TALKING ELECTRONICS MICROCOMP EPROM LISTING

CODE WRITTEN BY COLIN MITCHELL (PROBABLY)
ANNOTATION BY BRIAN CHIHA

FILES: microcomp_2k.rom <- THIS LISTING
microcomp_variant.rom <- SEE ADDENDUM

THE MICROCOMP IS A 3-CHIP Z80 COMPUTER DEVELOPED BY COLIN MITCHELL FOR TALKING ELECTRONICS. IT WAS FEATURED IN ISSUES 13 AND 14 OF THE TALKING ELECTRONICS MAGAZINE PUBLISHED IN THE LATE 80'S. ADD ON AND OTHER PROGRAMS WERE MENTIONED IN THE BD679 BOOK. IT SOLD FOR $55.75 INCLUDING PARTS AND CASE. KEN STONE HAD A SMALL INVOLVEMENT IN THE DESIGN OF THE BOARD.

THERE WERE A FEW ADD-ON THAT WERE ADVERTISED, BUT MOST WERN'T WIDELY KNOWN AND SOME WERN'T PUBLISHED.

THE MICROCOMP WAS SOLD AS A Z80 LEARNING DEVICE THOUGH IT HAS LIMITATIONS. FIRSTLY, TO PROGRAM THE EPROM ANOTHER COMPUTER WAS NEEDED, THE BOARD HAD NO RAM AND NO KEYBOARD. ONLY AN INPUT DIP SWITCH WAS PROVIDED AND SOME BUTTONS TO INTERFACE WITH THE EPROM. THERE WERE SOME CLEVER DESIGN FEATURES TO OVER COME THE LACK OF CHIPS. BIT 7 ON THE OUTPUT LATCH WAS USED TO SELECT WHICH SEVEN SEGMENT DISPLAY TO USE, TRANSISTOR LOGIC WAS USED TO DRIVE THE INPUT OUTPUT REQUEST AND AN AUDIBLE PROBE WAS USED TO 'HEAR' THE Z80 LINES.

THE DEFINITIONS BELOW WILL HELP WITH UNDERSTANDING THE IO ROUTINES.

THE INPUT LATCH IS THE 8 DIP SWITCHES THAT IS ONLY ACTIVE WHEN AN "IN A,(01)" IS CALLED. THIS PLACES THE 8 BITS ONTO THE DATA BUS FOR READING.

THE DISPLAYS ARE THE 2 SEVEN SEGMENT DISPLAYS, THE 4X4 LED MATRIX AND THE 8 DATA LINE LEDS. THEY ARE ALL WIRED TOGETHER AND TO THE SAME OUTPUT PORT. THEY ARE ALSO WIRED DIFFERENTLY TO EACH OTHER.

THE OUTPUT LATCH DRIVES THE 2 SEVEN SEGMENT DISPLAYS, THE 4X4 LED MATRIX AND THE 8 DATA LINE LEDS. YOU CAN'T CONTROL THESE INDIVIDUALLY AS THEY ARE ALL CONNECTED TOGETHER TO THE ONE LATCH. PROGRAMS FOR THE 4X4 MATRIX WILL STILL DISPLAY ON THE SEVEN SEGMENT DISPLAYS BUT WILL BE MEANINGLESS, AND VICE VERSA. TO SEND DATA TO THE OUTPUT, USE "OUT (02),A". BIT 7 OF REGISTER A IS A SPECIAL CASE WHERE IF SET WILL ACTIVATE THE LEFT SEVEN SEGMENT DISPLAY AND IF NOT SET WILL ACTIVATE THE RIGHT SEVEN SEGMENT DISPLAY. THE OTHER BITS ARE USED TO LIGHT UP THE INDIVIDUAL SEGMENTS. THE 4X4 MATRIX AND 8 DATA LINE LEDS USE BIT 7 AS NORMAL.

CODING TIPS

IF DEVELOPING YOUR OWN PROGRAMS, SOME CONSIDERATIONS ARE NEEDED. FIRSTLY AS THERE IS NO RAM, THE STACK CAN'T BE USED. YOU CAN'T USE ANY COMMANDS THAT UTILISE THE STACK, IE: PUSH, POP, CALL, RET, RST AND SOME SP OP CODES. ALSO, THE PROGRAM MUST USE A JUMP AT THE END TO CONTINUALLY LOOP IT. IF YOU REQUIRE A ROUTINE TO BE 'CALLED' AND RETURNED, USE JP (HL), OR JP (IX) WHERE YOU CAN SET HL/IX TO THE RETURN ADDRESS: IE:

```
    CALL PROG   ;JUMP TO PROG:
CONT:
    ...

PROG:
    ...       ;DO SOMETHING
    RET       ;JUMP BACK TO CONT:
```

IS THE SAME AS

```
    LD HL,CONT
    JP PROG
CONT:
    ...

PROG:
    ...       ;DO SOMETHING
    JP (HL)   ;JUMP BACK TO CONT:
```

YOU CAN USE THE SP REGISTER FOR STORAGE ONLY.

LIST OF PROGRAMS AND TABLES ON THE ROM

ADDR	DIP	TYPE	NAME
0000	00	P	JUMP ROUTINE
0010	01	P	TONE
0020	02	P	QUICK DRAW
0080	08	P	RUNNING NAMES
00D0	0D	U	RUNNING LETTER ROUTINE
00F5	--	D	QUICK DRAW ANIMATION TABLE
0100	--	D	LIST OF NAMES TABLE
0200	20	P	LOOKING AT DATA
0290	29	P	FROM INPUT TO 8 LEDS
02A0	2A	P	INCREMENT VIA BUTTON A
02C0	2C	P	AUTO INCREMENT (FAST)
02D0	2D	P	AUTO INCREMENT (VARIABLE)
02E0	2E	P	AUTO DECREMENT
02F0	2F	P	AUTO DECREMENT (VARIABLE)
0300	30	P	4X4 LED EFFECTS
0370	37	P	0 - 9 COUNTER
0390	39	P	0 - F COUNTER
03A0	3A	P	A - Z, 0 - F COUNTER
03F0	--	U	VERY LONG DELAY (PART 1)
0400	40	P	00 - 99 COUNTER
045A	--	U	VERY LONG DELAY (PART 2)
0470	47	P	DICE
0520	52	P	EPROM IN BINARY
0530	53	P	POKER
0630	63	P	BINARY CLOCK
06C0	6C	P	ONE MINUTE TIMER
06D0	6D	P	3 MINUTE TIMER
06E0	6E	P	1 HOUR TIMER
06F0	6F	P	ADJUSTABLE TIMER
0740	--	U	1 MINUTE DELAY
0765	--	D	ADJUSTABLE TIMER DATA TABLE
07A0	7A	P	FINAL MESSAGE

KEY:

ADDR = ADDRESS LOCATION ON ROM
DIP = INPUT LATCH DIP SETTING
TYPE = P - PROGRAM TO RUN, U - UTILITY, D - DATA TABLE LOOKUP
NAME = NAME OF PROGRAM OR TABLE

START OF LISTING

JUMP PROGRAM ROUTINE IS USED EVERY TIME YOU WANT TO ACCESS ONE OF THE PROGRAMS. WHEN AN ADDRESS IS SET ON THE INPUT LATCH AND RESET IS PRESSED IT WILL SHIFT THE ADDRESS TO THE LEFT BY ONE BYTE AND JUMP TO THAT LOCATION. IE: IF 0X02 IS ON THE LATCH, IT WILL JUMP TO LOCATION 0X0020, AND IF 0X47 IS ON THE LATCH IT WILL JUMP TO 0X0470. WITH THIS IN MIND, THE START LOCATION OF EACH PROGRAM MUST HAVE ITS LAST BYTE AS ZERO. IF THE ROM ONLY HAS ONE PROGRAM THEN THIS ROUTINE ISN'T NEEDED. (SEE THE ADDENDUM FOR THE ALTERNATE JUMP PROGRAM)

```
0000 06 00          LD B,00       ;RESET B TO ZERO, TO BE USED AFTER THE JUMP
0002 DB 01          IN A,(01)     ;READ THE INPUT LATCH
0004 21 00 00       LD HL,0000    ;RESET HL
0007 6F             LD L,A        ;LOAD THE BITS SET ON THE LATCH TO L
0008 29             ADD HL,HL     ;MULTIPLY HL BY 10 TO SHIFT THE FOUR BYTE
0009 29             ADD HL,HL     ;REGISTER HL TO THE LEFT BY ONE BYTE
000A 29             ADD HL,HL     ;IE: IF HL = 0X0047 IT WILL NOW BE 0X0470
000B 29             ADD HL,HL     ;
000C E9             JP (HL)       ;JUMP TO THE ADDRESS ON HL
000D 00 00 00                     ;FILL
```

TONE ROUTINE TO OSCILLATE D7 OR TERMINAL 80. THIS IS USED TO TEST THE PROBE. WHEN USED WITH THE 'PROBE', TOUCHING TERMINAL 80 WILL PRODUCE A SOUND THAT WILL CHANGE FREQUENCY WHEN THE CLOCK SPEED IS MODIFIED. IT ALSO LIGHTS UP SEGMENT 'A' ON THE LEFT DISPLAY. NOTE: WHEN D7 IS SET THE LEFT DISPLAY WILL ACTIVATE, WHEN NOT SET THE RIGHT DISPLAY WILL ACTIVATE.

```
0010 AF             XOR A         ;RESET A TO ZERO
0011 D3 02          OUT (02),A    ;BLANK ALL DISPLAYS
0013 3E 81          LD A,81       ;LOAD 0X81 TO A (SETS D0 AND D7 TO HIGH)
0015 D3 02          OUT (02),A    ;OUTPUT A TO THE DISPLAYS
0017 AF             XOR A         ;RESET A TO ZERO
0018 D3 02          OUT (02),A    ;BLANK ALL DISPLAYS
001A 3E 81          LD A,81       ;LOAD 0X81 TO A (SETS D0 AND D7 TO HIGH)
001C D3 02          OUT (02),A    ;OUTPUT A TO THE DISPLAYS
001E 18 F0          JR 0010       ;JUMP TO START OF TONE ROUTINE
```

QUICK DRAW ROUTINE IS A REACTION GAME FOR TWO PLAYERS. WHEN THE SEVEN SEGMENTS DISPLAY C AND BACK C, THE FIRST PERSON TO PRESS THEIR BUTTON (A OR B) WINS. PRESS RESET TO START AGAIN. THIS PROGRAM IS SPLIT INTO THREE PARTS, AN ANIMATION, A DELAY WITH BLANK SCREEN AND THE REACTION SCREEN.

```
0020 0E 02          LD C,02       ;ANIMATE THE SEGMENTS TWICE
0022 16 08          LD D,08       ;EIGHT SEGMENTS SEQUENCE
0024 21 F5 00       LD HL,00F5    ;SEGMENT LOOK UP TABLE
0027 7E             LD A,(HL)     ;LOAD SEGMENT DATA TO A
0028 D3 02          OUT (02),A    ;OUTPUT TO SEVEN SEGMENTS
002A 10 FE          DJNZ 002A     ;SHORT DELAY
002C 23             INC HL        ;MOVE TO NEXT DATA
002D 15             DEC D         ;DECREASE SEQUENCE LEFT
002E 20 F7          JR NZ,0027    ;DISPLAY THE NEXT SEGMENT UNTIL ALL DONE
0030 0D             DEC C         ;DECREASE REPEAT SEGMENT
0031 20 EF          JR NZ,0022    ;DO SEGMENT LOOP TWICE
0033 3E 00          LD A,00       ;RESET A TO ZERO
0035 D3 02          OUT (02),A    ;BLANK OUTPUT
0037 11 02 06       LD DE,0602    ;LOAD DE WITH DELAY FOR BLANK SCREEN
003A 1B             DEC DE        ;DECREASE DELAY COUNT
003B 7A             LD A,D        ;LOAD D WITH A
```

```
003C B3              OR E          ;COMPARE A WITH E
003D 20 FB           JR NZ,003A    ;IF D AND E DOESN'T EQUAL ZERO DECREASE AGAIN
003F DB 01           IN A,(01)     ;CHECK IF A BUTTON HAS BEEN PRESSED TOO EARLY
0041 CB 77           BIT 6,A       ;IS BUTTON B PRESSED?
0043 C2 20 00        JP NZ,0020    ;RESTART GAME IF PRESSED
0046 CB 7F           BIT 7,A       ;IS BUTTON A PRESSED?
0048 C2 20 00        JP NZ,0020    ;RESTART GAME IF PRESSED
004B 3E 0F           LD A,0F       ;LOAD A WITH A BACKWARD 'C' FOR DISPLAY
004D D3 02           OUT (02),A    ;OUTPUT TO SEVEN SEGMENTS
004F 06 08           LD B,08       ;LOAD B WITH SHORT DELAY
0051 10 FE           DJNZ 0051     ;DELAY
0053 3E B9           LD A,B9       ;LOAD A WITH A 'C' FOR DISPLAY
0055 D3 02           OUT (02),A    ;OUTPUT TO SEVEN SEGMENTS
0057 DB 01           IN A,(01)     ;CHECK FOR BUTTON INPUT
0059 CB 77           BIT 6,A       ;HAS BUTTON B BEEN PRESSED?
005B 20 09           JR NZ,0066    ;JUMP IF IT HAS
005D CB 7F           BIT 7,A       ;HAS BUTTON A BEEN PRESSED?
005F 28 EA           JR Z,004B     ;IF NO BUTTON PRESSED JUMP TO LOOP DISPLAY
0061 3E B0           LD A,B0       ;LOAD A WITH A '1' FOR DISPLAY ON LEFT
0063 D3 02           OUT (02),A    ;OUTPUT TO LEFT SEVEN SEGMENT, A WINS
0065 76              HALT          ;HALT CPU
0066 CB 7F           BIT 7,A       ;CHECK IF BUTTON A WAS ALSO PRESSED FOR DRAW
0068 28 0A           JR Z,0074     ;NO DRAW, JUMP TO PLAYER B AS WINNER
006A 3E 06           LD A,06       ;DRAW, LOAD A WITH A '1' FOR DISPLAY ON RIGHT
006C D3 02           OUT (02),A    ;OUTPUT TO SEVEN SEGMENT ON RIGHT
006E 3E B0           LD A,B0       ;LOAD A WITH A '1' FOR DISPLAY ON LEFT
0070 D3 02           OUT (02),A    ;OUTPUT TO SEVEN SEGMENT ON RIGHT
0072 18 F6           JR 006A       ;REPEAT DRAW OUTPUT
0074 3E 06           LD A,06       ;LOAD A WITH A '1' FOR DISPLAY ON RIGHT
0076 D3 02           OUT (02),A    ;OUTPUT TO SEVEN SEGMENT ON RIGHT, B WINS
0078 76              HALT          ;HALT CPU
0079 00 00 00 00 00 00 00          ;FILL
```

RUNNING NAMES ROUTINE. THIS PROGRAM SCROLLS TEXT ACROSS THE TWO SEVEN SEGMENTS DISPLAYS. IT HAS THREE COMPONENTS, IN INTRO TEXT, A NAME WHICH IS USER DEFINED AND A COPYRIGHT TEXT. IT WORKS BY POINTING TO A ASCII DATA TABLE, AND CALLING SCROLL ROUTINE THAT MULTIPLEXES THE DISPLAY. WHEN AN 'FF' IS REACHED THE PROGRAM GOES TO THE NEXT COMPONENT.

```
0080 DD 21 00 01     LD IX,0100    ;LOOKUP TABLE FOR INTRO TEXT
0084 21 8A 00        LD HL,008A    ;STORE THE RETURN ADDRESS 008A IN HL
0087 C3 D0 00        JP 00D0       ;JUMP TO RUNNING LETTER ROUTINE
008A 0E 00           LD C,00       ;LOAD C WITH ZERO FOR NAME INDEX COUNTER
008C DD 21 14 01     LD IX,0114    ;LOAD IX WITH POSITION OF FIRST NAME IN TABLE
0090 DB 01           IN A,(01)     ;CHECK INPUT LATCH
0092 FE 00           CP 00         ;IF ITS ZERO
0094 28 13           JR Z,00A9     ;SKIP INDEXING AND DISPLAY FIRST NAME
0096 57              LD D,A        ;SAVE INPUT DATA IN D
0097 DD 7E 00        LD A,(IX+0)   ;LOAD A WITH DATA AT IX
009A FE FF           CP FF         ;IF IT'S FF, THEN END NAME FOUND
009C 28 04           JR Z,00A2     ;JUMP TO COMPARE INDEX TO INPUT
009E DD 23           INC IX        ;NO END OF NAME FOUND, MOVE TO NEXT CHARACTER
00A0 18 F5           JR 0097       ;LOOP UNTIL END OF NAME FOUND
00A2 0C              INC C         ;INCREASE NAME INDEX
00A3 79              LD A,C        ;LOAD A INTO C
00A4 BA              CP D          ;COMPARE INDEX TO INPUT LATCH
00A5 20 F7           JR NZ,009E    ;IF DIFFERENT, LOOP TO CHECK THE NEXT NAME
00A7 18 02           JR 00AB       ;INDEX FOUND, JUMP TO DISPLAY NAME
00A9 DD 2B           DEC IX        ;BACK IX IF INPUT IS ZERO
```

```
00AB 21 B3 00       LD HL,00B3    ;STORE THE RETURN ADDRESS 00B3 IN HL
00AE DD 23          INC IX        ;MOVE TO FIRST CHARACTER AS IX IS ON 'FF'
00B0 C3 D0 00       JP 00D0       ;JUMP TO RUNNING LETTER ROUTINE
00B3 0E 08          LD C,08       ;LOAD C WITH 8 TO REPEAT 8 TIMES
00B5 3E 58          LD A,58       ;LOAD A WITH SMALL 'c' FOR COPYRIGHT
00B7 D3 02          OUT (02),A    ;OUTPUT SEVEN SEGMENT
00B9 10 FE          DJNZ 00B9     ;DELAY
00BB 3E 00          LD A,00       ;LOAD A WITH ZERO TO BLANK SCREEN
00BD D3 02          OUT (02),A    ;OUTPUT SEVEN SEGMENT
00BF 10 FE          DJNZ 00BF     ;DELAY
00C1 0D             DEC C         ;DECREASE C
00C2 20 F1          JR NZ,00B5    ;IF C ISN'T ZERO, LOOP COPYRIGHT OUTPUT
00C4 DD 21 F8 01    LD IX,01F8    ;LOAD IX WITH LOOKUP TABLE FOR 1985 DATE
00C8 21 80 00       LD HL,0080    ;STORE THE RETURN ADDRESS 0080 IN HL
00CB C3 D0 00       JP 00D0       ;JUMP TO RUNNING LETTER ROUTINE
00CE 00 00                        ;FILL
```

RUNNING LETTER ROUTINE. WILL SCROLL TEXT ACROSS THE TWO SEVEN SEGMENTS UNTIL AN 'FF' IS FOUND. REQUIRES IX TO POINT TO DATA TABLE AND HL TO STORE THE RETURN ADDRESS.

```
00D0 0E 0B          LD C,0B       ;EACH LETTER APPEARS 11 TIMES
00D2 DD 7E 00       LD A,(IX+0)   ;LOAD A WITH FIRST LETTER
00D5 CB FF          SET 7,A       ;SET BIT 7 TO DISPLAY ON LEFT SEGMENT
00D7 D3 02          OUT (02),A    ;OUTPUT TO LEFT SEGMENT
00D9 06 20          LD B,20       ;LOAD B WITH DELAY
00DB 10 FE          DJNZ 00DB     ;DELAY
00DD DD 7E 01       LD A,(IX+1)   ;LOAD A WITH SECOND LETTER
00E0 D3 02          OUT (02),A    ;OUTPUT TO RIGHT SEGMENT
00E2 06 20          LD B,20       ;LOAD B WITH DELAY
00E4 10 FE          DJNZ 00E4     ;DELAY
00E6 0D             DEC C         ;DECREASE COUNTER
00E7 20 E9          JR NZ,00D2    ;REDISPLAY TO TWO LETTERS UNTIL C IS ZERO
00E9 DD 23          INC IX        ;MOVE TO NEXT LETTER IN TABLE
00EB 0E 0C          LD C,0C       ;LOAD C WITH 12 (NO NEEDED IF JUMPED TO D0?)
00ED DD 7E 01       LD A,(IX+1)   ;CHECK FOR NEXT LETTER
00F0 FE FF          CP FF         ;IS IT 'FF'?
00F2 20 DE          JR NZ,00D2    ;NO THEN REPEAT SCROLL
00F4 E9             JP (HL)       ;NO MORE LETTERS JUMP BACK TO CALLING ROUTINE
```

SEGMENT LOOKUP TABLE FOR QUICKDRAW. THE EIGHT BYTES WHEN OUTPUTTED TO THE SEVEN SEGMENT DISPLAYS WILL LIGHT UP AROUND THE TWO SEVEN SEGMENT DISPLAYS

```
00F5 01 02 04 08 88 90 A0 81      ;OUTER SEQUENTIAL SEGMENTS
00FD 00 00 00                     ;FILL
```

RUNNING NAMES LOOKUP TABLE. CONTAINS THE INTRO MESSAGE, NAME TABLE AND COPYRIGHT DATE. DATA IS FOR RIGHT SEGMENT, BIT 7 IS SET TO DISPLAY ON LEFT SEGMENT. SEGMENTS ON DISPLAY IS ASSUMED AS:
A = 01, B = 02, C = 04, D = 08, E = 10, F = 20, G = 40
A COMBINATION OF THESE BITS SET WILL DISPLAY THE DESIRED CHARACTER. AN 'FF' REPRESENTS THE END OF THE WORD(S)

```
0100 4F 40 39 76 06 73 00 1C      ;3-CHIP_uP_BUILT_BY_
0108 73 00 7C 3E 06 38 78 00      ;
0110 7C 6E 00 FF                  ;

0114 77 37 5E 6E FF               ;ANDY       INDEX=0
0119 7C 77 6D 06 38 FF            ;BASIL
```

```
011F 7C 79 33 78 FF              ;BERT
0124 7C 06 38 38 FF              ;BILL
0129 7C 3F 7C FF                 ;BOB
012D 7C 33 3E 39 79 FF           ;BRUCE
0133 39 77 33 38 FF              ;CARL
0138 39 76 77 33 38 79 6D FF     ;CHARLES
0140 00 00 79 37 78 79 33 00     ;__ENTER_   INDEX=8 (INITIAL MESSAGE)
0148 06 37 73 3E 78 00 1C 77     ;INPUT_VA
0150 38 3E 79 00 00 00 00 00 FF  ;LUE_____
0159 39 38 06 71 71 FF           ;CLIFF
015F 39 38 06 3E 79 FF           ;CLIVE
0165 39 33 06 6D FF              ;CRIS
016A 39 3F 38 06 37 FF           ;COLIN
0170 39 33 77 06 3D FF           ;CRAIG
0176 5E 77 3E 06 5E FF           ;DAVID
017C 5E 3F 3E 3D FF              ;DOUG
0181 79 5E FF                    ;ED
0184 79 3E 77 37 FF              ;EVAN
0189 3D 79 3F 33 3D 79 FF        ;GEORGE
0190 3D 38 79 37 FF              ;GLEN
0195 3D 33 79 3D FF              ;GREG
019A 06 77 37 FF                 ;IAN
019E 1E 3F 76 37 FF              ;JOHN
01A3 73 77 78 FF                 ;PAT
01A7 73 79 78 79 33 FF           ;PETER
01AD 73 76 06 38 06 73 FF        ;PHILIP
01B4 33 77 38 73 76 FF           ;RALPH
01BA 33 3F 6E FF                 ;ROY
01BE 6D 39 3F 78 78 FF           ;SCOTT
01C4 6D 78 77 37 FF              ;STAN
01C9 78 3F 37 6E FF              ;TONY
01CE 38 06 78 78 38 79 00 3F     ;LITTLE_OL_I
01D6 38 00 06 FF                 ;
01DA 53 53 53 FF                 ;???
01DE 40 40 40 3D 3E 79 6D 6D     ;---GUESS---
01E6 40 40 40 FF                 ;
01EA 77 37 00 3F 38 5E 00 73     ;AN_OLD_PRO  INDEX=22
01F2 33 3F 00                    ;(SHOULD HAVE FF INSTEAD OF 00!)
01F5 00 00 00                    ;FILL
01F8 06 6F 7F 6D 00 00 FF        ;1985__
01FF 00                          ;FILL
```

LOOKING AT DATA PROGRAM. THIS IS A CLEVER PROGRAM THAT DISPLAYS THE CONTENT OF THE EPROM. THE FIRST VALUE DISPLAYED IS THE ADDRESS LOCATION BASED ON THE PAGE IT IS VIEWING AND THE SECOND VALUE IS THE DATA. TO ADVANCE THE ADDRESS PRESS BUTTON 'A', IT WILL DISPLAY THE ADDRESS LOCATION THEN THE DATA. TO JUMP 8 BYTES FORWARD PRESS BUTTON 'B' WHEN VIEWING THE DATA. TO DISTINGUISH BETWEEN DATA AND ADDRESS, THE ADDRESS VALUE WILL BE 'DULLER' IN APPEARANCE. THIS DULLNESS IS ACHIEVED BY CREATING A LONGER DELAY IN THE MULTIPLEXING. THE PAGES CAN BE SELECTED BY CHANGING THE INPUT LATCH BETWEEN 00 AND 07. 00 TO FF BYTES ARE SHOWN PER PAGE AND THIS WILL LOOP. HL STORES THE 0-F ASCII TABLE ADDRESS, DE STORES THE CURRENT ADDRESS BEING EXAMINED.

```
0200 0E 00          LD C,00       ;C IS USED TO CHECK IF BUTTON 'A' IS PRESSED
0202 1E 00          LD E,00       ;DEFAULT START LSB ADDRESS LOCATION
0204 DB 01          IN A,(01)     ;READ INPUT LATCH FOR PAGE SETTING
0206 E6 07          AND 07        ;ONLY BITS 0,1,2 ARE USED
0208 57             LD D,A        ;SAVE MSB ADDRESS IN D, DE STORES CURRENT ADR
0209 7B             LD A,E        ;LOAD LSB ADDRESS IN A
```

```
020A E6 0F          AND 0F        ;MASK OFF HIGH NIBBLE FOR RIGHT DISPLAY
020C 21 80 02       LD HL,0280    ;POINT HL TO 0-F ASCII LOOKUP TABLE
020F 85             ADD A,L       ;INDEX L BASED ON
0210 6F             LD L,A        ;VALUE OF A
0211 3E 00          LD A,00       ;SET A TO ZERO
0213 D3 02          OUT (02),A    ;BLANK THE SEGMENTS
0215 06 10          LD B,10       ;LOAD 10 TO B FOR
0217 10 FE          DJNZ 0217     ;DELAY TO DULL DISPLAY
0219 7E             LD A,(HL)     ;LOAD ASCII VALUE TO A FOR DISPLAY
021A D3 02          OUT (02),A    ;OUTPUT LOW NIBBLE ON RIGHT DISPLAY
021C 7B             LD A,E        ;RELOAD A WITH LSB ADDRESS
021D 1F             RRA           ;SHIFT A FOUR TIMES TO SWAP LOWER NIBBLE WITH
021E 1F             RRA           ;UPPER NIBBLE
021F 1F             RRA           ;
0220 1F             RRA           ;
0221 E6 0F          AND 0F        ;MASK OFF HIGH NIBBLE FOR LEFT DISPLAY
0223 21 80 02       LD HL,0280    ;POINT HL TO 0-F ASCII LOOKUP TABLE
0226 85             ADD A,L       ;INDEX L BASED ON
0227 6F             LD L,A        ;VALUE OF A
0228 3E 00          LD A,00       ;SET A TO ZERO
022A D3 02          OUT (02),A    ;BLANK THE SEGMENTS
022C 06 10          LD B,10       ;LOAD 10 TO B FOR
022E 10 FE          DJNZ 022E     ;DELAY
0230 7E             LD A,(HL)     ;LOAD ASCII VALUE TO A FOR DISPLAY
0231 CB FF          SET 7,A       ;SET BIT 7 ON A TO DISPLAY ON LEFT DISPLAY
0233 D3 02          OUT (02),A    ;OUTPUT HIGH NIBBLE ON LEFT DISPLAY
0235 DB 01          IN A,(01)     ;READ INPUT LATCH
0237 CB 7F          BIT 7,A       ;HAS BUTTON 'A' BEEN PRESSED?
0239 28 08          JR Z,0243     ;JUMP IF NOT PRESSED
023B CB C9          SET 1,C       ;SET BIT 1 TO INDICATE BUTTON 'A' PRESSED
023D CB 51          BIT 2,C       ;CHECK BIT 2 SET (ADDRESS OR DATA DISPLAY)
023F 20 C3          JR NZ,0204    ;IF SET REPEAT ADDRESS DISPLAY
0241 18 0C          JR 024F       ;JUMP TO DATA DISPLAY (SHOULD JMP TO 024E!)
0243 CB 91          RES 2,C       ;RESET BIT 2
0245 CB 77          BIT 6,A       ;CHECK IF BUTTON 'B' PRESSED
0247 28 BB          JR Z,0204     ;IF NOT PRESSED REPEAT ADDRESS DISPLAY
0249 1C             INC E         ;MOVE TO NEXT ADDRESS
024A 00 00                        ;FILL
024C 18 B6          JR 0204       ;REPEAT ADDRESS DISPLAY
024E 21 80 02       LD HL,0280    ;POINT HL TO 0-F ASCII LOOKUP TABLE
0251 1A             LD A,(DE)     ;LOAD A WITH THE DATA POINTING TO DE
0252 E6 0F          AND 0F        ;MASK OFF HIGH NIBBLE FOR RIGHT DISPLAY
0254 85             ADD A,L       ;INDEX L BASED ON
0255 6F             LD L,A        ;VALUE OF A
0256 7E             LD A,(HL)     ;LOAD ASCII VALUE TO A FOR DISPLAY
0257 D3 02          OUT (02),A    ;OUTPUT LOW NIBBLE ON RIGHT DISPLAY
0259 1A             LD A,(DE)     ;RELOAD A WITH DATA POINTING TO DE
025A 1F             RRA           ;SHIFT A FOUR TIMES TO SWAP LOWER NIBBLE WITH
025B 1F             RRA           ;UPPER NIBBLE
025C 1F             RRA           ;
025D 1F             RRA           ;
025E E6 0F          AND 0F        ;MASK OFF HIGH NIBBLE FOR LEFT DISPLAY
0260 21 80 02       LD HL,0280    ;POINT HL TO 0-F ASCII LOOKUP TABLE
0263 85             ADD A,L       ;INDEX L BASED ON
0264 6F             LD L,A        ;VALUE OF A
0265 7E             LD A,(HL)     ;LOAD ASCII VALUE TO A FOR DISPLAY
0266 CB FF          SET 7,A       ;SET BIT 7 ON A TO DISPLAY ON LEFT DISPLAY
0268 D3 02          OUT (02),A    ;OUTPUT HIGH NIBBLE ON LEFT DISPLAY
026A DB 01          IN A,(01)     ;READ INPUT LATCH
```

```
026C CB 7F          BIT 7,A       ;HAS BUTTON 'A' BEEN PRESSED?
026E 28 0B          JR Z,027B     ;NO, THEN JUMP
0270 CB D1          SET 2,C       ;SET BIT 2 TO INDICATE 'DATA' DISPLAY
0272 CB 49          BIT 1,C       ;CHECK IF BIT 1 (BUTTON 'A') IS PRESSED
0274 20 D8          JR NZ,024E    ;REPEAT DATA DISPLAY
0276 1C             INC E         ;INCREMENT ADDRESS
0277 00 00                        ;FILL
0279 18 89          JR 0204       ;JUMP TO ADDRESS DISPLAY
027B CB 89          RES 1,C       ;RESET BIT 1 AS BUTTON 'A' WASN'T PRESSED
027D 18 CF          JR 024E       ;REPEAT DATA DISPLAY
027F 00                           ;FILL

LOOKUP TABLE FOR LOOK AT DATA SEVEN SEGMENT DISPLAY
0280 3F 06 5B 4F 66 6D 7D 07      ;0-7
0288 7F 67 77 7C 39 5E 79 71      ;8-F

FROM INPUT LATCH TO 8 LED ROUTINE.  THIS SIMPLY OUTPUTS WHAT IN ON THE INPUT
LATCH TO THE OUTPUT LATCH.  ITS A GOOD WAY TO CHECK WHAT VALUES ARE NEEDED TO
DRIVE THE OUTPUT CORRECTLY.  IE: HOW TO LIGHT UP THE 4X4, OR SEVEN SEGMENT
DISPLAY TO THE WAY YOU WANT IT.  LATCH IS COPIED TO THE 8 BIT LEDS TOO.

0290 DB 01          IN A,(01)     ;READ INPUT LATCH TO A
0292 D3 02          OUT (02),A    ;SEND A TO OUTPUT DISPLAYS
0294 18 FA          JR 0290       ;REPEAT FROM START

0296 00 00 00 00 00 00 00 00      ;FILL
029E 00 00                        ;FILL

INCREMENT VIA BUTTON A ROUTINE.  THIS OUTPUTS THE BYTE VALUE FROM 00 TO FF
ON THE DISPLAYS.  BUTTON 'A' INCREMENTS THE COUNTER.  AS EACH DISPLAY WILL SHOW
ITS UNIQUE WAY OF DISPLAYING THE VALUE.

02A0 3E 00          LD A,00       ;RESET A TO ZERO
02A2 4F             LD C,A        ;STORE A IN C AS THE CURRENT BYTE COUNT
02A3 DB 01          IN A,(01)     ;CHECK THE INPUT LATCH
02A5 CB 7F          BIT 7,A       ;FOR BUTTON 'A' PRESSED
02A7 28 FA          JR Z,02A3     ;IF NOT PRESSED, REPEAT INPUT LATCH CHECK
02A9 79             LD A,C        ;LOAD CURRENT COUNT TO A
02AA 3C             INC A         ;INCREMENT A
02AB 4F             LD C,A        ;STORE CURRENT COUNT BACK TO C
02AC D3 02          OUT (02),A    ;OUTPUT BYTE VALUE
02AE DB 01          IN A,(01)     ;CHECK FOR INPUT LATCH
02B0 CB 7F          BIT 7,A       ;FOR BUTTON 'A' PRESSED
02B2 20 FA          JR NZ,02AE    ;IF STILL PRESSED, REPEAT INPUT LATCH CHECK
02B4 18 ED          JR 02A3       ;IF RELEASED, GO TO START

02B6 00 00 00 00 00 00 00 00      ;FILL
02BE 00 00                        ;FILL

AUTO INCREMENT (FAST) ROUTINE IS THE SAME AS THE ABOVE PROGRAM BUT THE
INCREMENT IS AUTOMATIC.  THREE FULL 8 BIT DELAYS ARE USED TO SLOW THE UPDATE.

02C0 3E 00          LD A,00       ;RESET A TO ZERO FOR INITIAL COUNT
02C2 3C             INC A         ;INCREMENT A
02C3 D3 02          OUT (02),A    ;OUTPUT BYTE VALUE
02C5 10 FE          DJNZ 02C5     ;DELAY
02C7 10 FE          DJNZ 02C7     ;DELAY
02C9 10 FE          DJNZ 02C9     ;DELAY
02CB 18 F5          JR 02C2       ;JUMP BACK TO NEXT INCREMENT
```

```
02CD 00 00 00                    ;FILL
```

AUTO INCREMENT (VARIABLE) ROUTINE IS THE SAME AS THE ABOVE ROUTINE BUT THE DELAY IS SET BY THE VALUE ON THE INPUT LATCH.

```
02D0 16 01          LD D,01       ;LOAD D WITH INTIAL BYTE VALUE
02D2 DB 01          IN A,(01)     ;READ INPUT LATCH FOR DELAY VALUE
02D4 4F             LD C,A        ;STORE DELAY VALUE IN C
02D5 7A             LD A,D        ;LOAD A WITH BYTE VALUE
02D6 D3 02          OUT (02),A    ;OUTPUT BYTE VALUE
02D8 0D             DEC C         ;DECREASE DELAY
02D9 20 FD          JR NZ,02D8    ;REPEAT DELAY IF NOT ZERO
02DB 14             INC D         ;INCREMENT BYTE VALUE
02DC 18 F4          JR 02D2       ;JUMP BACK TO START

02DE 00 00                        ;FILL
```

AUTO DECREMENT ROUTINE IS THE OPPOSITE TO THE AUTO INCREMENT ROUTINE EXCEPT THAT VALUES DISPLAYED ARE DECREMENTED.

```
02E0 3E 00          LD A,00       ;RESET A TO ZERO FOR INITIAL COUNT
02E2 3D             DEC A         ;DECREMENT A
02E3 D3 02          OUT (02),A    ;OUTPUT BYTE VALUE
02E5 10 FE          DJNZ 02E5     ;DELAY
02E7 10 FE          DJNZ 02E7     ;DELAY
02E9 10 FE          DJNZ 02E9     ;DELAY
02EB 18 F5          JR 02E2       ;JUMP BACK TO NEXT DECREMENT

02ED 00 00 00                     ;FILL
```

AUTO DECREMENT (VARIABLE) IS NOT ACTUALLY VARIABLE! IT DECREMENTS THE DISPLAY BYTE WHEN BUTTON 'A' IS PRESSED. FOR SOME REASON THEY CALL IT VARIABLE IN THE MAGAZINE!

```
02F0 1E FF          LD E,FF       ;SET E TO FF
02F2 7B             LD A,E        ;LOAD COUNTER TO A
02F3 D3 02          OUT (02),A    ;OUTPUT BYTE VALUE
02F5 10 FE          DJNZ 02F5     ;DELAY
02F7 DB 01          IN A,(01)     ;CHECK INPUT LATCH
02F9 CB 7F          BIT 7,A       ;HAS BUTTON 'A' BEEN PRESSED?
02FB 28 F5          JR Z,02F2     ;NO, JUMP BACK TO DISPLAY
02FD 1D             DEC E         ;YES, DECREASE E
02FE 18 F2          JR 02F2       ;JUMP BACK TO DISPLAY
```

4X4 LED EFFECTS ROUTINE. THIS PROGRAM CYCLES THROUGH TWO SEQUENCES OF 4X4 LED PATTERNS. IT WILL PRODUCE ALMOST NO INTERPRETABLE EFFECTS ON EITHER OF THE OTHER DISPLAYS. BOTH SEQUENCES USE SIMILAR CODE, JUST POINT TO DIFFERENT TABLES AND USE DIFFERENT LENGTHS.
THE MAGAZINE TALKS BIG ABOUT THE USES OF THE 4X4. HERE IS AN EXCERPT...
"OUR 4X4 CAN BE MULTIPLIED-UP MANY TIMES TO PRODUCE AN ENORMOUS ARRAY OF LEDS OR GLOBES AND OBVIOUSLY THE ULTIMATE IS TO PRODUCE A VIDEO SCREEN WITH COLOURED GLOBES TO DUPLICATE A TV. BUT THE COST OF THIS KIND OF VENTURE IS ENORMOUS AS THE PARTS ALONE WOULD COST A FORTUNE AND THE TIME TAKEN TO WIRE IT UP WOULD BE TOO MUCH FOR AN INDIVIDUAL CONSTRUCTOR."

```
                                  ;FIRST SEQUENCE
0300 06 08          LD B,08       ;DO THE FULL SEQUENCE 8 TIMES
0302 21 38 03       LD HL,0338    ;LOAD HL WITH 4X4 DATA TABLE
```

```
0305 0E 18          LD C,18       ;LOAD C WITH DATA TABLE SIZE
0307 0D             DEC C         ;DECREASE C
0308 28 0E          JR Z,0318     ;IF ZERO EXIT TO REPEAT ROUTINE
030A 7E             LD A,(HL)     ;LOAD A WITH SEQUENCE VALUE
030B D3 02          OUT (02),A    ;OUTPUT TO 4X4 DISPLAYS
030D 23             INC HL        ;MOVE TO NEXT SEQUENCE VALUE
030E 11 80 00       LD DE,0080    ;DO A
0311 1B             DEC DE        ;SMALL
0312 7A             LD A,D        ;DELAY
0313 B3             OR E          ;BEFORE DISPLAYING THE
0314 20 FB          JR NZ,0311    ;NEXT VALUE
0316 18 EF          JR 0307       ;REPEAT FOR NEXT VALUE
0318 10 E8          DJNZ 0302     ;REPEAT TOTAL SEQUENCE EIGHT TIMES

                                  ;SECOND SEQUENCE
031A 06 08          LD B,08       ;DO THE FULL SEQUENCE 8 TIMES
031C 21 50 03       LD HL,0350    ;LOAD HL WITH 4X4 DATA TABLE
031F 0E 20          LD C,20       ;LOAD C WITH DATA TABLE SIZE
0321 0D             DEC C         ;DECREASE C
0322 28 0E          JR Z,0332     ;IF ZERO EXIT TO REPEAT ROUTINE
0324 7E             LD A,(HL)     ;LOAD A WITH SEQUENCE VALUE
0325 D3 02          OUT (02),A    ;OUTPUT TO 4X4 DISPLAYS
0327 23             INC HL        ;MOVE TO NEXT SEQUENCE VALUE
0328 11 80 00       LD DE,0080    ;DO A
032B 1B             DEC DE        ;SMALL
032C 7A             LD A,D        ;DELAY
032D B3             OR E          ;BEFORE DISPLAYING THE
032E 20 FB          JR NZ,032B    ;NEXT VALUE
0330 18 EF          JR 0321       ;REPEAT FOR NEXT VALUE
0332 10 E8          DJNZ 031C     ;REPEAT TOTAL SEQUENCE EIGHT TIMES
0334 18 CA          JR 0300       ;GO BACK TO THE START AGAIN.

0336 FE 10                        ;BYTES NO USED!

DATA TABLE FOR 4X4 LED SEQUENCE
0338 01 02 04 08 EF DF BF 7F      ;FIRST 4X4 LED SEQUENCE
0340 03 0C 03 0C CF 3F CF 3F      ;
0348 96 FF 96 FF 33 CC C3 3C      ;

0350 0F FF 0F FF 0F FF 0F FF      ;SECOND 4X4 LED SEQUENCE
0358 71 72 74 78 B8 D8 E8 E4      ;
0360 E2 E1 D1 B1 71 72 74 B4      ;
0368 D4 D2 B2 B4 D4 D2 B2 B4      ;

0 - 9 COUNTER ROUTINE. THIS PROGRAM WILL COUNT FROM 0-9 AND REPEAT WHEN
BUTTON 'A' IS PRESSED.  NUMBERS WILL BE DISPLAYED ON THE SEVEN SEGMENT DISPLAY

0370 0E 0A          LD C,0A       ;LOAD C WITH 10
0372 11 DF 03       LD DE,03DF    ;LOAD DE WITH START OF 0-9 ASCII TABLE LESS 1
0375 DB 01          IN A,(01)     ;READ INPUT LATCH
0377 CB 7F          BIT 7,A       ;IS BUTTON 'A' PRESSED?
0379 28 FA          JR Z,0375     ;REPEAT READ UNTIL BUTTON HAS BEEN PRESSED
037B 13             INC DE        ;MOVE TO NEXT LOCATION
037C 1A             LD A,(DE)     ;LOAD ASCII VALUE TO A
037D D3 02          OUT (02),A    ;OUTPUT TO RIGHT SEVEN SEGMENT
037F DB 01          IN A,(01)     ;READ INPUT LATCH
0381 CB 7F          BIT 7,A       ;IS BUTTON 'A' STILL PRESSED?
0383 20 FA          JR NZ,037F    ;REPEAT READ UNTIL BUTTON IS RELEASED
0385 0D             DEC C         ;DECREASE COUNTER
```

```
0386 28 E8          JR Z,0370     ;IF ZERO REPEAT FROM START
0388 18 EB          JR 0375       ;DISPLAY NEXT NUMBER

038A 00 00 00 00 00 00            ;FILL
```

0 - F COUNTER ROUTINE USES THE DISPLAY AND KEY PRESS ROUTINE IN THE FULL COUNTER BUT ONLY FOR 0-F

```
0390 0E 10          LD C,10       ;16 DIGITS TO DISPLAY
0392 11 DF 03       LD DE,03DF    ;DATA TABLE LOCATION FOR 0-F
0395 21 90 03       LD HL,0390    ;RETURN LOCATION FOR JUMP ROUTINE
0398 18 0E          JR 03A8       ;DISPLAY AND INPUT ROUTINE JUMP

039A 00 00 00 00 00 00            ;FILL
```

A - Z, 0 - F COUNTER. SIMILAR TO THE ABOVE BUT TABLE LOOKUP FOR FULL TABLE

```
03A0 0E 2A          LD C,2A       ;FULL TABLE COUNT OF 42 CHARACTERS
03A2 11 C5 03       LD DE,03C5    ;START OF ASCII CHARACTER TABLE
03A5 21 A0 03       LD HL,03A0    ;RETURN ADDRESS FROM JUMP
03A8 DB 01          IN A,(01)     ;READ INPUT LATCH
03AA CB 7F          BIT 7,A       ;HAS BUTTON 'A' BEEN PRESSED?
03AC 28 FA          JR Z,03A8     ;NO, REPEAT READ LATCH
03AE 13             INC DE        ;YES, MOVE TO NEXT CHARACTER
03AF 1A             LD A,(DE)     ;LOAD CHARCTER IN A
03B0 D3 02          OUT (02),A    ;OUTPUT TO SEVEN SEGMENT
03B2 DB 01          IN A,(01)     ;READ INPUT LATCH
03B4 CB 7F          BIT 7,A       ;HAS BUTTON 'A' BEEN RELEASED?
03B6 20 FA          JR NZ,03B2    ;JUMP TO READ UNTIL RELEASED
03B8 0D             DEC C         ;MOVE COUNTER TO NEXT CHARACTER
03B9 28 02          JR Z,03BD     ;IF ZERO, RETURN TO START
03BB 18 EB          JR 03A8       ;DISPLAY NEXT CHARACTER
03BD E9             JP (HL)       ;JUMP BACK TO START ADDRESS

03BE 00 00 00 00 00 00 00 00      ;FILL
```

ASCII CHARACTER LOOKUP TABLE FOR SEVEN SEGMENT DISPLAY

```
03C6 77             ;A
03C7 7C             ;B
03C8 39             ;C
03C9 5E             ;D
03CA 79             ;E
03CB 71             ;F
03CC 3D             ;G
03CD 76             ;H
03CE 06             ;I
03CF 1E             ;J
03D0 72             ;K
03D1 38             ;L
03D2 47             ;M
03D3 37             ;N
03D4 3F             ;O
03D5 73             ;P
03D6 67             ;Q
03D7 33             ;R
03D8 6D             ;S
03D9 78             ;T
03DA 3E             ;U
03DB 1C             ;V
```

```
03DC 4E             ;W
03DD 4C             ;X
03DE 6E             ;Y
03DF 1B             ;Z
03E0 3F             ;0
03E1 06             ;1
03E2 5B             ;2
03E3 4F             ;3
03E4 66             ;4
03E5 6D             ;5
03E6 7D             ;6
03E7 07             ;7
03E8 7F             ;8
03E9 67             ;9
03EA 77             ;A
03EB 7C             ;B
03EC 39             ;C
03ED 5E             ;D
03EE 79             ;E
03EF 71             ;F
```

VERY LONG DELAY (PART 1) ROUTINE ISN'T USED ANYWHERE! BUT IS THERE AS AN EXAMPLE. IT IS IN TWO PARTS (I ASSUME TO FILL IN GAPS IN MEMORY). THIS PART DOES A FULL 16 BIT DELAY AND WHEN COMPLETE OUTPUTS A COUNTER TO THE DISPLAYS. THEN THE PROCESS IS REPEATED AGAIN. NOT REALLY USEFUL AS DELAYS ARE ALL THROUGH THE CODE THAT CAN BE USED.

```
03F0 3E 01          LD A,01       ;LOAD COUNTER TO A
03F2 ED 47          LD I,A        ;STORE IT IN THE INDEX REGISTER FOR LATER
03F4 11 FF FF       LD DE,FFFF    ;LOAD DE WITH ALL BITS SET
03F7 21 FF FF       LD HL,FFFF    ;LOAD HL WITH ALL BITS SET
03FA 2B             DEC HL        ;DECREASE HL
03FB 7C             LD A,H        ;AND SET ZERO FLAG
03FC B5             OR L          ;IF H AND L BOTH EQUAL ZERO
03FD C3 5A 04       JP 045A       ;JUMP TO SECOND PART OF DELAY (SEE BELOW)
```

00 - 99 COUNTER ROUTINE. THIS PROGRAM IS A BIT MORE COMPLICATED THAN THE SINGLE COUNTERS AS IT USES BOTH SEGMENT DISPLAYS, WHICH MEANS IT MUST USE MULTIPLEXING TO DISPLAY BOTH DIGITS AT THE SAME TIME. IT WORKS BY STORING THE COUNT IN E AND USE DAA TO CONVERT THE HEX VALUE TO DECIMAL. THEN EACH NIBBLE IS SEPARATED AND DISPLAYED IN EACH SEVEN SEGMENT. WHEN BUTTON 'A' IS PRESSED, THE VALUE INCREASED. WHEN BUTTON 'B' IS PRESSED THE VALUE DECREASES IT REPEATS WHEN COUNTER HITS 99.

```
0400 1E 00          LD E,00       ;LOAD E WITH ZERO FOR THE INTIAL COUNT
0402 7B             LD A,E        ;LOAD A WITH E
0403 E6 0F          AND 0F        ;MASK OFF UPPER NIBBLE
0405 21 E0 03       LD HL,03E0    ;LOAD HL WITH 0-9 ASCII TABLE
0408 85             ADD A,L       ;INDEX A WITH BASE OF TABLE
0409 6F             LD L,A        ;INDEX TABLE WITH A
040A 7E             LD A,(HL)     ;LOAD A WITH DIGIT IN ASCII TABLE
040B D3 02          OUT (02),A    ;OUTPUT IT TO THE RIGHT SEGMENT
040D 7B             LD A,E        ;RELOAD A WITH THE ORIGINAL E
040E 1F             RRA           ;SWAP UPPER
040F 1F             RRA           ;NIBBLE WITH
0410 1F             RRA           ;THE LOWER
0411 1F             RRA           ;NIBBLE IN A
0412 E6 0F          AND 0F        ;MASK OFF UPPER NIBBLE
0414 21 E0 03       LD HL,03E0    ;LOAD HL WITH 0-9 ASCII TABLE
```

```
0417 85           ADD A,L       ;INDEX A WITH BASE OF TABLE
0418 6F           LD L,A        ;INDEX TABLE WITH A
0419 7E           LD A,(HL)     ;LOAD A WITH DIGIT IN ASCII TABLE
041A CB FF        SET 7,A       ;SET BIT 7 TO DISPLAY IT ON LEFT SEGMENT
041C D3 02        OUT (02),A    ;OUTPUT IT TO THE LEFT SEGMENT
041E DB 01        IN A,(01)     ;READ INPUT LATCH
0420 CB 7F        BIT 7,A       ;HAS BUTTON 'A' BEEN PRESSED?
0422 28 06        JR Z,042A     ;NO, CHECK OTHER BUTTON
0424 7B           LD A,E        ;LOAD COUNTER IN A
0425 3C           INC A         ;INCREASE A
0426 27           DAA           ;CONVERT A TO DECIMAL IF GONE INTO A-F
0427 5F           LD E,A        ;LOAD A BACK INTO E
0428 18 08        JR 0432       ;CONTINUE PROGRAM
042A CB 77        BIT 6,A       ;HAS BUTTON 'B' BEEN PRESSED?
042C 28 D4        JR Z,0402     ;NO, JUMP BACK TO START AND REPEAT DISPLAY
042E 7B           LD A,E        ;LOAD COUNTER IN A
042F 3D           DEC A         ;DECREASE A
0430 27           DAA           ;CONVERT A TO DECIMAL IF GONE INTO A-F
0431 5F           LD E,A        ;LOAD A BACK INTO E
                                ;REPEAT DISPLAY ROUTINE? (NOT NEEDED AS IT
                                 COULD HAVE JUST JUMPED UP??)
0432 7B           LD A,E        ;RELOAD A WITH THE ORIGINAL E
0433 E6 0F        AND 0F        ;MASK OFF UPPER NIBBLE
0435 21 E0 03     LD HL,03E0    ;LOAD HL WITH 0-9 ASCII TABLE
0438 85           ADD A,L       ;INDEX A WITH BASE OF TABLE
0439 6F           LD L,A        ;INDEX TABLE WITH A
043A 7E           LD A,(HL)     ;LOAD A WITH DIGIT IN ASCII TABLE
043B D3 02        OUT (02),A    ;OUTPUT IT TO THE RIGHT SEGMENT
043D 7B           LD A,E        ;RELOAD A WITH THE ORIGINAL E
043E 1F           RRA           ;SWAP UPPER
043F 1F           RRA           ;NIBBLE WITH
0440 1F           RRA           ;THE LOWER
0441 1F           RRA           ;NIBBLE IN A
0442 E6 0F        AND 0F        ;MASK OFF UPPER NIBBLE
0444 21 E0 03     LD HL,03E0    ;LOAD HL WITH 0-9 ASCII TABLE
0447 85           ADD A,L       ;INDEX A WITH BASE OF TABLE
0448 6F           LD L,A        ;INDEX TABLE WITH A
0449 7E           LD A,(HL)     ;LOAD A WITH DIGIT IN ASCII TABLE
044A CB FF        SET 7,A       ;SET BIT 7 TO DISPLAY IT ON LEFT SEGMENT
044C D3 02        OUT (02),A    ;OUTPUT IT TO THE LEFT SEGMENT
044E DB 01        IN A,(01)     ;READ INPUT LATCH
0450 CB 7F        BIT 7,A       ;IS BUTTON 'A' STILL PRESSED?
0452 20 DE        JR NZ,0432    ;YES JUMP TO DISPLAY ROUTINE
0454 CB 77        BIT 6,A       ;IS BUTTON 'B' STILL PRESSED?
0456 20 DA        JR NZ,0432    ;YES JUMP TO DISPLAY ROUTINE
0458 18 A8        JR 0402       ;BUTTON RELEASE, REPEAT FROM START
```

VERY LONG DELAY (PART 1) CONTINUES THE DELAY ROUTINE. WHEN DELAY IS DONE A COUNTER BYTE IS OUTPUTTED TO THE DISPLAY.

```
045A C2 FA 03     JP NZ,03FA    ;IF H AND L ARE NOT ZERO DECREASE AGAIN
045D 1B           DEC DE        ;DECREASE DE
045E 7A           LD A,D        ;AND SET ZERO FLAG
045F B3           OR E          ;IF D AND E BOTH EQUAL ZERO
0460 C2 F7 03     JP NZ,03F7    ;IF D AND L ARE NOT ZERO DECREASE AGAIN
0463 ED 57        LD A,I        ;LOAD COUNTER BACK TO A
0465 D3 02        OUT (02),A    ;OUTPUT TO DISPLAYS
0467 3C           INC A         ;INCREASE COUNTER
0468 ED 47        LD I,A        ;SAVE COUNTER BACK TO I
```

```
046A C3 F4 03       JP 03F4       ;REPEAT LONG DELAY AGAIN

046D FF FF FF                     ;FILL
```

DICE ROUTINE. THIS IS ONE OF THE MORE COMPLEX PROGRAMS THAT COMBINES MANY DIFFERENT TECHNIQUES. IT STARTS WITH AN ANIMATION LOOP AND WHEN A KEY IS PRESSED, FLASHES THE SCREEN AND DISPLAYS A DICE VALUE THAT IS MULTIPLEXED. IT ALSO HAS A RANDOM FUNCTION TO GENERATE A UNIQUE DICE ROLL.

```
                                  ;PART 1
0470 16 0C          LD D,0C       ;LOAD D WITH 12 FOR LED SEQUENCE
0472 21 D3 04       LD HL,04D3    ;LOAD HL WITH LED DATA TABLE
0475 7E             LD A,(HL)     ;LOAD A WITH LED DATA
0476 D3 02          OUT (02),A    ;OUTPUT TO 4X4 LEDS
0478 23             INC HL        ;GET NEXT LED DATA SEQUENCE
0479 06 15          LD B,15       ;LOAD B WITH BUTTON TIMER FOR RANDOM NUMBER
047B 0E 06          LD C,06       ;LOAD C WITH RANDOM DICE VALUE
047D DB 01          IN A,(01)     ;READ INPUT LATCH
047F CB 7F          BIT 7,A       ;HAS BUTTON 'A' BEEN PRESSED?
0481 20 0A          JR NZ,048D    ;YES, JUMP TO DICE ROLE
0483 0D             DEC C         ;DECREASE C
0484 20 F7          JR NZ,047D    ;RE READ INPUT LATCH
0486 10 F3          DJNZ 047B     ;COUNTDOWN REPEAT BUTTON CHECK
0488 15             DEC D         ;MOVE COUNTER TO NEXT LED SEQUENCE
0489 28 E5          JR Z,0470     ;LOOP BACK TO FIRST SEQUENCE
048B 18 E8          JR 0475       ;PRINT THE NEXT LED SEQUENCE
                                  ;PART 2
048D 16 06          LD D,06       ;LOAD D WITH 6 FOR LED SEQUENCE REPEAT
048F 3E 0F          LD A,0F       ;LOAD A WITH 0F (BLANK ALL 4X4 LEDS)
0491 D3 02          OUT (02),A    ;OUTPUT TO 4X4
0493 10 FE          DJNZ 0493     ;DELAY
0495 3E FF          LD A,FF       ;LOAD A WITH FF (LIGHT ALL 4X4 LEDS)
0497 D3 02          OUT (02),A    ;OUTPUT TO 4X4
0499 10 FE          DJNZ 0499     ;DELAY
049B 15             DEC D         ;DECREASE D
049C 20 F1          JR NZ,048F    ;REPEAT PART 2
                                  ;PART 3
049E 16 80          LD D,80       ;OUTPUT THE DICE FOR 80 CYCLES
04A0 79             LD A,C        ;LOAD A WITH DICE ROLL VALUE
04A1 21 E0 04       LD HL,04E0    ;LOAD HL WITH TABLE TO DISPLAY 1
04A4 FE 01          CP 01         ;IS ROLL 1?
04A6 CA F5 04       JP Z,04F5     ;JUMP TO DICE DISPLAY
04A9 21 E3 04       LD HL,04E3    ;LOAD HL WITH TABLE TO DISPLAY 2
04AC FE 02          CP 02         ;IS ROLL 2?
04AE CA F5 04       JP Z,04F5     ;JUMP TO DICE DISPLAY
04B1 21 E6 04       LD HL,04E6    ;LOAD HL WITH TABLE TO DISPLAY 3
04B4 FE 03          CP 03         ;IS ROLL 3?
04B6 CA F5 04       JP Z,04F5     ;JUMP TO DICE DISPLAY
04B9 21 E9 04       LD HL,04E9    ;LOAD HL WITH TABLE TO DISPLAY 4
04BC FE 04          CP 04         ;IS ROLL 4?
04BE CA F5 04       JP Z,04F5     ;JUMP TO DICE DISPLAY
04C1 21 EC 04       LD HL,04EC    ;LOAD HL WITH TABLE TO DISPLAY 5
04C4 FE 05          CP 05         ;IS ROLL 5?
04C6 CA F5 04       JP Z,04F5     ;JUMP TO DICE DISPLAY
04C9 21 EF 04       LD HL,04EF    ;LOAD HL WITH TABLE TO DISPLAY 6
04CC FE 06          CP 06         ;IS ROLL 6?
04CE CA F5 04       JP Z,04F5     ;JUMP TO DICE DISPLAY

04D1 FF FF                        ;FILL
```

DICE LED SEQUENCE THAT MOVES AROUND THE 4X4 DISPLAY

```
04D3 71 72 74 78 B8 D8 E8 E4      ;4X4 LED SEQUENCE
04DB E2 E1 D1 B1                  ;

04DF FF                           ;FILL
```

DICE DISPLAY MULTIPLEX. THE FOLLOWING 3 BYTES PER DICE ROLL ARE THE LEDS NEEDED TO DISPLAY THE DICE VALUE. EACH BYTE IS SHOWN ONE AT A TIME AND ARE MULTIPLEXED TO CREATE THE ILLUSION THAT ALL BYTES ARE SHOWN AT THE SAME TIME.

```
04E0 B4 00 00       ;ONE
04E3 D2 00 78       ;TWO
04E6 72 B4 D8       ;THREE
04E9 52 00 58       ;FOUR
04EC 52 B4 58       ;FIVE
04EF 52 54 58       ;SIX

04F2 FF FF FF                     ;FILL
```

DICE DISPLAY ROUTINE TAKES IN THE THREE BYTES TO REPRESENT THE LEDS AND MULTIPLEXES THEM. ONCE 80 DISPLAY CYCLES ARE UP, AND BUTTON 'A' IS RELEASED, THE PROGRAM STARTS AGAIN.

```
04F5 7E             LD A,(HL)     ;LOAD A WITH FIRST VALUE OF DICE LED VALUE
04F6 D3 02          OUT (02),A    ;OUTPUT TO 4X4 LEDS
04F8 06 0A          LD B,0A       ;LOAD B WITH SMALL DELAY
04FA 10 FE          DJNZ 04FA     ;DELAY
04FC 23             INC HL        ;MOVE TO NEXT DICE LED VALUE
04FD 7E             LD A,(HL)     ;LOAD A WITH SECOND VALUE OF DICE LED VALUE
04FE D3 02          OUT (02),A    ;OUTPUT TO 4X4 LEDS
0500 06 0A          LD B,0A       ;LOAD B WITH SMALL DELAY
0502 10 FE          DJNZ 0502     ;DELAY
0504 23             INC HL        ;MOVE TO NEXT DICE LED VALUE
0505 7E             LD A,(HL)     ;LOAD A WITH THRID VALUE OF DICE LED VALUE
0506 D3 02          OUT (02),A    ;OUTPUT TO 4X4 LEDS
0508 06 0A          LD B,0A       ;LOAD B WITH SMALL DELAY
050A 10 FE          DJNZ 050A     ;DELAY
050C 2B             DEC HL        ;MOVE HL BACK TO
050D 2B             DEC HL        ;START OF DICE LED VALUE
050E 15             DEC D         ;REDUCE DELAY CYCLE BY ONE
050F 20 E4          JR NZ,04F5    ;REPEAT DICE DISPLAY IF NOT ZERO
0511 AF             XOR A         ;SET A TO ZERO TO BLANK DISPLAY
0512 D3 02          OUT (02),A    ;OUTPUT TO 4X4 LEDS
0514 DB 01          IN A,(01)     ;READ INPUT LATCH
0516 CB 7F          BIT 7,A       ;IS BUTTON 'A' STILL PRESSED?
0518 20 FA          JR NZ,0514    ;RE READ INPUT IF TRUE
051A C3 70 04       JP 0470       ;RESTART DICE ROUTINE

051D FF FF FF                     ;FILL
```

NOTE: THE FOLLOWING CODE IS UNDOCUMENTED IN THE TEC MAGAZINES ISSUE 13 AND 14. BUT CAN BE FOUND IN THE BD679 BOOK.

EPROM IN BINARY ROUTINE DISPLAYS THE CONTENTS OF THE EPROM STARTING AT LOCATION 0X0000. IT IS ONLY USEFUL VIEWING ON THE 8 LED DATA BITS.

```
0520 21 00 00       LD HL,0000    ;SET HL TO START OF EPROM
0523 7E             LD A,(HL)     ;LOAD THE CONTENTS OF HL INTO A
0524 23             INC HL        ;MOVE TO NEXT ADDRESS LOCATION
0525 D3 02          OUT (02),A    ;OUTPUT DATA TO 8 DATA BITS DISPLAY
0527 10 FE          DJNZ 0527     ;FULL DELAY
0529 10 FE          DJNZ 0529     ;FULL DELAY
052B 10 FE          DJNZ 052B     ;FULL DELAY
052D 18 F4          JR 0523       ;JUMP BACK TO START FOR NEXT VALUE

052F 04                           ;UNUSED?
```

POKER ROUTINE. THIS IS A GAME WHERE EACH COLUMN OF THE 4X4 HAS AN INDIVIDUAL LED THAT MOVES FROM THE TOP TO THE BOTTOM AND REPEATS. THIS IS TO SIMULATE A POKER JACKPOT MACHINE. THESE LEDS FALL AT DIFFERENT RATES. WHEN BUTTON 'A' IS PRESSED THE LEDS STOP FALLING FOR A SMALL MOMENT AND FALL RATE OF ONE OF THE LEDS WILL CHANGE. THE AIM IS TO PRESS BUTTON 'A' WHEN ALL FOUR LEDS ARE LINED UP ON THE SECOND ROW. AS THERE IS NO WAY TO CHECK THE RATE OF FALL WHEN THE BUTTON IS PRESSED, THIS GAME SEEMS TO BE MOSTLY RANDOM. AS RAM ISN'T AVAILABLE, THIS PROGRAM USES MOST REGISTERS TO STORE DATA.

```
  O O O O
  * * * * <- PRESS BUTTON 'A' TO WIN WHEN ALL LEDS ARE HERE
  O O O O     IF YOU STOP PRESSING BUTTON 'A' SOME LEDS WILL STOP!
  O O O O
```

```
0530 11 CA 05       LD DE,05CA    ;POINT DE TO FIRST COLUMN TABLE OF 4X4
0533 21 DD 05       LD HL,05DD    ;POINT HL TO SECOND COLUMN TABLE OF 4X4
0536 DD 21 F0 05    LD IX,05F0    ;POINT IX TO THIRD COLUMN TABLE OF 4X4
053A FD 21 0A 06    LD IY,060A    ;POINT IY TO FOURTH COLUMN TABLE OF 4X4
053E 0E 20          LD C,20       ;LOAD C WITH 20, IT'S USED TO MAKE LEDS FALL
0540 1A             LD A,(DE)     ;LOAD A WITH FIRST COLUMN LED VALUE
0541 D3 02          OUT (02),A    ;OUTPUT TO FIRST COLUMN OF 4X4
0543 7E             LD A,(HL)     ;LOAD A WITH SECOND COLUMN LED VALUE
0544 D3 02          OUT (02),A    ;OUTPUT TO SECOND COLUMN OF 4X4
0546 DD 7E 00       LD A,(IX+0)   ;LOAD A WITH THIRD COLUMN LED VALUE
0549 D3 02          OUT (02),A    ;OUTPUT TO THIRD COLUMN OF 4X4
054B FD 7E 00       LD A,(IY+0)   ;LOAD A WITH FOURTH COLUMN LED VALUE
054E D3 02          OUT (02),A    ;OUTPUT TO FOURTH COLUMN OF 4X4
0550 DB 01          IN A,(01)     ;READ INPUT LATCH
0552 CB 7F          BIT 7,A       ;IS BUTTON 'A' PRESSED?
0554 20 05          JR NZ,055B    ;YES!, JUMP TO USE C FOR FALL RATE
0556 0D             DEC C         ;NO, JUST DECREASE C
0557 20 E7          JR NZ,0540    ;IF C ISN'T ZERO, RE DISPLAY LEDS
0559 18 03          JR 055E       ;IF C IS ZERO, FALL USING OLD RATE
055B 79             LD A,C        ;LOAD A WITH C
055C ED 47          LD I,A        ;LOAD A TO INDEX REGISTER
055E ED 57          LD A,I        ;LOAD INDEX REGISTER TO A
0560 1F             RRA           ;BIT ROTATE A ONE BIT TO THE RIGHT
0561 3C             INC A         ;INCREASE A SO IT ISN'T ZERO
0562 ED 47          LD I,A        ;STORE A IN INDEX REGISTER
0564 E6 07          AND 07        ;MASK OFF TOP 5 BITS
0566 FE 05          CP 05         ;IS A EQUAL TO 5?
0568 CA CF 05       JP Z,05CF     ;MAKE COLUMN 1 DROP BY ONE
```

```
056B FE 02          CP 02          ;IS A EQUAL TO 2?
056D CA E2 05       JP Z,05E2      ;MAKE COLUMN 2 DROP BY ONE
0570 FE 03          CP 03          ;IS A EQUAL TO 3?
0572 CA F5 05       JP Z,05F5      ;MAKE COLUMN 3 DROP BY ONE
0575 FE 04          CP 04          ;IS A EQUAL TO 4?
0577 CA 0F 06       JP Z,060F      ;MAKE COLUMN 4 DROP BY ONE
057A DB 01          IN A,(01)      ;READ INPUT LATCH
057C CB 7F          BIT 7,A        ;HAS BUTTON 'A' BEEN PRESSED?
057E CA 3E 05       JP Z,053E      ;NO, JUMP BACK TO DISPLAY AT THE BEGINNING
0581 0E 03          LD C,03        ;YES, LOAD C WITH 3
0583 06 00          LD B,00        ;LOAD B WITH 00 FOR FULL DELAY
0585 1A             LD A,(DE)      ;DISPLAY THE
0586 D3 02          OUT (02),A     ;CURRENT
0588 7E             LD A,(HL)      ;LEDS
0589 D3 02          OUT (02),A     ;AND
058B DD 7E 00       LD A,(IX+0)    ;PAUSE
058E D3 02          OUT (02),A     ;FOR
0590 FD 7E 00       LD A,(IY+0)    ;THREE
0593 D3 02          OUT (02),A     ;FULL
0595 10 EE          DJNZ 0585      ;BYTE
0597 0D             DEC C          ;DELAYS
0598 20 EB          JR NZ,0585     ;REPEAT DISPLAY
                                   ;CHECK FOR WIN
059A 1A             LD A,(DE)      ;LOAD FIRST COLUMN VALUE
059B FE B1          CP B1          ;IS IT ON THE SECOND ROW?
059D C2 3E 05       JP NZ,053E     ;NO, JUMP TO START
05A0 7E             LD A,(HL)      ;LOAD SECOND COLUMN VALUE
05A1 FE B2          CP B2          ;IS IT ON THE SECOND ROW?
05A3 C2 3E 05       JP NZ,053E     ;NO, JUMP TO START
05A6 DD 7E 00       LD A,(IX+0)    ;LOAD THRID COLUMN VALUE
05A9 FE B4          CP B4          ;IS IT ON THE SECOND ROW?
05AB C2 3E 05       JP NZ,053E     ;NO, JUMP TO START
05AE FD 7E 00       LD A,(IY+0)    ;LOAD FOURTH COLUMN VALUE
05B1 FE B8          CP B8          ;IS IT ON THE SECOND ROW?
05B3 C2 3E 05       JP NZ,053E     ;NO, JUMP TO START
                                   ;DISPLAY WIN BY FLASHING ALL LEDS
05B6 3E 0F          LD A,0F        ;LOAD A WITH ALL LEDS OFF
05B8 D3 02          OUT (02),A     ;OUTPUT TO 4X4
05BA 10 FE          DJNZ 05BA      ;DELAY
05BC 10 FE          DJNZ 05BC      ;DELAY
05BE 3E FF          LD A,FF        ;LOAD A WITH ALL LEDS ON
05C0 D3 02          OUT (02),A     ;OUTPUT TO 4X4
05C2 10 FE          DJNZ 05C2      ;DELAY
05C4 10 FE          DJNZ 05C4      ;DELAY
05C6 18 EE          JR 05B6        ;JUMP TO REPEAT WIN

05C8 00 00                         ;FILL

POKER LED MOVE DOWN ROUTINES FOR COLUMNS 1,2,3 AND 4

05CA B1 D1 E1 71 FF                ;4X4 LED VALUES FOR FIRST COLUMN
05CF 13             INC DE         ;INCREASE INDEX BY ONE
05D0 1A             LD A,(DE)      ;LOAD A WITH INDEX VALUE
05D1 FE FF          CP FF          ;IS IT FF?
05D3 C2 3E 05       JP NZ,053E     ;NO, EXIT ROUTINE
05D6 1B             DEC DE         ;YES,
05D7 1B             DEC DE         ;MOVE INDEX
05D8 1B             DEC DE         ;BACK TO
05D9 1B             DEC DE         ;START
```

```
05DA C3 3E 05       JP 053E       ;EXIT ROUTINE

05DD 72 B2 D2 E2 FF               ;4X4 LED VALUES FOR SECOND COLUMN
05E2 23             INC HL        ;INCREASE INDEX BY ONE
05E3 7E             LD A,(HL)     ;LOAD A WITH INDEX VALUE
05E4 FE FF          CP FF         ;IS IT FF?
05E6 C2 3E 05       JP NZ,053E    ;NO, EXIT ROUTINE
05E9 2B             DEC HL        ;YES,
05EA 2B             DEC HL        ;MOVE INDEX
05EB 2B             DEC HL        ;BACK TO
05EC 2B             DEC HL        ;START
05ED C3 3E 05       JP 053E       ;EXIT ROUTINE

05F0 D4 E4 74 B4 FF               ;4X4 LED VALUES FOR THRID COLUMN
05F5 DD 23          INC IX        ;INCREASE INDEX BY ONE
05F7 DD 7E 00       LD A,(IX+0)   ;LOAD A WITH INDEX VALUE
05FA FE FF          CP FF         ;IS IT FF?
05FC C2 3E 05       JP NZ,053E    ;NO, EXIT ROUTINE
05FF DD 2B          DEC IX        ;YES,
0601 DD 2B          DEC IX        ;MOVE INDEX
0603 DD 2B          DEC IX        ;BACK TO
0605 DD 2B          DEC IX        ;START
0607 C3 3E 05       JP 053E       ;EXIT ROUTINE

060A E8 78 B8 D8 FF               ;4X4 LED VALUES FOR FOURTH COLUMN
060F FD 23          INC IY        ;INCREASE INDEX BY ONE
0611 FD 7E 00       LD A,(IY+0)   ;LOAD A WITH INDEX VALUE
0614 FE FF          CP FF         ;IS IT FF?
0616 C2 3E 05       JP NZ,053E    ;NO, EXIT ROUTINE
0619 FD 2B          DEC IY        ;YES,
061B FD 2B          DEC IY        ;MOVE INDEX
061D FD 2B          DEC IY        ;BACK TO
061F FD 2B          DEC IY        ;START
0621 C3 3E 05       JP 053E       ;EXIT ROUTINE

0624 00 00 00 00 00 00 00 00      ;FILL
062C 00 00 00 00                  ;FILL
```

BINARY CLOCK ROUTINE DISPLAYS A BINARY CLOCK USING THE 4X4 LED DISPLAY. THE CLOCK USES MULTIPLEXING TO DISPLAY MULTIPLE VALUES ON THE 4X4. BINARY VALUES ARE DISPLAYED BOTTOM TO TOP AND TIME FROM RIGHT TO LEFT. TIME IS SPLIT IN LOW MINUTES, MINUTES, HOURS AND TENS OF HOURS. THE INITIAL TIME IS 0100. IF BUTTON 'A' IS PRESSED IT WILL AUTO INCREMENT THE CLOCK, OTHERWISE IT WILL INCREMENT AUTOMATICALLY BASED ON THE VALUE ON THE INPUT LATCH.

```
    O O O O
    O O O O    (T)EN HOURS
    O O O O    (H)OURS
    O O O O    (M)INUTES
    ^ ^ ^ ^    (L)OW MINUTES
    T H M L
```

```
0630 11 00 01       LD DE,0100    ;START TIME OF 0100 IE:1PM
0633 31 D0 09       LD SP,09D0    ;MINUTE TIMER STORED IN SP REGISTER
0636 21 A0 06       LD HL,06A0    ;POINT HL TO LOW MINUTES TABLE
0639 7B             LD A,E        ;INDEX A WITH MINUTES
063A E6 0F          AND 0F        ;MASK OFF TENS OF MINUTES
063C 85             ADD A,L       ;INDEX A WITH L
063D 6F             LD L,A        ;INDEX TABLE WITH A
```

```
063E 7E          LD A,(HL)     ;LOAD LED VALUE TO A
063F D3 02       OUT (02),A    ;OUTPUT TO 4X4
0641 21 AA 06    LD HL,06AA    ;POINT HL TO TENS OF MINUTES TABLE
0644 7B          LD A,E        ;INDEX A WITH MINUTES
0645 1F          RRA           ;SWAP LOW
0646 1F          RRA           ;NIBBLE WITH
0647 1F          RRA           ;HIGH
0648 1F          RRA           ;NIBBLE
0649 E6 0F       AND 0F        ;MASK OFF LOW MINUTES
064B 85          ADD A,L       ;INDEX A WITH L
064C 6F          LD L,A        ;INDEX TABLE WITH A
064D 7E          LD A,(HL)     ;LOAD LED VALUE TO A
064E D3 02       OUT (02),A    ;OUTPUT TO 4X4
0650 21 B0 06    LD HL,06B0    ;POINT HL TO LOW HOURS TABLE
0653 7A          LD A,D        ;INDEX A WITH HOURS
0654 E6 0F       AND 0F        ;MASK OFF TENS OF HOURS
0656 85          ADD A,L       ;INDEX A WITH L
0657 6F          LD L,A        ;INDEX TABLE WITH A
0658 7E          LD A,(HL)     ;LOAD LED VALUE TO A
0659 D3 02       OUT (02),A    ;OUTPUT TO 4X4
065B 21 BA 06    LD HL,06BA    ;POINT HL TO TENS OF HOURS TABLE
065E 7A          LD A,D        ;INDEX A WITH HOURS
065F 1F          RRA           ;SWAP LOW
0660 1F          RRA           ;NIBBLE WITH
0661 1F          RRA           ;HIGH
0662 1F          RRA           ;NIBBLE
0663 E6 0F       AND 0F        ;MASK OFF LOW HOURS
0665 85          ADD A,L       ;INDEX A WITH L
0666 6F          LD L,A        ;INDEX TABLE WITH A
0667 7E          LD A,(HL)     ;LOAD LED VALUE TO A
0668 D3 02       OUT (02),A    ;OUTPUT TO 4X4
066A DB 01       IN A,(01)     ;READ INPUT LATCH
066C CB 7F       BIT 7,A       ;CHECK IF BUTTON 'A' IS PRESSED
066E 28 04       JR Z,0674     ;NOT PRESSED JUMP TO AUTO INCREMENT
0670 10 FE       DJNZ 0670     ;BUTTON PRESSED SO DELAY
0672 18 14       JR 0688       ;AND JUMP TO TIME INCREMENT
0674 3B          DEC SP        ;AUTO INCREMENT TIMER BY DECREASING SP
0675 21 00 00    LD HL,0000    ;LOAD HL WITH ZEROS
0678 39          ADD HL,SP     ;ADD HL TO SP
0679 7D          LD A,L        ;CHECK IF L
067A B4          OR H          ;EQUALS H AND EQUALS ZERO
067B C2 36 06    JP NZ,0636    ;NOT ZERO, RE DISPLAY TIME
067E DB 01       IN A,(01)     ;READ INPUT LATCH FOR CLOCK UPDATE DELAY
0680 47          LD B,A        ;LOAD VALUE TO B
0681 0E FF       LD C,FF       ;LOAD C WITH FF
0683 0B          DEC BC        ;DECREASE BC
0684 79          LD A,C        ;CHECK IF B
0685 B0          OR B          ;EQUALS C EQUALS ZERO
0686 20 FB       JR NZ,0683    ;REPEAT DELAY IF NOT ZERO
0688 7B          LD A,E        ;LOAD MINUTES TO A
0689 3C          INC A         ;INCREASE MINUTES
068A 27          DAA           ;CONVERT TO DECIMAL
068B 5F          LD E,A        ;LOAD DECIMAL VALUE BACK TO E
068C FE 60       CP 60         ;COMPARE A WITH 60 MINUTES
068E C2 33 06    JP NZ,0633    ;SKIP HOURS UPDATE AND JUMP TO DISPLAY UPDATE
0691 1E 00       LD E,00       ;LOAD E WITH ZERO TO RESET MINUTES
0693 7A          LD A,D        ;LOAD D WITH HOURS
0694 3C          INC A         ;INCREASE HOURS
0695 27          DAA           ;CONVERT TO DECIMAL
```

```
0696 57            LD D,A        ;LOAD DECIMAL VALUE BACK TO D
0697 FE 13         CP 13         ;COMPARE A WITH 13 HOURS
0699 C2 33 06      JP NZ,0633    ;NOT 13 THEN JUMP TO DISPLAY UPDATE
069C C3 30 06      JP 0630       ;RESTART COUNTER BACK TO 0100
069F 00                          ;FILL

BINARY CLOCK 4X4 LED SEQUENCE.  EACH VALUE IS INDEXED BASED ON CLOCK VALUE
06A0 F8 E8 D8 C8 B8 A8 98 88 78 68 ;SECONDS
06AA F4 E4 D4 C4 B4 A4             ;MINUTES
06B0 F2 E2 D2 C2 B2 A2 92 82 72 62 ;HOURS (ONES)
06BA F1 E1                         ;HOURS (TENS)

06BC 00 FF 00 FF                   ;FILL

ONE MINTUE TIMER ROUTINE.  TAKES IN THE INPUT LATCH VALUE OF 6C AND SETS
'A' WITH ONE, CALLS THE ONE MINUTE TIMER SUB ROUTINE.
06C0 DB 01         IN A,(01)     ;READ INPUT LATCH
06C2 47            LD B,A        ;SAVE VALUE IN B
06C3 3E 01         LD A,01       ;LOAD A WITH ONE MINUTE
06C5 DD 21 C0 06   LD IX,06C0    ;LOAD IX WITH JUMP RETURN ADDRESS
06C9 C3 40 07      JP 0740       ;CALL ONE MINUTE DELAY SUB ROUTINE

06CC 00 00 00 00                 ;FILL

THREE MINTUE TIMER ROUTINE.  TAKES IN THE INPUT LATCH VALUE OF 6D AND SETS
'A' WITH THREE, CALLS THE ONE MINUTE TIMER SUB ROUTINE.
06D0 DB 01         IN A,(01)     ;READ INPUT LATCH
06D2 47            LD B,A        ;SAVE VALUE IN B
06D3 3E 03         LD A,03       ;LOAD A WITH THREE MINUTES
06D5 DD 21 D0 06   LD IX,06D0    ;LOAD IX WITH JUMP RETURN ADDRESS
06D9 C3 40 07      JP 0740       ;CALL ONE MINUTE DELAY SUB ROUTINE

06DC 00 00 00 00                 ;FILL

ONE HOUR TIMER ROUTINE.  TAKES IN THE INPUT LATCH VALUE OF 6E AND SETS
'A' WITH 60, CALLS THE ONE MINUTE TIMER SUB ROUTINE.
06E0 DB 01         IN A,(01)     ;READ INPUT LATCH
06E2 47            LD B,A        ;SAVE VALUE IN B
06E3 3E 3C         LD A,3C       ;LOAD A WITH SIXTY MINUTES
06E5 DD 21 E0 06   LD IX,06E0    ;LOAD IX WITH JUMP RETURN ADDRESS
06E9 C3 40 07      JP 0740       ;CALL ONE MINUTE DELAY SUB ROUTINE

06EC 00 00 00 00                 ;FILL

ADJUSTABLE TIMER ROUTINE WILL USE THE INPUT LATCH TO SET THE TIMER REQUIRED. IT
USES THE RUNNING LETTER ROUTINE TO DISPLAY INSTRUCTIONS ON WHEN TO SET HE INPUT
LATCH.  ASCII FOR THE INSTRUCTIONS ARE FOUND AT 0X0765.  IT FIRST ASK TO SET
THE INPUT LATCH TO ZERO, THEN TO PUSH BUTTON 'B', THEN TO SET DELAY VALUE AND
PRESS BUTTON 'A'.  THEN IT WILL CALL THE ONE MINUTE DELAY ROUTINE WITH THE
INPUTTED DELAY VALUE IN MINUTES. DELAY CAN BE BETWEEN 0 AND 127 MINUTES

06F0 DD 21 65 07   LD IX,0765    ;LOAD IX WITH FIRST ASCII DATA TABLE
06F4 21 FA 06      LD HL,06FA    ;LOAD HL WITH JUMP RETURN ADDRESS
06F7 C3 D0 00      JP 00D0       ;CALL RUNNING LETTER ROUTINE
06FA DB 01         IN A,(01)     ;READ INPUT LATCH
06FC FE 00         CP 00         ;CHECK FOR IT TO BE ZERO
06FE 20 F0         JR NZ,06F0    ;IF NOT ZERO REPEAT FIRST MESSAGE
0700 DD 21 78 07   LD IX,0778    ;LOAD IX WITH SECOND ASCII DATA TABLE
0704 21 0A 07      LD HL,070A    ;LOAD HL WITH JUMP RETURN ADDRESS
```

```
0707 C3 D0 00       JP 00D0       ;CALL RUNNING LETTER ROUTINE
070A DB 01          IN A,(01)     ;READ INPUT LATCH
070C CB 77          BIT 6,A       ;CHECK IF BUTTON 'B' IS PRESSED
070E 28 F0          JR Z,0700     ;IF NOT REPEAT SECOND MESSAGE
0710 DD 21 82 07    LD IX,0782    ;LOAD IX WITH THIRD ASCII DATA TABLE
0714 21 1A 07       LD HL,071A    ;LOAD HL WITH JUMP RETURN ADDRESS
0717 C3 D0 00       JP 00D0       ;CALL RUNNING LETTER ROUTINE
071A DB 01          IN A,(01)     ;READ INPUT LATCH
071C CB 7F          BIT 7,A       ;CHECK IF BUTTON 'A' IS PRESSED
071E 28 F0          JR Z,0710     ;IF NOT REPEAT THIRD MESSAGE
0720 06 80          LD B,80       ;LOAD B WITH 80
0722 3E 81          LD A,81       ;LOAD A LEFT TOP SEGMENT
0724 D3 02          OUT (02),A    ;OUTPUT TO SEVEN SEGMENT DISPLAY
0726 3E 00          LD A,00       ;LOAD A WITH ZERO
0728 D3 02          OUT (02),A    ;BLANK OUTPUT
072A 10 F6          DJNZ 0722     ;REPEAT DISPLAY 80 TIMES
072C 06 80          LD B,80       ;LOAD B WITH 80
072E DB 01          IN A,(01)     ;READ INPUT LATCH
0730 E6 7F          AND 7F        ;MASK OUT BIT 7
0732 ED 47          LD I,A        ;LOAD INDEX REGISTER WITH MINUTES
0734 DD 21 2A 07    LD IX,072A    ;LOAD IX WITH RETURN ADDRESS
0738 C3 40 07       JP 0740       ;CALL ONE MINUTE DELAY SUB ROUTINE

073B 00 00 00 00 00               ;FILL
```

1 MINUTE DELAY SUB ROUTINE WILL COUNT DOWN FROM A STARTING VALUE TO ZERO.
WHEN ZERO, THE TOP SEGMENT ON THE LEFT DISPLAY WILL LIGHT UP. IF BUTTON 'A'
IS PRESSED, THE TIMER WILL REPEAT. THE VALUE IN REGISTER A IS SET IN THE
CALLING ROUTINE THAT REPRESENTS THE NUMBER OF MINUTES TO COUNT DOWN. IX
IS THE RETURN ADDRESS. REGISTER B SET EXTERNALLY IS ANOTHER COUNTER.
OBVIOUSLY THE TIMER DEPENDS ON THE CLOCK SPEED SO SPEED
IS TO BE SET BASED ON AN EXTERNAL CLOCK.

```
0740 ED 47          LD I,A        ;SAVE NUMBER OF MINUTES IN THE INDEX REGISTER
0742 11 FF 8B       LD DE,8BFF    ;LOAD DE WITH A LONG DELAY
0745 1B             DEC DE        ;DECREASE DE
0746 7B             LD A,E        ;WHEN D
0747 B2             OR D          ;AND E BOTH EQUAL ZERO
0748 20 FB          JR NZ,0745    ;CONTINUE
074A ED 57          LD A,I        ;RELOAD A WITH NUMBER OF MINUTES
074C 3D             DEC A         ;DECREASE A
074D 20 F1          JR NZ,0740    ;LOOP COUNTER AGAIN
074F 10 FE          DJNZ 074F     ;DELAY ON B
0751 3E 81          LD A,81       ;LOAD A WITH LEFT TOP SEGMENT
0753 D3 02          OUT (02),A    ;OUTPUT TO SEVEN SEGMENTS
0755 3E 00          LD A,00       ;LOAD A WITH ZERO
0757 D3 02          OUT (02),A    ;BLANK OUTPUT
0759 DB 01          IN A,(01)     ;READ INPUT LATCH
075B CB 7F          BIT 7,A       ;HAS BUTTON 'A' BEEN PRESSED?
075D 28 F2          JR Z,0751     ;RE DISPLAY IF NOT PRESSED
075F DD E9          JP (IX)       ;PRESSED, REPEAT TIMER

0761 00 00 00 00                  ;FILL
```

ADJUSTABLE TIMER MESSAGE LOOKUP TABLE

```
0765 00 6D 79 78 00               ;SET
076A 77 38 38 00                  ;ALL
076E 78 3F 00                     ;TO
0771 1B 79 33 3F 00 00 FF         ;ZERO
```

```
0778 00 73 3E 6D 76 00 7C         ;PUSH B
077F 00 00 FF                     ;

0782 00 6D 79 78 00               ;SET
0787 5E 79 38 77 6E 00            ;DELAY
078D 1C 77 38 3E 79 00            ;VALUE
0793 40 00                        ; -
0795 73 3E 6D 76 00 77            ;PUSH A
079B 00 00 FF                     ;

079E 00 00                        ;FILL
```

FINAL MESSAGE ROUTINE. IT CALLS THE RUNNING LETTER ROUTINE, BUT AS OPPOSED TO THE RUNNING NAMES, THERE IS NO VARIABLE TEXT. THIS IS A SOMEWHAT HIDDEN TEXT MESSAGE. BUT I'VE DISASSEMBLED IT SO HERE IT IS!!

```
07A0 DD 21 AA 07    LD IX,07AA    ;POINT IX TO ASCII DATA TABLE
07A4 21 A0 07       LD HL,07A0    ;POINT HL TO RETURN ADDRESS FROM JUMP
07A7 C3 D0 00       JP 00D0       ;CALL RUNNING LETTER ROUTINE
```

FINAL MESSAGE LOOKUP TABLE

```
07AA 00 33 3F 47 00               ;ROM
07AF 79 37 5E 6D 00               ;ENDS
07B4 77 78 00                     ;AT
07B7 3F 07 71 71 00               ;07FF
07BC 39 76 77 37 3D 79 00         ;CHANGE
07C3 38 79 77 5E 00               ;LEAD
07C8 71 3F 33 00                  ;FOR
07CC 3E 73 73 79 33 00            ;UPPER
07D2 76 77 38 71 00               ;HALF
07D7 77 37 5E 00                  ;AND
07DB 3E 6D 79 00                  ;USE
07DF 71 3F 33 00                  ;FOR
07E3 6E 3F 3E 33 00               ;YOUR
07E8 3F 4E 37 00                  ;OWN
07EC 06 5E 79 77 6D 00            ;IDEAS
07F2 39 76 79 79 33 6D 00         ;CHEERS
07F9 39 3F 38 06 37 00 FF         ;COLIN
```

<END OF ROM>

ADDENDUM

THIS ROM COMES IN ANOTHER VARIATION. EVERYTHING IS IDENTICAL ON BOTH ROMS EXCEPT FOR THE JUMP ROUTINE AT 0X0000. THE FOLLOWING LISTING IS THE ALTERNATE JUMP ROUTINE WHICH IS USED TO SELECT WHICH PROGRAM IS TO BE RUN. IT DOES EXACTLY THE SAME AS THE ONE PUBLISHED BUT IN AN INEFFICIENT WAY.
IE: IF 7A IS ON THE DIP SWITCH, IT WILL JUMP TO LOCATION 07A0.
AS IT IT BIGGER THAT 16 BYTES, THE TONE ROUTINE AT 0X0010 IS REMOVED AND WITH THE LEFT OVER BYTES, THE QUICK DRAW ANIMATION CODE IS PLACED AT 0X0017. I'M NOT SURE WHICH VERSION CAME FIRST.

```
0000 06 00          LD B,00       ;RESET B TO ZERO, TO BE USED AFTER THE JUMP
0002 DB 01          IN A,(01)     ;READ THE INPUT LATCH
0004 17             RLA           ;MOVE LOW NIBBLE
0005 17             RLA           ;INTO UPPER NIBBLE
0006 17             RLA           ;
0007 17             RLA           ;
0008 E6 F0          AND F0        ;MASK OUT LOWER NIBBLE
000A 6F             LD L,A        ;SAVE A IN L
000B DB 01          IN A,(01)     ;READ THE INPUT LATCH
000D 1F             RRA           ;MOVE HIGH NIBBLE
000E 1F             RRA           ;INTO LOWER NIBBLE
000F 1F             RRA           ;
0010 1F             RRA           ;
0011 E6 0F          AND 0F        ;MASK OUT UPPER NIBBLE
0013 67             LD H,A        ;SAVE A IN H
0014 E9             JP (HL)       ;JUMP TO ADDRESS

0015 00 00                        ;FILL

QUICK DRAW LOOKUP FOR ANIMATION.  NOTE FIRST VALUE IS DIFFERENT TO ORIGINAL

0017 EF 02 04 08 88 90 A0 81 00   ;OUTER SEQUENTIAL SEGEMENTS
```