AN563

Using PIC16C5X Microcontrollers as LCD Drivers

INTRODUCTION

This application report describes an LCD controller implementation using a PIC16C55 microcontroller. This technique offers display capabilities for applications that require a small display at a low cost together with the capabilities of the standard PIC16C55 microcontroller. We start by an overview of LCD devices and their theory of operation followed by software implementation issues of the controller. The source code for controlling a multiplexed LCD display is included in Appendix A.

LIQUID CRYSTAL DISPLAYS

The Liquid Crystal Display (LCD) is a thin layer of "Liquid Crystal Material" deposited between two plates of glass. The raw LCD is often referred to as "glass". Electrodes are attached to both sides of the glass. One side is referred to as common or backplane, while the other side is referred to as segment.

An LCD is modelled as a capacitor, with one side connected to the common plane and the other side connected to the segment, as shown in Figure 1. LCDs are sensitive to Root Mean Square Voltage levels. When a V_{RMS} level of zero volts is applied to the LCD, the LCD is practically transparent.

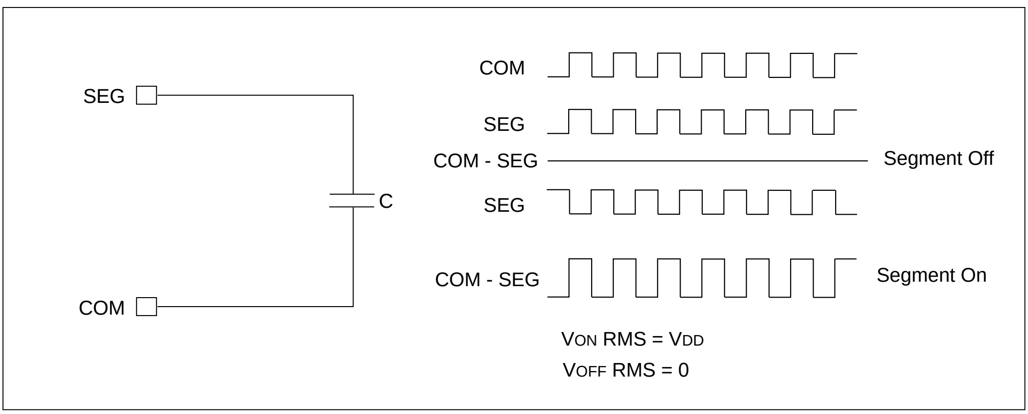
To turn an LCD segment "on", which makes the segment turn dark or opaque, an LCD RMS voltage that is greater than the LCD threshold voltage is applied to the LCD. The RMS LCD voltage is the RMS voltage across the capacitor C in Figure 1, which is equal to the potential difference between SEG and COM values.

Different LCDs have different characteristics; Figure 2 shows typical voltage vs relative contrast characteristics. Notation on curve shows operating points for multiplex operation with the threshold voltage set to 1.7 Vrms. This voltage is often used as the measure of voltage for LCD to be "off" or transparent. The curve is normalized and assumes a viewing angle of 90° to the plane of the LCD.

Contrast control, the process of turning on a segment, is achieved by moving the operating point of the LCD by applying voltage to the LCD that is greater than the LCD threshold voltages. A typical circuit to accomplish this task is shown in Figure 3.

Driving a liquid crystal display at direct current (DC) will cause permanent damage to the display unit. In order to prevent irreversible electrochemical action from destroying the display, the voltage at all segment locations must reverse polarity periodically so that a zero net voltage is applied to the device. This process is referred to as AC voltage application. There are two LCD driving methods available: Static driving method and multiplexed driving method.

FIGURE 1: ELECTRICAL MODEL OF AN LCD SEGMENT WITH DRIVING VOLTAGES



Conventional LCDs have separate external connections for each and every segment plus a common plane. This is the most basic method that results in good display quality. The main disadvantage of this driving method is that each segment requires one liquid crystal driver. The static driving method uses the frame frequency, defined as a period of the common plane signal, of several tens to several hundred Hz. A lower frequency would result in blinking effects and higher frequencies would increase power requirements. To turn a segment on, a voltage that has an opposite polarity to the common plane signal must be applied resulting in a large RMS voltage across the plates. To turn off a segment a voltage that is of the same polarity to common plane signal is applied. This drive method is universal to driving LCD segments. Figure 1 shows an example of this driving method.

The LCD frequency is defined as the rate of output changes of the common plane and segment signals, whereas the frame rate is defined as $f_{frame} = \frac{f_{frame}}{N}$ where N is the multiplex rate or number of backplane. Typically, f_{frame} ranges from 25HZ to 300HZ. The most commonly used frame fequency is 40-70HZ. A lower frequency would result in flicker effects and higher frequency would increase power requirements.

Multiplexed LCDs maintain their liquid crystal characteristics. These are low power consumption, high contrast ratio under high ambient light levels, and reduce the number of external connections necessary for dot matrix and alphanumeric displays. The multiplex driving method reduces the number of driver circuits, or microcontroller I/O pins if a software method is used. The method of drive for multiplexed displays is Time Division Multiplex (TDM) with the number of time divisions equal to twice the number of common planes used in a given format. In order to prevent permanent damage to the LCD display, the voltage at all segment locations must reverse polarity periodically so that zero net voltage is applied. This is the reason for the doubling in time divisions; each common plane must be alternately driven with a voltage pulse of opposite polarity. The drive frequency should be greater than the flicker rate of 25 Hz. Since increasing the drive frequency significantly above this value increases current demand by the CMOS circuitry, an upper drive frequency level of 60 Hz is recommended by most LCD manufacturers. We have chosen a drive rate of 50 Hz for this application report which results in a frame period of 20 ms. The most commonly available formats are 2x4, 3x3, and 5x7. In this report we use a 2x4 format LCD to display hexadecimal digits.

FIGURE 2: TYPICAL LCD CHARACTERISTICS

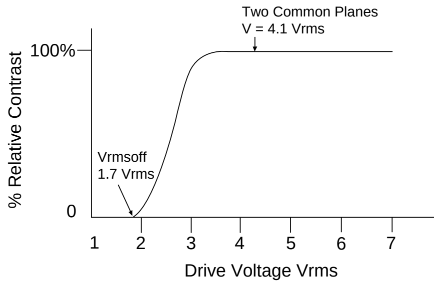
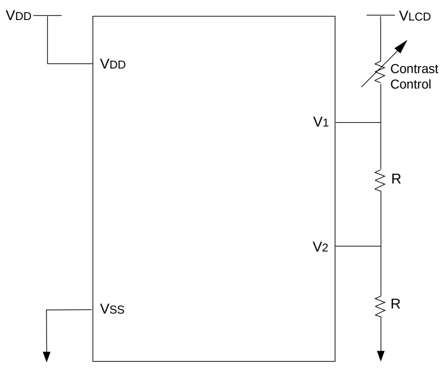


FIGURE 3: CONTRAST CONTROL CIRCUIT



To better understand multiplexed LCD control it is best to look at the general case. The segments in a multiplexed LCD are arranged in an X-Y grid form as shown in Figure 4. The common plane signals maintain their relative shape at all times, as shown in Figure 5. To turn on segment 1 (SEG1), we need to apply a voltage Vd, such that Vs+Vd turns the segment on and Vs-Vd turns the segment off. Note that the segment signal Vd is symmetrical. This is a consequence of the intervals that the common plane signal is not present at all times. Use of nonsymmetrical waveform will result in a higher Vrms present on the unaddressed segments. The symmetrical nature of the waveforms theoretically result in a zero DC voltage levels. CMOS drivers (e.g. microcontrollers) operate at 0 to +5V levels (rail voltage levels). This would require driving voltages beyond the range of operation. This constraint is addressed by a technique referred to as “level shifting” or “biasing”. Level shifting allows application of voltages in the range of 0 to +2.5V, which is compatible with these drivers. This would require an additional voltage level of +2.5V, which can be implemented through a simple resistive voltage divider circuit.

FIGURE 4: MULTIPLEXED LCD SEGMENT ARRANGEMENT

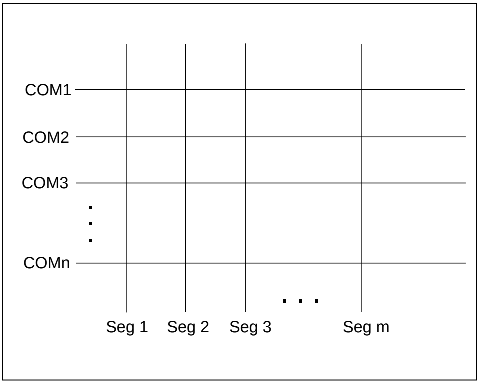
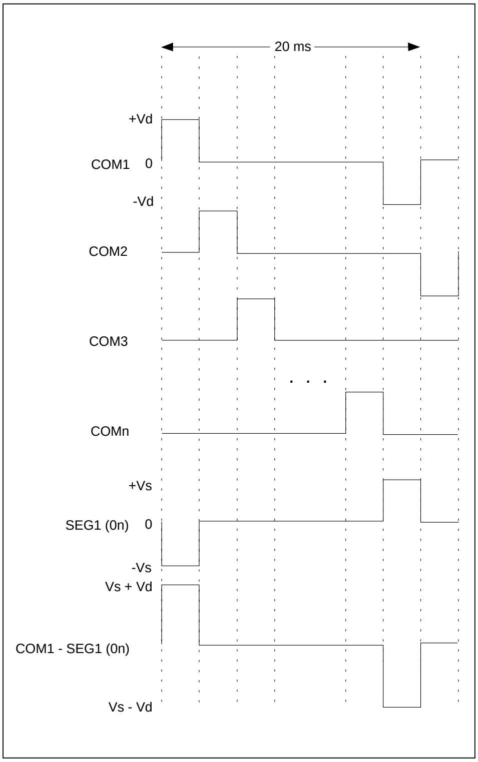


FIGURE 5: MULTIPLEXED LCD DRIVE WAVEFORMS



IMPLEMENTATION

The ideas presented in the previous section can be applied to any size multiplexed LCD display. In our implementation we used a 4-digit LCD from Ocular Inc. [1]. The circuit diagram used in this application report is shown in Figure 6. Each I/O pin on the PIC16C55 device controls the state of two segments (see Figure 6) which requires a total of 16 I/O pins. The reference voltages are generated through a simple resistive voltage divider circuit. The voltage levels are generated by taking advantage of PIC16C5X I/O pin set to input, which tristates the voltage level seen on the pin. This method uses 4 I/O pins to generate the proper voltage levels. Figure 7 shows the truth table for generating the voltage levels. Figure 8 shows how to create a bitmap for different digits. Figure 9 shows the waveforms generated for the accompanying software which implements a hexadecimal counter.

FIGURE 6: SYSTEM CONFIGURATION WITH LCD PINOUT

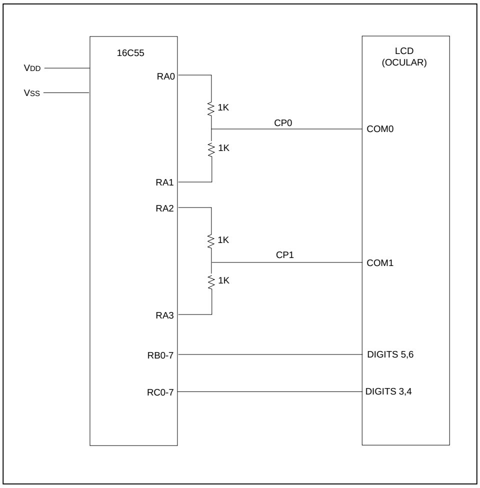
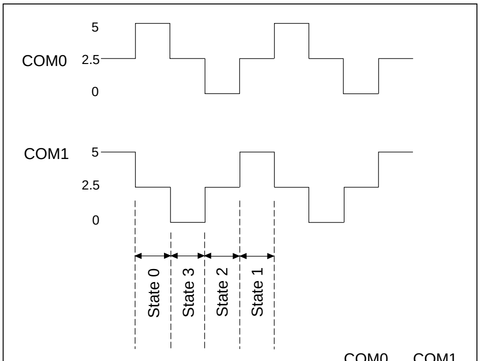


FIGURE 7: COMMON PLANE SIGNAL GENERATION

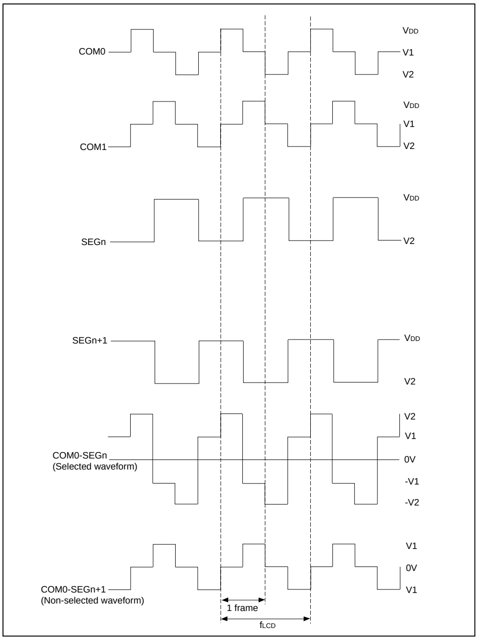


	COM0	COM1
State 0	+5V	+2.5V
State 1	+2.5V	+5V
State 2	0V	+2.5V
State 3	+2.5V	0V

FIGURE 8: LCD CHARACTER BITMAP

Digit	COM0 SEG 0				COM1 SEG 1				COM0 SEG 2				COM1 SEG3			
	F	E	D	DP	A	G	C	B	F	E	D	DP	A	G	C	B
0	0	0	0	1	0	0	1	0	1	1	1	0	1	1	0	1
1	1	1	1	1	1	0	1	0	0	0	0	0	0	1	0	1
2	1	0	0	1	0	0	0	1	0	1	1	0	1	1	1	0
3	1	1	0	1	0	0	0	0	0	0	1	0	1	1	1	1
4	0	1	1	1	1	0	0	0	1	0	0	0	0	1	1	1
5	0	1	0	1	0	1	0	0	1	0	1	0	1	0	1	1
6	1	1	1	1	1	1	1	1	1	1	1	0	1	0	1	1
7	1	1	1	1	0	0	1	0	0	0	0	0	1	1	0	1
8	0	0	0	1	0	0	0	0	1	1	1	0	1	1	1	1
9	0	1	0	1	0	0	0	0	1	0	1	0	1	1	1	1
a	0	0	1	1	0	0	0	0	1	1	0	0	1	1	1	1
b	0	0	0	1	1	1	0	0	1	1	1	0	0	0	1	1
c	1	0	0	1	1	1	0	1	0	1	1	0	0	0	1	0
d	1	0	0	1	1	0	0	0	0	1	1	0	0	1	1	1
e	0	0	0	1	0	1	0	1	1	1	1	0	1	0	1	0
f	0	0	1	1	0	1	0	1	1	1	0	0	1	0	1	0

FIGURE 9: EXAMPLE OF OUTPUT WAVEFORMS FOR DIGIT 4



CONCLUSION

In this application report we have demonstrated the use of PIC16C5X devices to implement a simple LCD controller. As discussed earlier, it is important to keep the generated DC voltage to a minimum to extend the life of the LCD. Ideally one should switch all the I/O lines simultaneously, however, a software implementation of the LCD controller will necessarily introduce a delay which is proportional to the instruction cycle of the microcontroller, as shown in Figure 10. Therefore it is necessary to keep the switching time to a minimum. Our implementation introduced less than 50 mV of DC voltage on the segment lines which is below the manufacturer's recommended DC offset voltage of 60 mV.

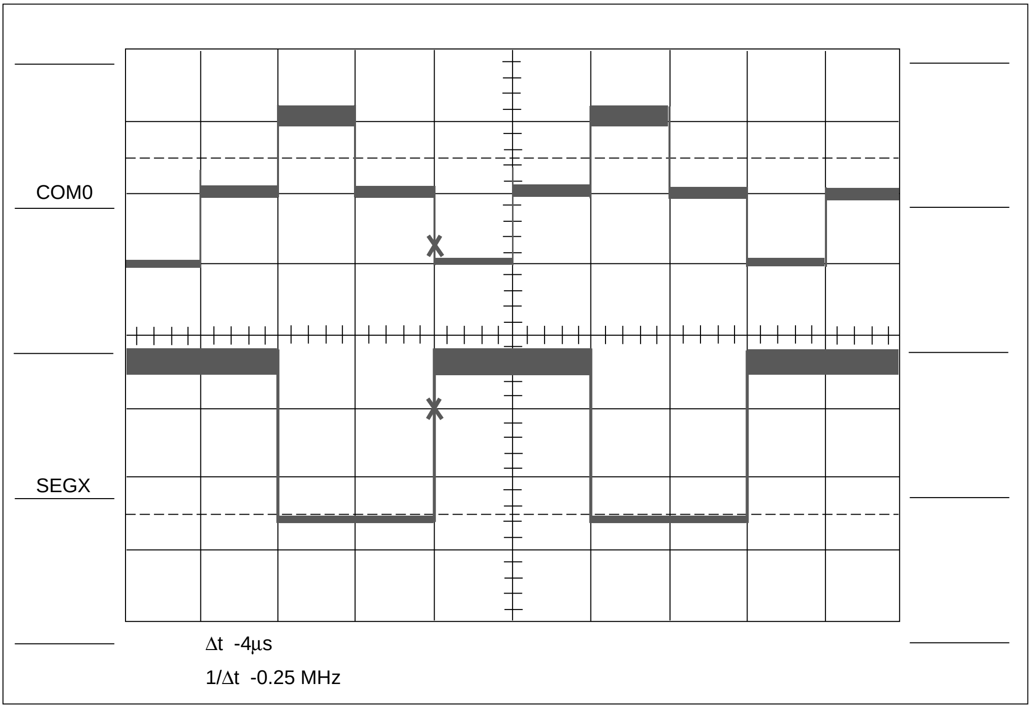
REFERENCES

[1] Ocular Inc., Drawing number JH074.

AUTHOR: Al Lovrich
Logic Products Division

2

FIGURE 10: MICROCONTROLLER GENERATED OUTPUT WAVEFORM



```
MPASM B0.54                                                                       PAGE  1



               LIST    C=132,n=0,p=16c55,r=dec

               ;*******************************************************
               ; Project: PIC16C5X as a multiplexed LCD driver.      *
               ;                                                     *
               ; Revision history:                                   *
               ;      04/14/93       original                        *
               ;*******************************************************

               ; Equates

01FF           pic54   equ    0x1ff                       ; Define Reset Vectors
01FF           pic55   equ    0x1ff
03FF           pic56   equ    0x3ff
07FF           pic57   equ    0x7ff

0001           rtcc    equ    1                           ; f1
0002           pc      equ    2                           ; f2
0003           status  equ    3                           ; f3
0004           fsr     equ    4                           ; f4

0005           porta   equ    5                           ; f5
0006           portb   equ    6                           ; f6
0007           portc   equ    7                           ; f8

               ; realtime mode registers

0008           currentState   equ    8
0009           msTimer        equ    currentState+1       ; Millisecond timer
000A           sTimerLow      equ    msTimer+1            ; Lower byte second timer
000B           sTimerHigh     equ    sTimerLow+1          ; Upper byte second timer
000C           digit56        equ    sTimerHigh+1
000D           digit34        equ    digit56+1

               ; Misc definitions

0060           FIVEMSEC       equ    96                   ; Assuming 4.096 MHz crystal

0000           w       equ    0
0001           f       equ    1

0002           z       equ    2

               ; Status register bits

               ;*******************************************************
               ; Port assignments                 *
               ;                                  *
               ;  porta - bit0: Common Plane 0    *
               ;         bit1: Common Plane 0     *
               ;         bit2: Common Plane 1     *
               ;         bit3: Common Plane 1     *
               ;                                  *
               ;  portb - bit0: 6B/DP             *
               ;         bit1: 6C/6D              *
               ;         bit2: 6G/6E              *
               ;         bit3: 6A/6F              *
               ;         bit4: 5B/DP              *
               ;         bit5: 5C/5D              *
               ;         bit6: 5G/5E              *
               ;         bit7: 5A/5F              *
               ;                                  *
               ;  portc - bit0: 4B/DP             *
```

```
                    ;        - bit1: 4C/4D         *
                    ;        - bit2: 4G/4E         *
                    ;        - bit3: 4A/4F         *
                    ;        - bit4: 3B/DP         *
                    ;        - bit5: 3C/3D         *
                    ;        - bit6: 3G/3E         *
                    ;        - bit7: 3A/3F         *
                    ;                       *
                    ;********************************************************

                    ;********************************************************
                    ;              Macro definitions      *
                    ;********************************************************
                    UpdateState     macro   State, Table

                            swapf   sTimerLow, w
                            andlw   0xf                  ; Isolate digit 5 (offset)
                            call    Table
                            movwf   digit56
                            swapf   digit56, f

                            movf    sTimerLow, w
                            andlw    0xf                 ; Isolate digit 6 (offset)
                            call    Table
                            iorwf   digit56, f

                            swapf   sTimerHigh, w
                            andlw   0xf                  ; Isolate digit 5 (offset)
                            call    Table
                            movwf   digit34
                            swapf   digit34, f

                            movf    sTimerHigh, w
                            andlw   0xf                  ; Isolate digit 6 (offset)
                            call    Table
                            iorwf   digit34, f

                            movf    digit34, w           ; Display digits 3 & 4
                            movwf   portc
                            movf    digit56, w
                            movwf   portb                ; Display digits 5 & 6
                            endm

                            org     0

                    ; Initialize ports A, B, and C and RTCC. In case of output data
                    ; values, set the data latch first, then set the port direction.

                    Initialize
0000 0C01           movlw    00000001b                ; Set data latch
0001 0025           movwf    porta
0002 0C08           movlw    00001000b                ; Set I/O direction
0003 0005           tris     porta

0004 0C00           movlw    00000000b                ; Set levels to low
0005 0026           movwf    portb
0006 0C00           movlw    00000000b                ; Set as outputs
0007 0006           tris     portb

0008 0C00           movlw    00000000b                ; Set levels to low
0009 0027           movwf    portc
000A 0C00           movlw    00000000b                ; Set as outputs
000B 0007           tris     portc

000C 0C04           movlw    0x04                     ; Set prescaler
000D 0002           option

000E 0C60           movlw    FIVEMSEC                 ; rtcc = 5ms
000F 0021           movwf    rtcc
```

```
0010 0C04             movlw   4
0011 0028             movwf   currentState

0012 0C0D             movlw   0xd
0013 0029             movwf   msTimer                      ; Initialize millisecond timer

0014 006A             clrf    sTimerLow                    ; Clear second counter
0015 006B             clrf    sTimerHigh

0016 0800             retlw    0

               ; Check timer register for timing out (rtcc = 0). Remain in the
               ; loop until the timer times out.


               ; Wait for 5ms timer timeout

               Timer_Check
0017 0201             movf    rtcc, w
0018 0743             btfss   status, z
0019 0A17             goto    Timer_Check

001A 0C60             movlw   FIVEMSEC
001B 0021             movwf   rtcc

001C 02E9             decfsz  msTimer, f
001D 0A21             goto    Update_Backplane

001E 03EA             incfsz  sTimerLow                    ; Update second counter
001F 0A21             goto    Update_Backplane
0020 02AB             incf    sTimerHigh, f

               ; RA0 and RA1 are used to control voltage level for common plane 0.
               ; RA2 and RA3 are used to control voltage level for common plane 1.
               ; There are four possible states with different voltage levels as
               ; follows:
               ;
               ; State 0 - cp0 = +5v     ra0=1,  ra1=x
               ;           cp1 = +2.5v   ra2=1,  ra3=0
               ; State 1 - cp0 = +2.5v   ra0=1,  ra1=0
               ;           cp1 = +5v     ra2=1,  ra3=x
               ; State 2 - cp0 =  0v     ra0=0,  ra1=x
               ;           cp1 = +2.5v   ra2=1,  ra3=0
               ; State 3 - cp0 = +2.5v   ra0=1,  ra1=0
               ;           cp1 =  0v     ra2=0,  ra3=x


               Update_Backplane
0021 0004             clrwdt                               ; Reset watchdog timer

0022 00C8             decf    currentState, w              ; Update w register
0023 0E03             andlw   0x03                         ; Use only bit0/1
0024 0028             movwf   currentState                 ; Update currentState
0025 01E2             addwf   pc, f

0026 0AAD             goto    State3
0027 0A81             goto    State2
0028 0A55             goto    State1
               ;      goto    State0

               ;              State 0

                              State0
               UpdateState    State0, S0_Table

0029 038A             swapf   sTimerLow, w
002A 0E0F             andlw   0xf                  ; Isolate digit 5 (offset)
002B 0944             call    S0_Table
```

```
002C 002C              movwf   digit56
002D 03AC              swapf   digit56, f

002E 020A              movf    sTimerLow, w
002F 0E0F              andlw   0xf                  ; Isolate digit 6 (offset)
0030 0944              call    S0_Table
0031 012C              iorwf   digit56, f

0032 038B              swapf   sTimerHigh, w
0033 0E0F              andlw   0xf                  ; Isolate digit 5 (offset)
0034 0944              call    S0_Table
0035 002D              movwf   digit34
0036 03AD              swapf   digit34, f

0037 020B              movf    sTimerHigh, w
0038 0E0F              andlw   0xf                  ; Isolate digit 6 (offset)
0039 0944              call    S0_Table
003A 012D              iorwf   digit34, f

003B 020D              movf    digit34, w           ; Display digits 3 & 4
003C 0027              movwf   portc
003D 020C              movf    digit56, w
003E 0026              movwf   portb                ; Display digits 5 & 6

003F 0C05              movlw   00000101b
0040 0025              movwf   porta
0041 0C02              movlw   00000010b
0042 0005              tris    porta

0043 0800              retlw   0

               S0_Table
0044 01E2              addwf    pc, f               ; Add offset to pc

0045 0804              retlw   0100b                ; 0
0046 080C              retlw   1100b                ; 1
0047 0802              retlw   0010b                ; 2
0048 0800              retlw   0000b                ; 3
0049 0808              retlw   1000b                ; 4
004A 0801              retlw   0001b                ; 5
004B 080F              retlw   1111b                ; 6
004C 0804              retlw   0100b                ; 7
004D 0800              retlw   0000b                ; 8
004E 0800              retlw   0000b                ; 9
004F 0800              retlw   0000b                ; a
0050 0809              retlw   1001b                ; b
0051 080B              retlw   1011b                ; c
0052 0808              retlw   1000b                ; d
0053 0803              retlw   0011b                ; e
0054 0803              retlw   0011b                ; f

               ; State 1

               State1
                            UpdateState      State1, S1_Table

0055 038A              swapf   sTimerLow, w
0056 0E0F              andlw   0xf                  ; Isolate digit 5 (offset)
0057 0970              call    S1_Table
0058 002C              movwf   digit56
0059 03AC              swapf   digit56, f

005A 020A              movf    sTimerLow, w
005B 0E0F              andlw   0xf                  ; Isolate digit 6 (offset)
005C 0970              call    S1_Table
005D 012C              iorwf   digit56, f

005E 038B              swapf   sTimerHigh, w
```

```
005F 0E0F            andlw   0xf                    ; Isolate digit 5 (offset)
0060 0970            call    S1_Table
0061 002D            movwf   digit34
0062 03AD            swapf   digit34, f

0063 020B            movf    sTimerHigh, w
0064 0E0F            andlw   0xf                    ; Isolate digit 6 (offset)
0065 0970            call    S1_Table
0066 012D            iorwf   digit34, f

0067 020D            movf    digit34, w             ; Display digits 3 & 4
0068 0027            movwf   portc
0069 020C            movf    digit56, w
006A 0026            movwf   portb                  ; Display digits 5 & 6



006B 0C05            movlw   00000101b
006C 0025            movwf   porta
006D 0C08            movlw   00001000b
006E 0005            tris    porta

006F 0800            retlw   0

               S1_Table
0070 01E2            addwf    pc, f

0071 0801            retlw   0001b                  ; 0
0072 080F            retlw   1111b                  ; 1
0073 0809            retlw   1001b                  ; 2
0074 080D            retlw   1101b                  ; 3
0075 0807            retlw   0111b                  ; 4
0076 0805            retlw   0101b                  ; 5
0077 080F            retlw   1111b                  ; 6
0078 080F            retlw   1111b                  ; 7
0079 0801            retlw   0001b                  ; 8
007A 0805            retlw   0101b                  ; 9
007B 0803            retlw   0011b                  ; a
007C 0801            retlw   0001b                  ; b
007D 0809            retlw   1001b                  ; c
007E 0809            retlw   1001b                  ; d
007F 0801            retlw   0001b                  ; e
0080 0803            retlw   0011b                  ; f

               ; State 2

               State2
               UpdateState     State2, S2_Table

0081 038A            swapf   sTimerLow, w
0082 0E0F            andlw   0xf                    ; Isolate digit 5 (offset)
0083 099C            call    S2_Table
0084 002C            movwf   digit56
0085 03AC            swapf   digit56, f

0086 020A            movf    sTimerLow, w
0087 0E0F            andlw   0xf                    ; Isolate digit 6 (offset)
0088 099C            call    S2_Table
0089 012C            iorwf   digit56, f

008A 038B            swapf   sTimerHigh, w
008B 0E0F            andlw   0xf                    ; Isolate digit 5 (offset)
008C 099C            call    S2_Table
008D 002D            movwf   digit34
008E 03AD            swapf   digit34, f

008F 020B            movf    sTimerHigh, w
0090 0E0F            andlw   0xf                    ; Isolate digit 6 (offset)
0091 099C            call    S2_Table
```

```
0092 012D            iorwf   digit34, f

0093 020D            movf    digit34, w          ; Display digits 3 & 4
0094 0027            movwf    portc
0095 020C            movf    digit56, w
0096 0026            movwf   portb               ; Display digits 5 & 6


0097 0C04            movlw   00000100b
0098 0025            movwf   porta
0099 0C02            movlw   00000010b
009A 0005            tris    porta

009B 0800            retlw   0

               S2_Table
009C 01E2            addwf   pc, f

009D 080B            retlw   1011b   ; 0
009E 0803            retlw   0011b   ; 1
009F 080D            retlw   1101b   ; 2
00A0 080F            retlw   1111b   ; 3
00A1 0807            retlw   0111b   ; 4
00A2 080E            retlw   1110b   ; 5
00A3 080E            retlw   1110b   ; 6
00A4 080B            retlw   1011b   ; 7
00A5 080F            retlw   1111b   ; 8
00A6 080F            retlw   1111b   ; 9
00A7 080F            retlw   1111b   ; a
00A8 0806            retlw   0110b   ; b
00A9 0804            retlw   0100b   ; c
00AA 0807            retlw   0111b   ; d
00AB 080C            retlw   1100b   ; e
00AC 080C            retlw   1100b   ; f

               ; State 3

               State3
               UpdateState     State3, S3_Table

00AD 038A            swapf   sTimerLow, w
00AE 0E0F            andlw   0xf                 ; Isolate digit 5 (offset)
00AF 09C8            call    S3_Table
00B0 002C            movwf   digit56
00B1 03AC            swapf   digit56, f

00B2 020A            movf    sTimerLow, w
00B3 0E0F            andlw   0xf                 ; Isolate digit 6 (offset)
00B4 09C8            call    S3_Table
00B5 012C            iorwf   digit56, f

00B6 038B            swapf   sTimerHigh, w
00B7 0E0F            andlw   0xf                 ; Isolate digit 5 (offset)
00B8 09C8            call    S3_Table
00B9 002D            movwf   digit34
00BA 03AD            swapf   digit34, f

00BB 020B            movf    sTimerHigh, w
00BC 0E0F            andlw   0xf                 ; Isolate digit 6 (offset)
00BD 09C8            call    S3_Table
00BE 012D            iorwf   digit34, f

00BF 020D            movf    digit34, w          ; Display digits 3 & 4
00C0 0027            movwf   portc
00C1 020C            movf    digit56, w
00C2 0026            movwf   portb          ; Display digits 5 & 6
```

```
00C3 0C01               movlw   00000001b
00C4 0025               movwf   porta
00C5 0C08               movlw   00001000b
00C6 0005               tris    porta

00C7 0800               retlw   0

               S3_Table
00C8 01E2               addwf   pc, f

00C9 080E               retlw   1110b           ; 0
00CA 0800               retlw   0000b           ; 1
00CB 0806               retlw   0110b           ; 2
00CC 0802               retlw   0010b           ; 3
00CD 0808               retlw   1000b           ; 4
00CE 080A               retlw   1010b           ; 5
00CF 080E               retlw   1110b           ; 6
00D0 0800               retlw   0000b           ; 7
00D1 080E               retlw   1110b           ; 8
00D2 080A               retlw   1010b           ; 9
00D3 080C               retlw   1100b           ; a
00D4 080E               retlw   1110b           ; b
00D5 0806               retlw   0110b           ; c
00D6 0806               retlw   0110b           ; d
00D7 080E               retlw   1110b           ; e
00D8 080C               retlw   1100b           ; f

               ; Main code

               Start
00D9 0900               call    Initialize
               Repeat
00DA 0917               call    Timer_Check
00DB 0ADA               goto    Repeat

                        org     pic55

               System_Reset
01FF 0AD9                      goto    Start

                    END

Errors   :    0
Warnings :    0
```

WORLDWIDE SALES & SERVICE

AMERICAS

Corporate Office
Microchip Technology Inc.
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 602 786-7200 Fax: 602 786-7277
Technical Support: 602 786-7627
Web: http://www.mchip.com/microhip

Atlanta
Microchip Technology Inc.
500 Sugar Mill Road, Suite 200B
Atlanta, GA 30350
Tel: 770 640-0034 Fax: 770 640-0307

Boston
Microchip Technology Inc.
5 Mount Royal Avenue
Marlborough, MA 01752
Tel: 508 480-9990 Fax: 508 480-8575

Chicago
Microchip Technology Inc.
333 Pierce Road, Suite 180
Itasca, IL 60143
Tel: 708 285-0071 Fax: 708 285-0075

Dallas
Microchip Technology Inc.
14651 Dallas Parkway, Suite 816
Dallas, TX 75240-8809
Tel: 214 991-7177 Fax: 214 991-8588

Dayton
Microchip Technology Inc.
35 Rockridge Road
Englewood, OH 45322
Tel: 513 832-2543 Fax: 513 832-2841

Los Angeles
Microchip Technology Inc.
18201 Von Karman, Suite 455
Irvine, CA 92715
Tel: 714 263-1888 Fax: 714 263-1338

New York
Microchip Technology Inc.
150 Motor Parkway, Suite 416
Hauppauge, NY 11788
Tel: 516 273-5305 Fax: 516 273-5335

AMERICAS (continued)

San Jose
Microchip Technology Inc.
2107 North First Street, Suite 590
San Jose, CA 95131
Tel: 408 436-7950 Fax: 408 436-7955

ASIA/PACIFIC

Hong Kong
Microchip Technology
Unit No. 3002-3004, Tower 1
Metroplaza
223 Hing Fong Road
Kwai Fong, N.T. Hong Kong
Tel: 852 2 401 1200 Fax: 852 2 401 3431

Korea
Microchip Technology
168-1, Youngbo Bldg. 3 Floor
Samsung-Dong, Kangnam-Ku,
Seoul, Korea
Tel: 82 2 554 7200 Fax: 82 2 558 5934

Singapore
Microchip Technology
200 Middle Road
#10-03 Prime Centre
Singapore 188980
Tel: 65 334 8870 Fax: 65 334 8850

Taiwan
Microchip Technology
10F-1C 207
Tung Hua North Road
Taipei, Taiwan, ROC
Tel: 886 2 717 7175 Fax: 886 2 545 0139

EUROPE

United Kingdom
Arizona Microchip Technology Ltd.
Unit 6, The Courtyard
Meadow Bank, Furlong Road
Bourne End, Buckinghamshire SL8 5AJ
Tel: 44 0 1628 851077 Fax: 44 0 1628 850259

France
Arizona Microchip Technology SARL
2 Rue du Buisson aux Fraises
91300 Massy - France
Tel: 33 1 69 53 63 20 Fax: 33 1 69 30 90 79

Germany
Arizona Microchip Technology GmbH
Gustav-Heinemann-Ring 125
D-81739 Muenchen, Germany
Tel: 49 89 627 144 0 Fax: 49 89 627 144 44

Italy
Arizona Microchip Technology SRL
Centro Direzionale Colleoni
Palazzo Pegaso Ingresso No. 2
Via Paracelso 23, 20041
Agrate Brianza (MI) Italy
Tel: 39 039 689 9939 Fax: 39 039 689 9883

JAPAN

Microchip Technology Intl. Inc.
Benex S-1 6F
3-18-20, Shin Yokohama
Kohoku-Ku, Yokohama
Kanagawa 222 Japan
Tel: 81 45 471 6166 Fax: 81 45 471 6122

9/22/95