;
060E  D1                POP D            ;
060F  E1                POP H            ;
0610  CDE705    POINT:  CALL    SCRADR   ;LOAD & STORE TOP OF SCREEN ADR
0613  E5                PUSH H           ;
0614  D5                PUSH D           ;
0615  C5                PUSH B           ;B HAS X SCREEN LOC, C HAS Y SCREEN LOC
0616  50                MOV D,B          ;X SCREEN LOCATION TO THE D REGISTER
0617  59                MOV E,C          ;Y SCREEN LOCATION TO THE E REGISTER
0618  78                MOV A,B          ;X SCREEN LOCATION TO THE A REGISTER
0619  E603              ANI     03H      ;MASK OFF BOTTOM 2 BITS OF X LOCATION
061B  47                MOV B,A          ;SAVE BIT PAIR # OF SCREEN BYTE TO B
061C  CD2D05            CALL    BYTMSK   ;GET SCREEN ADR & PIXEL MASK FOR BYTE OUTPUT
061F  7E        PT1     MOV A,M          ;GET SCREEN BYTE TO A
0620  0F        GETBTS: RRC              ;FIND PROPER BIT PAIR USING COUNT IN B
0621  0F                RRC
0622  05                DCR B            ;DECREMENT BIT PAIR COUNTER
0623  F22006            JP      GETBTS   ;
0626  07                RLC              ;ROTATED RIGHT ONCE TOO MANY TIMES,...
0627  07                RLC              ;(...SO BACK IT UP TO COMPENSATE
0628  E603              ANI     03H      ;MAXIMUM COLOR # IS 3
062A  C1                POP B            ;
062B  D1                POP D            ;
062C  E1                POP H            ;
062D  C9                RET              ;A RETURNS W/SCREEN COLOR AT BC COORDS
                        ;################################################################
062E  79        CCOLOR: MOV A,C          ;CHAR COLOR TO THE A REGISTER
062F  CDE805    ACOLOR: CALL    MSKSET   ;FIND CHAR COLOR MASK
0632  32EE5F            STA     CLRMSK   ;STORE CHAR COLOR AWAY
0635  C9                RET              ;RETURN W/COLOR MASK IN THE A REGISTER
                        ;################################################################
```

```
                        ;/##########################################################################
0636  D5      SYSCLR: PUSH D
0637  3ADB5F          LDA     CLRSO2  ;GET PREVIOUS SYSTEM COLORS 0 & 2
063A  E6C0            ANI     0C0H    ;GRAB TWO UPPER BITS INTO A (IF SET)
063C  CD5E06          CALL    CLRSET  ;MERGE SAVED DATA & COLORS 0 & 2 INTO A
063F  320010          STA     CLRGA   ;SET NEW SYSTEM COLORS 0 & 2
0642  32DB5F          STA     CLRSO2  ;& SAVE THEM AWAY IN RAM
0645  0B              DCX B           ;POINT BC AT COLOR #1 STORAGE LOCATION
0646  3ADC5F          LDA     CLRS13  ;GET PREVIOUS SYSTEM COLORS 1 & 3
0649  E680            ANI     80H     ;MASK OFF SOFTWARE SOUND BIT INTO A
064B  CD5E06          CALL    CLRSET  ;MERGE S/W SOUND BIT & COLORS 0 & 2 INTO A
064E  57              MOV D,A         ;SAVE MERGED VALUE INTO D
064F  03              INX B           ;POINT BC AT END FLAG PAST COLOR TABLE
0650  0A              LDAX B          ;GRAB END FLAG, ALSO INTENSITY OF COLOR 2 BYTE
0651  0F              RRC             ;(POSITION BIT #0 AT BIT #6 LOCATION
0652  0F              RRC             ;( BIT #'S = 7654 3210
0653  E640            ANI     40H     ;MASK OFF INTENSITY BIT INTO A
0655  B2              ORA D           ;MERGE BIT WITH COLOR BITS & DATA BIT
0656  320018          STA     CLRGB   ;SET SYSTEM COLORS 1 & 3, W/INTENSITY FOR #2
0659  32DC5F          STA     CLRS13  ;& SAVE THEM AWAY IN RAM
065C  D1              POP D           ;
065D  C9              RET             ;
                        ;##########################################################################
065E  57      CLRSET: MOV D,A         ;SAVE OLD UPPER BITS INTO THE D REG
065F  0A              LDAX B          ;GRAB NEW COLOR #(0 ON 1ST PASS, 1 ON 2ND)
0660  03              INX B           ;BUMP COLOR TABLE POINTER
0661  E607            ANI     07H     ;MAXIMUM COLOR # = 7 (BOTTOM 3 BITS)
0663  B2              ORA D           ;MASK OLD UPPER BITS WITH NEW COLOR #
0664  57              MOV D,A         ;...& PUT RESULT BACK INTO D
0665  03              INX B           ;BC NOW POINTS AT NEXT COLOR (2 OR 3)
0666  0A              LDAX B          ;GRAB NEXT NEW COLOR VALUE
0667  E607            ANI     07H     ;MAXIMUM COLOR # = 7 (BOTTOM 3 BITS)
0669  17              RAL             ;(POSITION 2ND NEW COLOR # IN ITS' PROPER...
066A  17              RAL             ;(...LOCATION
066B  17              RAL             ;(
066C  B2              ORA D           ;& THEN MASK IN PRIOR CALCULATED RESULT
066D  C9              RET
                        ;##########################################################################
```

```
                        ;/##########################################################################
066E  05        CHRDEF: DB      05H     ;CHAR HEIGHT
066F  05                DB      05H     ;CHAR WIDTH
                        ;--------------------------------------------------------------------------
0670  00                DB      00H     ;ASCII 1E
0671  50                DB      50H     ;                                               X X
0672  20                DB      20H     ;SPECIAL-CHARACTER-(SMALL-X)                     X
0673  50                DB      50H     ;                                               X X
0674  00                DB      00H     ;
                        ;--------------------------------------------------------------------------
0675  20                DB      20H     ;ASCII 1F                                        X
0676  00                DB      00H     ;
0677  70                DB      70H     ;SPECIAL CHARACTER (DIVISION SIGN)              XXX
0678  00                DB      00H     ;                                                X
0679  20                DB      20H     ;
                        ;--------------------------------------------------------------------------
067A  00        ASCHRS: DB      00H     ;ASCII 20
067B  00                DB      00H     ;' '=NULL SPACE
067C  00                DB      00H     ;
067D  00                DB      00H     ;
067E  00                DB      00H     ;
                        ;--------------------------------------------------------------------------
067F  20                DB      20H     ;ASCII 21                                        X
0680  20                DB      20H     ;                                                X
0681  20                DB      20H     ;'!'=EXCLAMATION POINT                           X
0682  00                DB      00H     ;                                                X
0683  20                DB      20H     ;
                        ;--------------------------------------------------------------------------
0684  50                DB      50H     ;ASCII 22                                       X X
0685  50                DB      50H     ;                                               X X
0686  00                DB      00H     ;' '=DOUBLE QUOTE
0687  00                DB      00H     ;
0688  00                DB      00H     ;
                        ;--------------------------------------------------------------------------
0689  50                DB      50H     ;ASCII 23                                       X X
068A  F8                DB      0F8H    ;                                              XXXXX
068B  50                DB      50H     ;'#'=NUMBER SIGN                                X X
068C  F8                DB      0F8H    ;                                              XXXXX
068D  50                DB      50H     ;                                               X X
                        ;--------------------------------------------------------------------------
068E  70                DB      70H     ;ASCII 24                                       XXX
068F  A0                DB      0A0H    ;                                              X X
0690  70                DB      70H     ;'$'=DOLLAR SIGN                                XXX
0691  28                DB      28H     ;                                                X X
0692  70                DB      70H     ;                                               XXX
                        ;--------------------------------------------------------------------------
0693  C8                DB      0C8H    ;ASCII 25                                      XX   X
0694  D0                DB      0D0H    ;                                              XX  X
0695  20                DB      20H     ;'%'=PERCENT SIGN                                X
0696  58                DB      58H     ;                                               X  XX
0697  98                DB      98H     ;                                              X   XX
                        ;--------------------------------------------------------------------------
```

```
                        ;
0698  20                 DB    20H     ;ASCII 26
0699  50                 DB    50H     ;'&'=AMPERSAND SIGN
069A  20                 DB    20H     ;
069B  50                 DB    50H     ;
069C  28                 DB    28H     ;
                        ;
069D  20                 DB    20H     ;ASCII 27
069E  40                 DB    40H     ;'''=SINGLE QUOTE
069F  00                 DB    00H     ;
06A0  00                 DB    00H     ;
06A1  00                 DB    00H     ;
                        ;
06A2  20                 DB    20H     ;ASCII 28
06A3  40                 DB    40H     ;'('=OPEN PARENTHESIS
06A4  40                 DB    40H     ;
06A5  40                 DB    40H     ;
06A6  20                 DB    20H     ;
                        ;
06A7  20                 DB    20H     ;ASCII 29
06A8  10                 DB    10H     ;')'=CLOSE PARENTHESIS
06A9  10                 DB    10H     ;
06AA  10                 DB    10H     ;
06AB  20                 DB    20H     ;
                        ;
06AC  A8                 DB    0A8H    ;ASCII 2A
06AD  70                 DB    70H     ;'*'=ASTERISK
06AE  F8                 DB    0F8H    ;
06AF  70                 DB    70H     ;
06B0  A8                 DB    0A8H    ;
                        ;
06B1  00                 DB    00H     ;ASCII 2B
06B2  20                 DB    20H     ;'+'=PLUS SIGN
06B3  70                 DB    70H     ;
06B4  20                 DB    20H     ;
06B5  00                 DB    00H     ;
                        ;
06B6  00                 DB    00H     ;ASCII 2C
06B7  00                 DB    00H     ;','=COMMA
06B8  00                 DB    00H     ;
06B9  20                 DB    20H     ;
06BA  40                 DB    40H     ;
                        ;
06BB  00                 DB    00H     ;ASCII 2D
06BC  00                 DB    00H     ;'-'=MINUS SIGN
06BD  70                 DB    70H     ;
06BE  00                 DB    00H     ;
06BF  00                 DB    00H     ;
                        ;
```

```
                        ;/-----------------------------------------------------------
06C0  00                DB      00H     ;ASCII 2E
06C1  00                DB      00H     ;
06C2  00                DB      00H     ;'.'=PERIOD
06C3  00                DB      00H     ;
06C4  20                DB      20H     ;
                        ;-----------------------------------------------------------
06C5  08                DB      08H     ;ASCII 2F
06C6  10                DB      10H     ;
06C7  20                DB      20H     ;'/'=FRONT SLASH
06C8  40                DB      40H     ;
06C9  80                DB      80H     ;
                        ;-----------------------------------------------------------
06CA  70                DB      70H     ;ASCII 30
06CB  88                DB      88H     ;
06CC  88                DB      88H     ;'0'=ZERO
06CD  88                DB      88H     ;
06CE  70                DB      70H     ;
                        ;-----------------------------------------------------------
06CF  20                DB      20H     ;ASCII 31
06D0  60                DB      60H     ;
06D1  20                DB      20H     ;'1'=ONE
06D2  20                DB      20H     ;
06D3  70                DB      70H     ;
                        ;-----------------------------------------------------------
06D4  F8                DB      0F8H    ;ASCII 32
06D5  08                DB      08H     ;
06D6  F8                DB      0F8H    ;'2'=TWO
06D7  80                DB      80H     ;
06D8  F8                DB      0F8H    ;
                        ;-----------------------------------------------------------
06D9  F8                DB      0F8H    ;ASCII 33
06DA  08                DB      08H     ;
06DB  38                DB      38H     ;'3'=THREE
06DC  08                DB      08H     ;
06DD  F8                DB      0F8H    ;
                        ;-----------------------------------------------------------
06DE  90                DB      90H     ;ASCII 34
06DF  90                DB      90H     ;
06E0  F8                DB      0F8H    ;'4'=FOUR
06E1  10                DB      10H     ;
06E2  10                DB      10H     ;
                        ;-----------------------------------------------------------
06E3  F0                DB      0F0H    ;ASCII 35
06E4  80                DB      80H     ;
06E5  F0                DB      0F0H    ;'5'=FIVE
06E6  08                DB      08H     ;
06E7  F0                DB      0F0H    ;
                        ;-----------------------------------------------------------
```

```
                        ;/-------------------------------------------------------------------------
06E8  F8                DB      0F8H    ;ASCII 36                               xxxxx
06E9  80                DB      80H     ;                                       x
06EA  F8                DB      0F8H    ;'6'=SIX                                xxxxx
06EB  88                DB      88H     ;                                       x   x
06EC  F8                DB      0F8H    ;                                       xxxxx
                        ;-------------------------------------------------------------------------
06ED  F8                DB      0F8H    ;ASCII 37                               xxxxx
06EE  08                DB      08H     ;                                           x
06EF  10                DB      10H     ;'7'=SEVEN                                x
06F0  20                DB      20H     ;                                         x
06F1  20                DB      20H     ;                                         x
                        ;-------------------------------------------------------------------------
06F2  F8                DB      0F8H    ;ASCII 38                               xxxxx
06F3  88                DB      88H     ;                                       x   x
06F4  F8                DB      0F8H    ;'8'=EIGHT                              xxxxx
06F5  88                DB      88H     ;                                       x   x
06F6  F8                DB      0F8H    ;                                       xxxxx
                        ;-------------------------------------------------------------------------
06F7  F8                DB      0F8H    ;ASCII 39                               xxxxx
06F8  88                DB      88H     ;                                       x   x
06F9  F8                DB      0F8H    ;'9'=NINE                               xxxxx
06FA  08                DB      08H     ;                                           x
06FB  F8                DB      0F8H    ;                                       xxxxx
                        ;-------------------------------------------------------------------------
06FC  00                DB      00H     ;ASCII 3A                                   x
06FD  20                DB      20H     ;                                           x
06FE  00                DB      00H     ;':'=COLON
06FF  20                DB      20H     ;                                           x
0700  00                DB      00H     ;
                        ;-------------------------------------------------------------------------
0701  00                DB      00H     ;ASCII 3B                                   x
0702  20                DB      20H     ;                                           x
0703  00                DB      00H     ;';'=SEMI-COLON                             x
0704  20                DB      20H     ;                                         x
0705  40                DB      40H     ;                                       x
                        ;-------------------------------------------------------------------------
0706  10                DB      10H     ;ASCII 3C                                   x
0707  20                DB      20H     ;                                         x
0708  40                DB      40H     ;'<'=LESS THAN SIGN                     x
0709  20                DB      20H     ;                                         x
070A  10                DB      10H     ;                                           x
                        ;-------------------------------------------------------------------------
070B  00                DB      00H     ;ASCII 3D                               xxx
070C  70                DB      70H     ;                                       xxx
070D  00                DB      00H     ;'='=EQUAL SIGN
070E  70                DB      70H     ;
070F  00                DB      00H     ;
                        ;-------------------------------------------------------------------------
```

```
0710  40          ;/-------------------------------------------------------------
0711  20          DB      40H     ;ASCII 3E                       X
0712  10          DB      20H     ;                              X
0713  20          DB      10H     ;'>'=GREATER THAN SIGN        X
0714  40          DB      20H     ;                              X
                  DB      40H     ;                               X
                ;-------------------------------------------------------------
0715  F8          DB      0F8H    ;ASCII 3F                      XXXXX
0716  88          DB      88H     ;                              X   X
0717  38          DB      38H     ;'?'=QUESTION MARK               XXX
0718  00          DB      00H     ;                                X
0719  20          DB      20H     ;
                ;-------------------------------------------------------------
071A  30          DB      30H     ;ASCII 40                        XX
071B  48          DB      48H     ;                              X  X
071C  50          DB      50H     ;'@'=MASTERSPACE               X X
071D  40          DB      40H     ;                              X
071E  38          DB      38H     ;                                XXX
                ;-------------------------------------------------------------
071F  70          DB      70H     ;ASCII 41                        XXX
0720  88          DB      88H     ;                              X   X
0721  F8          DB      0F8H    ;'A'                           XXXXX
0722  88          DB      88H     ;                              X   X
0723  88          DB      88H     ;                              X   X
                ;-------------------------------------------------------------
0724  F8          DB      0F8H    ;ASCII 42                      XXXXX
0725  88          DB      88H     ;                              X   X
0726  F0          DB      0F0H    ;'B'                           XXXX
0727  88          DB      88H     ;                              X   X
0728  F8          DB      0F8H    ;                              XXXXX
                ;-------------------------------------------------------------
0729  F8          DB      0F8H    ;ASCII 43                      XXXXX
072A  80          DB      80H     ;                              X
072B  80          DB      80H     ;'C'                           X
072C  80          DB      80H     ;                              X
072D  F8          DB      0F8H    ;                              XXXXX
                ;-------------------------------------------------------------
072E  F0          DB      0F0H    ;ASCII 44                      XXXX
072F  88          DB      88H     ;                              X   X
0730  88          DB      88H     ;'D'                           X   X
0731  88          DB      88H     ;                              X   X
0732  F0          DB      0F0H    ;                              XXXX
                ;-------------------------------------------------------------
0733  F8          DB      0F8H    ;ASCII 45                      XXXXX
0734  80          DB      80H     ;                              X
0735  E0          DB      0E0H    ;'E'                           XXX
0736  80          DB      80H     ;                              X
0737  F8          DB      0F8H    ;                              XXXXX
                ;-------------------------------------------------------------
```

```
                        ;/----------------------------------------------------------
0738    F8                      DB      0F8H    ;ASCII 46       XXXXX
0739    80                      DB      80H     ;               X
073A    E0                      DB      0E0H    ;'F'            XXX
073B    80                      DB      80H     ;               X
073C    80                      DB      80H     ;               X
                        ;-----------------------------------------------------------
073D    F8                      DB      0F8H    ;ASCII 47       XXXXX
073E    80                      DB      80H     ;               X
073F    98                      DB      98H     ;'G'            X  XX
0740    88                      DB      88H     ;               X   X
0741    F8                      DB      0F8H    ;               XXXXX
                        ;-----------------------------------------------------------
0742    88                      DB      88H     ;ASCII 48       X   X
0743    88                      DB      88H     ;               X   X
0744    F8                      DB      0F8H    ;'H'            XXXXX
0745    88                      DB      88H     ;               X   X
0746    88                      DB      88H     ;               X   X
                        ;-----------------------------------------------------------
0747    70                      DB      70H     ;ASCII 49        XXX
0748    20                      DB      20H     ;                 X
0749    20                      DB      20H     ;'I'              X
074A    20                      DB      20H     ;                 X
074B    70                      DB      70H     ;                XXX
                        ;-----------------------------------------------------------
074C    08                      DB      08H     ;ASCII 4A           X
074D    08                      DB      08H     ;                   X
074E    08                      DB      08H     ;'J'                X
074F    88                      DB      88H     ;               X   X
0750    F8                      DB      0F8H    ;               XXXXX
                        ;-----------------------------------------------------------
0751    88                      DB      88H     ;ASCII 4B       X   X
0752    90                      DB      90H     ;               X  X
0753    E0                      DB      0E0H    ;'K'            XXX
0754    90                      DB      90H     ;               X  X
0755    88                      DB      88H     ;               X   X
                        ;-----------------------------------------------------------
0756    80                      DB      80H     ;ASCII 4C       X
0757    80                      DB      80H     ;               X
0758    80                      DB      80H     ;'L'            X
0759    80                      DB      80H     ;               X
075A    F8                      DB      0F8H    ;               XXXXX
                        ;-----------------------------------------------------------
075B    88                      DB      88H     ;ASCII 4D       X   X
075C    D8                      DB      0D8H    ;               XX XX
075D    A8                      DB      0A8H    ;'M'            X X X
075E    88                      DB      88H     ;               X   X
075F    88                      DB      88H     ;               X   X
                        ;-----------------------------------------------------------
```

```
                  ;-----------------------------------------
0760  88          DB    88H     ;ASCII 4E        X   X
0761  C8          DB    0C8H    ;'N'             XX  X
0762  A8          DB    0A8H    ;                X X X
0763  98          DB    98H     ;                X  XX
0764  88          DB    88H     ;                X   X
                  ;-----------------------------------------
0765  F8          DB    0F8H    ;ASCII 4F        XXXXX
0766  88          DB    88H     ;'O'             X   X
0767  88          DB    88H     ;                X   X
0768  88          DB    88H     ;                X   X
0769  F8          DB    0F8H    ;                XXXXX
                  ;-----------------------------------------
076A  F8          DB    0F8H    ;ASCII 50        XXXXX
076B  88          DB    88H     ;'P'             X   X
076C  F8          DB    0F8H    ;                XXXXX
076D  80          DB    80H     ;                X
076E  80          DB    80H     ;                X
                  ;-----------------------------------------
076F  F8          DB    0F8H    ;ASCII 51        XXXXX
0770  88          DB    88H     ;'Q'             X   X
0771  A8          DB    0A8H    ;                X X X
0772  90          DB    90H     ;                X  X
0773  E8          DB    0E8H    ;                XXX X
                  ;-----------------------------------------
0774  F8          DB    0F8H    ;ASCII 52        XXXXX
0775  88          DB    88H     ;'R'             X   X
0776  F8          DB    0F8H    ;                XXXXX
0777  90          DB    90H     ;                X  X
0778  88          DB    88H     ;                X   X
                  ;-----------------------------------------
0779  F8          DB    0F8H    ;ASCII 53        XXXXX
077A  80          DB    80H     ;'S'             X
077B  F8          DB    0F8H    ;                XXXXX
077C  08          DB    08H     ;                    X
077D  F8          DB    0F8H    ;                XXXXX
                  ;-----------------------------------------
077E  F8          DB    0F8H    ;ASCII 54        XXXXX
077F  20          DB    20H     ;'T'               X
0780  20          DB    20H     ;                  X
0781  20          DB    20H     ;                  X
0782  20          DB    20H     ;                  X
                  ;-----------------------------------------
0783  88          DB    88H     ;ASCII 55        X   X
0784  88          DB    88H     ;'U'             X   X
0785  88          DB    88H     ;                X   X
0786  88          DB    88H     ;                X   X
0787  F8          DB    0F8H    ;                XXXXX
                  ;-----------------------------------------
```

```
                      ;/----------------------------------------------------
0788   88             DB      88H     ;ASCII 56              X   X
0789   88             DB      88H     ;                      X   X
078A   50             DB      50H     ;'V'                    X X
078B   70             DB      70H     ;                       XXX
078C   20             DB      20H     ;                        X
                      ;-----------------------------------------------------
078D   88             DB      88H     ;ASCII 57              X   X
078E   88             DB      88H     ;                      X   X
078F   A8             DB      0A8H    ;'W'                   X X X
0790   D8             DB      0D8H    ;                      XX XX
0791   88             DB      88H     ;                      X   X
                      ;-----------------------------------------------------
0792   88             DB      88H     ;ASCII 58              X   X
0793   50             DB      50H     ;                       X X
0794   20             DB      20H     ;'X'                     X
0795   50             DB      50H     ;                       X X
0796   88             DB      88H     ;                      X   X
                      ;-----------------------------------------------------
0797   88             DB      88H     ;ASCII 59              X   X
0798   50             DB      50H     ;                       X X
0799   20             DB      20H     ;'Y'                     X
079A   20             DB      20H     ;                        X
079B   20             DB      20H     ;                        X
                      ;-----------------------------------------------------
079C   F8             DB      0F8H    ;ASCII 5A              XXXXX
079D   10             DB      10H     ;                         X
079E   20             DB      20H     ;'Z'                     X
079F   40             DB      40H     ;                       X
07A0   F8             DB      0F8H    ;                      XXXXX
                      ;-----------------------------------------------------
07A1   30             DB      30H     ;ASCII 5B                XX
07A2   20             DB      20H     ;                        X
07A3   20             DB      20H     ;' '=OPEN SQUARE BRACKET X
07A4   20             DB      20H     ;                        X
07A5   30             DB      30H     ;                        XX
                      ;-----------------------------------------------------
07A6   80             DB      80H     ;ASCII 5C              X
07A7   40             DB      40H     ;                       X
07A8   20             DB      20H     ;' '=BACK SLASH          X
07A9   10             DB      10H     ;                         X
07AA   08             DB      08H     ;                          X
                      ;-----------------------------------------------------
07AB   60             DB      60H     ;ASCII 5D               XX
07AC   20             DB      20H     ;                        X
07AD   20             DB      20H     ;' '=CLOSE SQUARE BRACKET X
07AE   20             DB      20H     ;                        X
07AF   60             DB      60H     ;                       XX
                      ;-----------------------------------------------------
```

```
0780  20          ;/-------------------------------------------------------
0781  50                DB      20H     ;ASCII 5E                    X
0782  86                DB      50H     ;'^'=EXPONENTIATION SIGN    X X
0783  00                DB      88H     ;                           X   X
0784  00                DB      00H     ;
                  ;-------------------------------------------------------
0785  00                DB      00H     ;ASCII 5F
0786  00                DB      00H     ;' '=UNDERLINE
0787  00                DB      00H     ;                           XXXXX
0788  00                DB      00H     ;
0789  F8                DB      0F8H    ;
                  ;-------------------------------------------------------
078A  20                DB      20H     ;ASCII 60                    X
078B  10                DB      10H     ;'`'=REVERSE QUOTE           X
078C  00                DB      00H     ;
078D  00                DB      00H     ;
078E  00                DB      00H     ;
                  ;-------------------------------------------------------
```

```
                        ;#########################################################################
07BF  21DC5F  TONE:     LXI H,   CLRS13  ;ADR OF COLOR REGISTER 0 RAM STORAGE
07C2  7E                MOV A,M          ;GET CURRENT COLOR REGISTER VALUE
07C3  EE80              XRI      80H     ;FLIP SOFTWARE SOUND BIT
07C5  77                MOV M,A          ;STORE CHANGED REGISTER BACK IN RAM STORAGE LOCATION
07C6  320018            STA      CLRGB   ;& AT ACTUAL COLOR REGISTER
07C9  F5                PUSH PSW         ;
07CA  C5                PUSH B           ;SAVE SOUND FREQUENCY ON STACK (PASSED PARAMETER)
07CB  60                MOV H,B          ;(MOVE FREQUENCY TO HL
07CC  69                MOV L,C          ;(
07CD  01FFFF            LXI B,   0FFFFH  ;BC = -1 (SORTOF)
07D0  09        ADDER:  DAD B            ;SUBTRACT 1 FROM FREQUENCY
07D1  DAD007            JC       ADDER   ;
07D4  C1                POP B            ;RESTORE FREQUENCY TO BC
07D5  F1                POP PSW          ;
07D6  FABF07            JM       TONE    ;IF UPPER BIT OF FREQUENCY IS SET, JUMP BACK
07D9  1B                DCX D            ;ELSE, DECREMENT SOUND TIME LENGTH (PASSED PARAMETER)
07DA  7A                MOV A,D          ;(CHECK TONE LENGTH PARAMETER FOR ZERO
07DB  B3                ORA E            ;(
07DC  C2BF07            JNZ      TONE    ;IF COUNTER NOT = ZERO, JUMP BACK
07DF  C9                RET              ;ELSE, RETURN
                        ;#########################################################################
07E0  3AD05F  KYWAIT:   LDA      KYSTAT  ;GET STATUS OF KEY-HIT OR NOT
07E3  B7                ORA A            ;CHECK IT FOR ZERO
07E4  CAE007            JZ       KYWAIT  ;IF ZERO, NO KEY WAS HIT, GO BACK & CHECK AGAIN
07E7  3AD05F  KYCHEK:   LDA      KYSTAT  ;GET STATUS OF KEY-HIT OR NOT
07EA  B7                ORA A            ;CHECK IT FOR ZERO
07EB  C8                RZ               ;IF ZERO, NO KEY WAS HIT, SO RETURN
07EC  07                RLC              ;IF UPPER BIT OF STATUS IS SET, THEN SET CARRY FLAG
07ED  3E00              MVI A,   00H     ;CLEAR A
07EF  32D05F            STA      KYSTAT  ;CLEAR KEY HIT STATUS LOCATION
07F2  3AD15F            LDA      LSTCHR  ;GET LAST CHAR TYPED IN A REGISTER
07F5  C9                RET              ;
                        ;#########################################################################
07F6  C5        DELAY:  PUSH B           ;SAVE WAIT CONSTANT ON STACK
07F7  0B        WAIT1:  DCX B            ;DECREMENT WAIT CONSTANT
07F8  78                MOV A,B          ;HIGH BYTE OF WAIT CONSTANT INTO A
07F9  B1                ORA C            ;CHECK IT AGAINST LOW BYTE
07FA  C2F707            JNZ      WAIT1   ;IF B & C ARE NOT ZERO, JUMP BACK
07FD  C1                POP B            ;RESTORE ORIGINAL WAIT CONSTANT
07FE  C9                RET              ;
                        ;#########################################################################
07FF  FF                DB       0FFH    ;****END OF INTERACT ROM****
                        ;#########################################################################
```

SOFTWARE COPYRIGHT BY R.A.FERRIS   ERRORS = 0 PAGE 81
INTERACT ROM & OPERATING SYSTEM 2.1 LISTING

SYMBOL TABLE

* = 01 REFERENCE ONLY

A	0007	A2DCN	5FF6	A2DFL	5FF5	A2FJM	0C10
ACHRO	043F	ACOLO	062F	ADDER	07DD	ADRCH	097F
ADKOU	0B77	AFBKP	0804	AFREG	4A20	ALTCL	5FED
ALTSE	5FE1	ALTSI	5FE3	AMXOU	055C	ASC2H	0C02
ASCHR	067A	ASCOU	0928	B	0000	BACKS	0CFA
BAKJM	002B	BCREG	4A2D	BEEP	0959	BHL2B	0CC1 *
BHL2D	0CB3	BLD1	039D	BLD2	03A5	BLD3	03A6
BLDBY	039A	BLKCH	0E89	BRKFL	4A25	BS	0008
BTCT1	00AE	BTCT2	00B5	BUFFE	4A3B	BUFPT	4A39
BULK1	02A0	BULKC	0E6E	BULKF	5FD9	BULKI	0299
BYT2T	0364	BYTMS	052D	BYTS2	0C8D	BYTSP	0C6B
C	0001	CCARR	0CC9 *	CCOLO	062E	CDEC1	0A05
CHGSC	09BE	CHRDE	066E	CHRER	0D15	CHROU	043E *
CHRSI	0FDF	CHRTB	0176	CKBUL	027B	CKSCR	0987
CLEAR	0BA4	CLERL	0D3A	CLKAD	3000	CLRBY	0552
CLRGA	1000	CLRGB	1800	CLRMS	5FEE	CLRSO	5FDB
CLRS1	5FDC	CLRSC	0573	CLRSE	065E	CMDER	0DED
CMXOU	055B	CNTSC	0080	CONRO	0D79	CONTR	0D65
CR	000D	CRGET	0019	CRLF	0D31	CTAB1	0217
CTABL	0FCC	CTLBR	0B3D	CTRLC	0B94	CVTDE	0AC3
CVTHE	0A29	D	0002	DATAP	0EA7	DCRBF	0CA9
DELAY	07F6	DEMHL	0CB9	DEREG	4A2F	DEXAP	051D
DISO	0EC2	DIS1	0ECB	DIS2	0ECC	DIS3	0FE9
DISPL	0912	DRWIM	049D	DTAPE	0EA1	E	0003
ENDCH	0AE9	ENDFI	0430	ENDPA	7FFF	ENDRS	0172
ENTER	0CF1	EVEN	0524	EXA2H	0518	EXAKI	0A95
EXIT	0F44	EXIT1	0CAE	FFREC	02AE	FILBU	0CCC
FILL	0940	FILMO	0946	FIXSC	098C	FLBCK	0A69
GAMES	4C00	GAPOU	0E2F	GET1H	0BC2	GET3H	0BC0
GETBT	0620	GETCH	0C99	GETDT	0B1C	GETHL	0BF0
GETKE	0956	GETRG	097A	GOTO	096B	GTFF	0E95
GTIMG	046F	H	0004	HDRDT	0EAE	HDRFL	5FD8
HEIGH	5FE9	HEXAD	0BC4	HIDE1	0CF0	HIDE2	0131
HIDE3	0395	HLMDE	0CBA	HLREG	4A35	HX2AS	0C14
HX2D1	0A4B	HX2D2	0A55	HX2DE	0A4A	IMGAD	5FE5
IMGDE	05BE	IMGLO	05B3	IMGPT	04ED	IOCHA	0C20
JMP38	04DB	JMP39	04DF	JMP41	0510	JMP44	05D2
JMPTB	08DE	JOYST	3B07	JOYVA	5FC1	K1RG0	5FC2
K1RG6	5FC8	K2RG0	5FC9	K2RG6	5FCF	KCHK1	0099
KCHK2	009A	KCHK3	009F	KFOUN	0089	KYBDB	3B00
KYBDT	3806	KYCHE	07E7	KYSTA	5FD0	KYWAI	07E0
L	0005	LCHAN	005D	LDROU	0E2C	LEADR	03B1
LFIRE	5FF7 *	LHJOY	5FF1 *	LINE1	0D12	LINEC	0D64
LNSTR	0642	LOAD	0DAF	LOAD1	0DC9	LPOT	5FF7 *
LSTCH	5FD1	LSTLI	4BED	LTLCA	045A	M	0006
MCHK	0AA7	MEMER	0F9B	MEMEX	0F75	MEMX1	0F78
MEMX2	0F7B	MEMX3	0F89	MERRO	4A22	MHL2B	0CC2 *
MHL2D	0CB4	MORGS	0803	MORLI	092E	MOVE	09C2
MOVED	09CF	MOVEU	09E8	MSG1	0FD1	MSG2	0FDB *

SOFTWARE COPYRIGHT BY R.A.FERRIS ERRORS = 0 PAGE 62
INTERACT ROM & OPERATING SYSTEM 2.1 LISTING

SYMBOL TABLE (CONTINUED)

* = OT REFERENCE ONLY

MSKSE	0SE8		MTRCH	02E5		NB1	0F30		NB2	0F31	
NBYTE	0F23		NEWCM	08A7		NEWLI	0B74		NEXTB	03E7	
NLOCK	00D3		NO2ND	0AE8		NOA2D	016A		NOALT	025F	
NOCHA	0D2A		NOCHG	03D4		NOGOO	03CC		NOKEY	0124	
NOLOR	0357		NOLXI	0187		NOMSK	044F		NORST	0DE9	
NOSET	0123		NOSHF	00F5		NOSPA	0C9D		NOTCR	0BEB	
NOTFU	0918		NOTHE	0BDA		NOTOF	0256		NOTUC	0103	
NRMVO	2801	*	NUMBY	4A5F		NXCHA	0552		NXTAD	0A5C	
NXTBI	051F		NXTCH	0CD7		NXTLD	0382		NXTMO	00F6	
NXTNU	0BC7		NXTPX	0482		OBYTE	0ACA		OCRCH	0C93	
OFF	00D3		OKHDR	025D		OLDSP	4A20		ONELI	0D66	
ORDRE	0AEA		OUT1	0D9D		OUTAC	0C37		OUTAD	0C53	
OUTAS	092B		OUTBY	0C58		OUTCG	03FF		OUTCH	0C67	
OUTDE	0C52		OUTFR	07F8		OUTHE	0C61		OUTIM	04A5	
OUTME	0915		OUTMS	054F		OUTPU	01D3		OUTSP	0D7A	
OUTUC	0C2B		PCHL1	04DA		PCHLA	5FEB		PLOT	0600	*
POINT	0610	*	POP1	0C86		POPPR	0493		PROCE	0E32	
PRT1	0F0B		PRT2	0F17		PRTFL	5FFF		PRTIA	09AA	
PRTIN	0F00		PRTPO	4A26		PSW	0006		PUTNC	09DB	
PXLMS	054B		RD16I	03D0		RDBLO	0300		RDLOO	0308	
RDSTA	5FD3		READ1	0238		READE	021C		READI	031A	
RECDO	034C		RECNT	034F		REGA	4A2C		REGB	4A2E	
REGD	4A30		REGH	4A36		REGPC	4A34		REGSP	4A32	
RERRO	0324		RESTO	000D		REST1	0008		REST7	003E	*
RESUL	0A23		RETHI	0849		REWIN	0053		RFILL	05A2	*
RFIRE	5FFA	*	RGLCT	0F9F		RGSOU	0AF3		RHJOY	5FF2	
RITE1	0E20		RITET	0DF5		RMSG1	01E8		RKSG2	0206	
ROM2	0800		RPLOT	0497	*	RPOT	5FFB	*	RTCLC	5FEF	
SCANC	0C3E		SCANK	007A		SCNST	5FE7		SCRAD	05F7	
SCRNT	4000		SCROL	4A24		SDRG1	2003	*	SDRG2	2803	*
SECRE	0B0A		SETDO	04E2		SETPX	04E8		SHIFT	5FD2	
SKIP1	000C		SKIP7	018C		SNDAB	2802		SP	0006	
SPCHA	0A79		SPCHR	00E9		SPINI	4BF0		SPREG	4A31	
START	0857		STKIO	5FC0		STOPR	024F		STRCH	0116	
STRT1	088D		STRTC	0196		SUBRG	0AB4		SUBST	0A59	
SUMDI	0991		SYSCL	0636		TADAD	4A37		TABL1	0F4F	
TAPEO	020D		TAPOF	07E3		TAPSN	02CA		TAPWR	5FDA	
TOGGL	0433		TONE	07BF		TPBYT	5FD6		TPDAT	0E52	
TPSTR	5FD4		TSTGE	0BF5		UBYTE	5FDF		UDRTC	006A	
UNC1	0835		UNCOV	0B2C		USER1	4BF7		USER2	4DFA	
USER3	4BF0		USRHO	0E61		USRPR	4A28		USRRS	5FF3	
USTAR	5FD0		VARCL	001E		VERIF	0B5D		VLDCN	0336	
VRFY1	0B65		VRFY2	0B6C		WAIT1	07F7		WAITC	0407	
WGAP	0424		WIDTH	5FEA		WRITE	0E23	*	WRTCH	0BAA	
WRTDS	4A61		WRTFM	4A55		WRTSR	4A5D		XMNRG	0A8C	
XQTAD	4A33		Z29JM	0C1D		ZEROS	0000				