AN563

Using PIC16C5X Microcontrollers as LCD Drivers

Author: *Al Lovrich*
Microchip Technology Inc.

INTRODUCTION

This application note 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 modeled 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 (V_{RMS}) levels. When a V_{RMS} level of zero volts is applied to the LCD, the LCD is practically transparent.

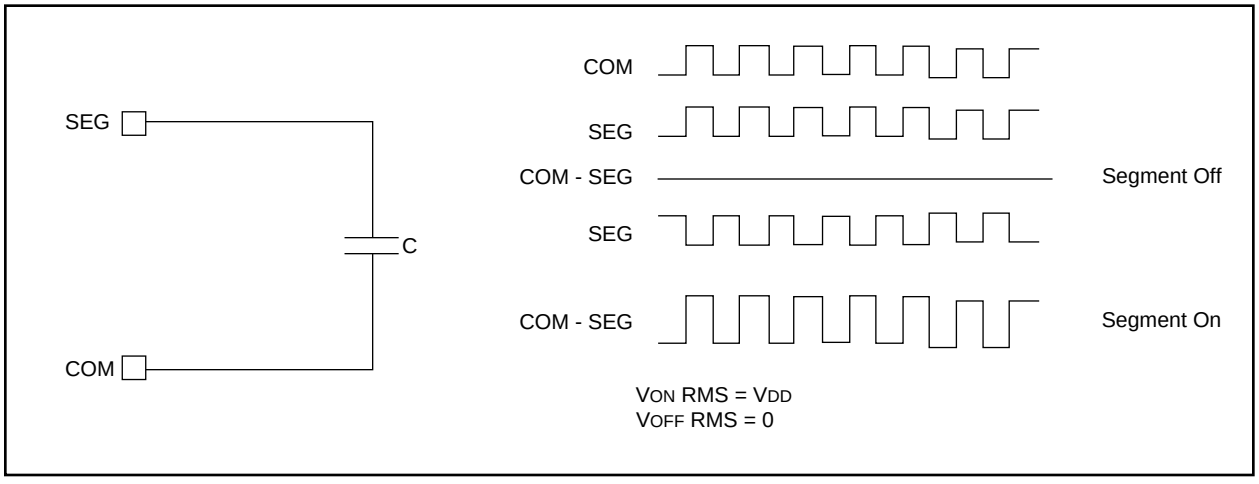
To turn an LCD segment "on," which makes the segment turn dark or opaque, an LCD RMS voltage that is greater than the LCD threshold voltage is applied to the LCD. The RMS LCD voltage is the RMS voltage across the capacitor *C* in Figure 1, which is equal to the potential difference between SEG and COM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

Different LCDs have different characteristics; Figure 2 shows typical voltage vs relative contrast characteristics. Notations on the curve show operating points for multiplex operation with the threshold voltage set to 1.7 V_{RMS}. This voltage is often used as the measure of voltage for the 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. This is achieved by applying a voltage to the LCD that is greater than the LCDs threshold voltage. 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 and multiplexed.

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 the 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 backplanes. Typically, FFRAME ranges from 25 Hz to 300 Hz. The most commonly used frame frequency is 40-70 Hz. A lower frequency would result in flicker effects and higher frequency would increase power requirements.

Multiplexed LCDs maintain their liquid crystal characteristics. They have a low power consumption, a high contrast ratio under high ambient light levels, and they 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 note 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 CHARATERISTICS

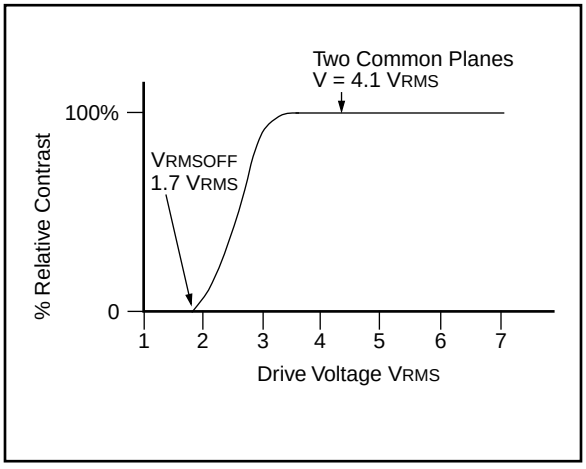
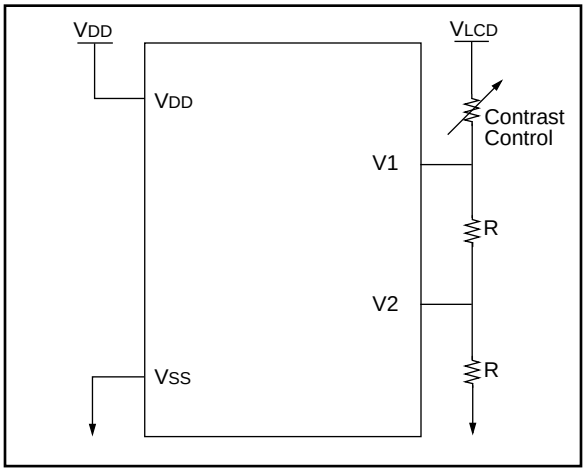


FIGURE 3: CONTRAST CONTROL CIRCUIT



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 V_D, such that $V_S + V_D$ turns the segment on and $V_S - V_D$ turns the segment off. Note that the segment signal V_D 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 V_{RMS} present on the unaddressed segments.

The symmetrical nature of the waveforms theoretically result in 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

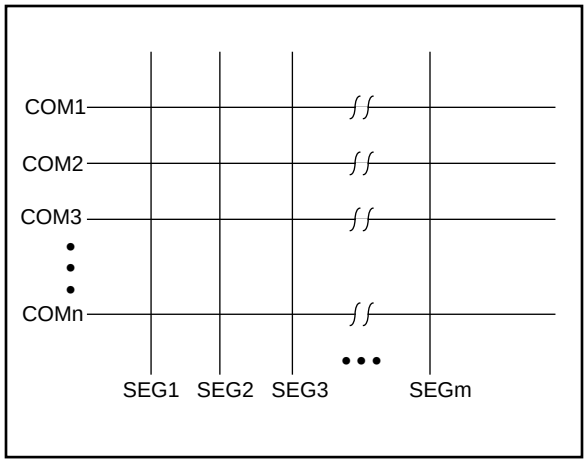
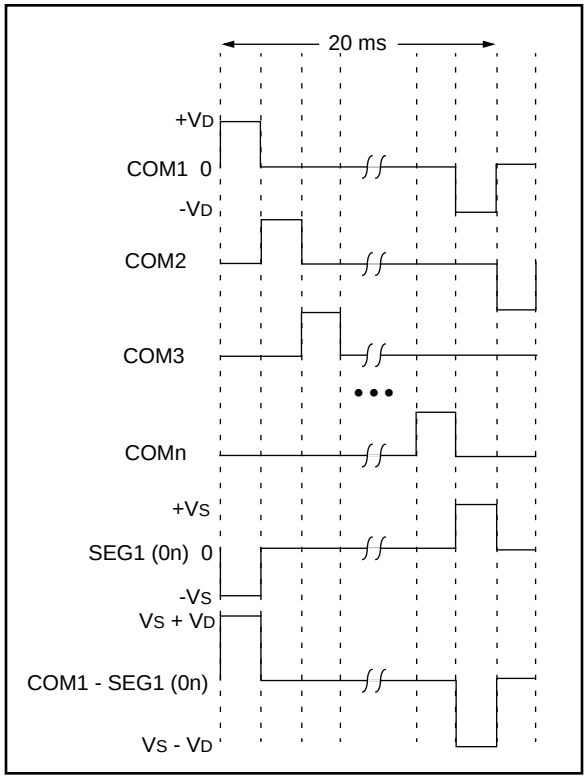


FIGURE 5: MULTIPLEXED LCD DRIVE WAVEFORMS



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 note is shown in Figure 6. Each I/O pin on the PIC16C55 device controls the state of two segments (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 tri-states 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

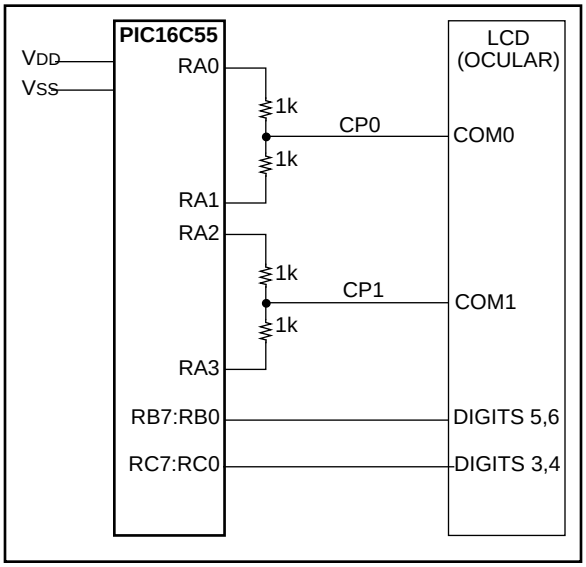


FIGURE 7: COMMON PLANE SIGNAL GENERATION

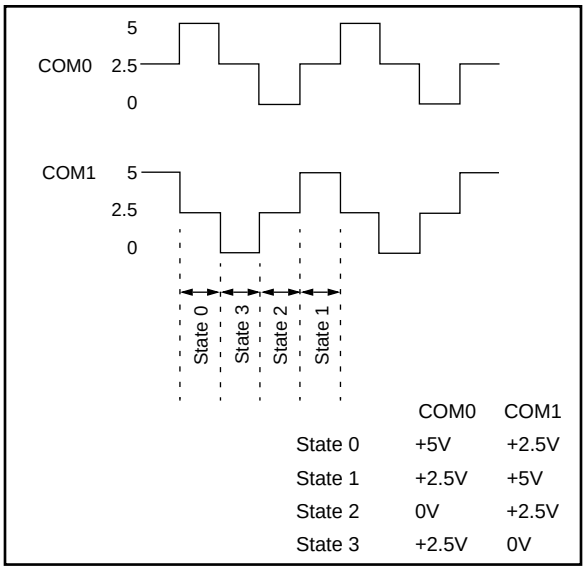
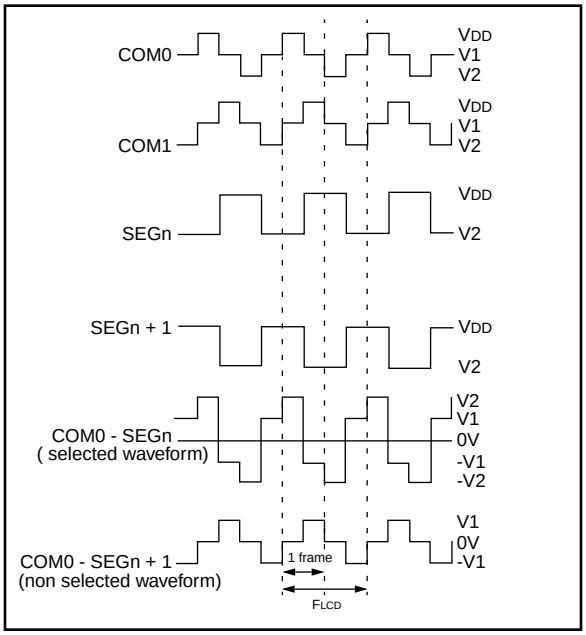


FIGURE 8: LCD CHARACTER BITMAP

Digit	COM0 SEG0				COM1 SEG1				COM0 SEG2				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



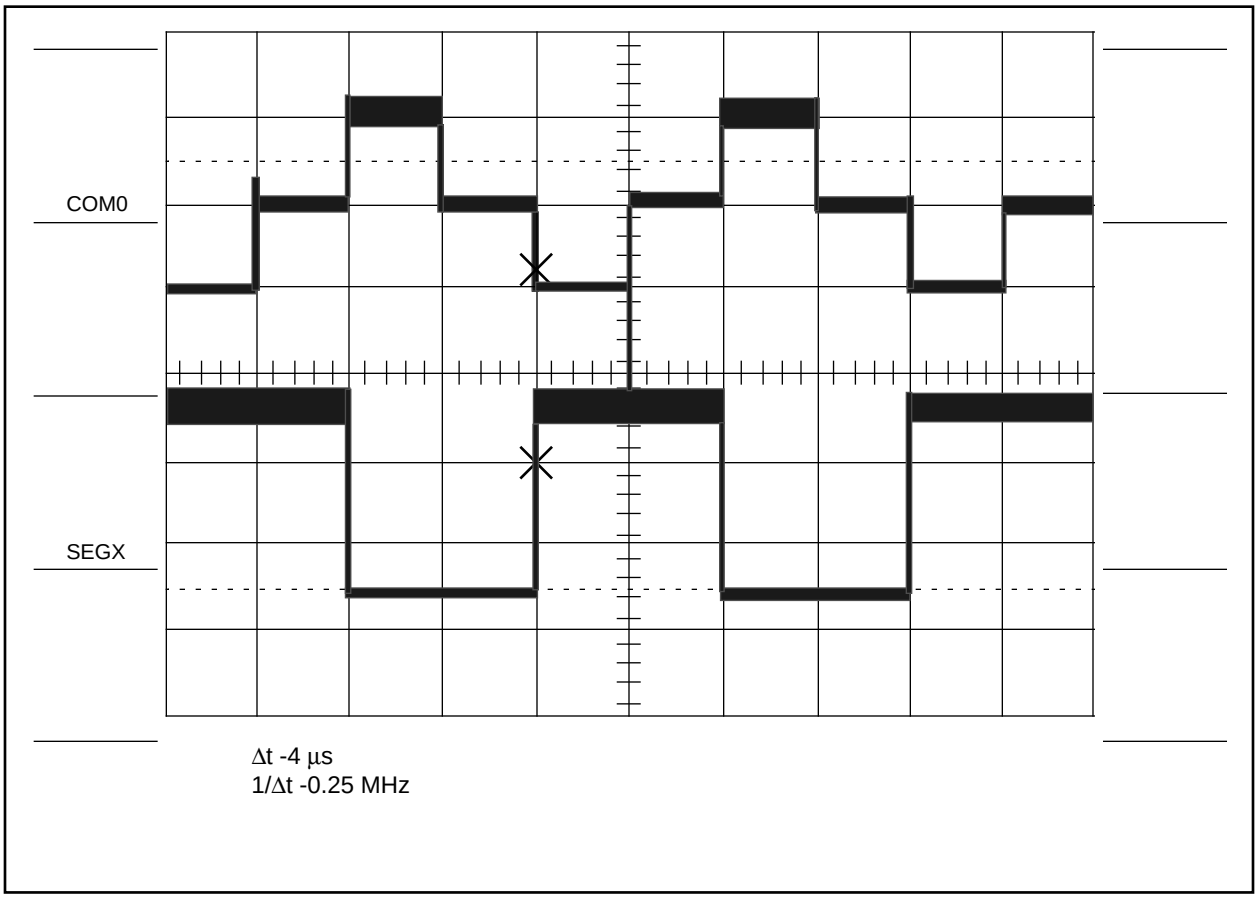
CONCLUSION

In this application note 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, implementation of software 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.

FIGURE 10: MICROCONTROLLER GENERATED OUTPUT WAVEFORM



Please check the Microchip BBS for the latest version of the source code. Microchip's Worldwide Web Address: www.microchip.com; Bulletin Board Support: MCHIPBBS using CompuServe® (CompuServe membership not required).

APPENDIX A: PIC16C5X AS A MULTIPLEXED LCD DRIVER

```
MPASM 01.40 Released             LCD.ASM   1-16-1997  16:45:44         PAGE  1


LOC  OBJECT CODE     LINE SOURCE TEXT
  VALUE

                     00001  LIST    C=132,n=0,p=16c55,r=dec
                     00002
                     00003 ;*******************************************************
                     00004 ; Project: PIC16C5X as a multiplexed LCD driver.          *
                     00005 ;                                                        *
                     00006 ; Revision history:                                      *
                     00007 ;       04/14/93        original                         *
                     00008 ;       1-15-97         Compatability with MPASMWIN 1.40 *
                     00009 ;*******************************************************
                     00010
                     00011 ; Equates
                     00012
  000001FF           00013 pic54        equ  0x1ff           ; Define Reset Vectors
  000001FF           00014 pic55        equ  0x1ff
  000003FF           00015 pic56        equ  0x3ff
  000007FF           00016 pic57        equ  0x7ff
                     00017
  00000001           00018 TMR0         equ  1               ; f1
  00000002           00019 pc           equ  2               ; f2
  00000003           00020 status       equ  3               ; f3
  00000004           00021 fsr          equ  4               ; f4
                     00022
  00000005           00023 porta        equ  5               ; f5
  00000006           00024 portb        equ  6               ; f6
  00000007           00025 portc        equ  7               ; f8
                     00026
                     00027 ; realtime mode registers
                     00028
  00000008           00029 currentState equ  8
  00000009           00030 msTimer      equ  currentState+1  ; Millisecond timer
  0000000A           00031 sTimerLow    equ  msTimer+1       ; Lower byte second timer
  0000000B           00032 sTimerHigh   equ  sTimerLow+1     ; Upper byte second timer
  0000000C           00033 digit56      equ  sTimerHigh+1
  0000000D           00034 digit34      equ  digit56+1
                     00035
                     00036 ; Misc definitions
                     00037
  00000060           00038 FIVEMSEC     equ  96              ; Assuming 4.096 MHz crystal
                     00039
  00000000           00040 w            equ  0
  00000001           00041 f            equ  1
                     00042
  00000002           00043 z            equ  2
                     00044
                     00045 ; Status register bits
                     00046
                     00047 ;*************************************************
                     00048 ; Port assignments                                *
                     00049 ;                                                 *
                     00050 ;  porta - bit0: Common Plane 0                  *
                     00051 ;          bit1: Common Plane 0                   *
                     00052 ;          bit2: Common Plane 1                   *
                     00053 ;          bit3: Common Plane 1                   *
                     00054 ;                                                 *
```

```
                    00055 ;  portb - bit0: 6B/DP                                *
                    00056 ;          bit1: 6C/6D                                *
                    00057 ;          bit2: 6G/6E                                *
                    00058 ;          bit3: 6A/6F                                *
                    00059 ;          bit4: 5B/DP                                *
                    00060 ;          bit5: 5C/5D                                *
                    00061 ;          bit6: 5G/5E                                *
                    00062 ;          bit7: 5A/5F                                *
                    00063 ;                                                     *
                    00064 ;  portc - bit0: 4B/DP                                *
                    00065 ;        - bit1: 4C/4D                                *
                    00066 ;        - bit2: 4G/4E                                *
                    00067 ;        - bit3: 4A/4F                                *
                    00068 ;        - bit4: 3B/DP                                *
                    00069 ;        - bit5: 3C/3D                                *
                    00070 ;        - bit6: 3G/3E                                *
                    00071 ;        - bit7: 3A/3F                                *
                    00072 ;                                                     *
                    00073 ;*****************************************************
                    00074
                    00075 ;*****************************************************
                    00076 ;               Macro definitions                     *
                    00077 ;*****************************************************
                    00078 UpdateState     macro   State, Table
                    00079
                    00080         swapf   sTimerLow, w
                    00081         andlw   0xf               ; Isolate digit 5 (offset)
                    00082         call    Table
                    00083         movwf   digit56
                    00084         swapf   digit56, f
                    00085
                    00086         movf    sTimerLow, w
                    00087         andlw   0xf               ; Isolate digit 6 (offset)
                    00088         call    Table
                    00089         iorwf   digit56, f
                    00090
                    00091         swapf   sTimerHigh, w
                    00092         andlw   0xf               ; Isolate digit 5 (offset)
                    00093         call    Table
                    00094         movwf   digit34
                    00095         swapf   digit34, f
                    00096
                    00097         movf    sTimerHigh, w
                    00098         andlw   0xf               ; Isolate digit 6 (offset)
                    00099         call    Table
                    00100         iorwf   digit34, f
                    00101
                    00102         movf    digit34, w        ; Display digits 3 & 4
                    00103         movwf   portc
                    00104         movf    digit56, w
                    00105         movwf   portb             ; Display digits 5 & 6
                    00106         endm
                    00107
0000                00108         org     0
                    00109
                    00110 ; Initialize ports A, B, and C and TMR0. In case of output data
                    00111 ; values, set the data latch first, then set the port direction.
                    00112
0000                00113 Initialize
0000 0C01           00114         movlw   00000001b         ; Set data latch
0001 0025           00115         movwf   porta
0002 0C08           00116         movlw   00001000b         ; Set I/O direction
0003 0005           00117         tris    porta
                    00118
0004 0C00           00119         movlw   00000000b         ; Set levels to low
0005 0026           00120         movwf   portb
```

```
0006 0C00           00121         movlw   00000000b         ; Set as outputs
0007 0006           00122         tris    portb
                    00123
0008 0C00           00124         movlw   00000000b         ; Set levels to low
0009 0027           00125         movwf   portc
000A 0C00           00126         movlw   00000000b         ; Set as outputs
000B 0007           00127         tris    portc
                    00128
000C 0C04           00129         movlw   0x04              ; Set prescaler
000D 0002           00130         option
                    00131
000E 0C60           00132         movlw   FIVEMSEC          ; TMR0 = 5ms
000F 0021           00133         movwf   TMR0
                    00134
0010 0C04           00135         movlw   4
0011 0028           00136         movwf   currentState
                    00137
0012 0C0D           00138         movlw   0xd
0013 0029           00139         movwf   msTimer           ; Initialize millisecond timer
                    00140
0014 006A           00141         clrf    sTimerLow         ; Clear second counter
0015 006B           00142         clrf    sTimerHigh
                    00143
0016 0800           00144         retlw   0
                    00145
                    00146 ; Check timer register for timing out (TMR0 = 0). Remain in the
                    00147 ; loop until the timer times out.
                    00148
                    00149
                    00150 ; Wait for 5ms timer timeout
                    00151
0017                00152 Timer_Check
0017 0201           00153         movf    TMR0, w
0018 0743           00154         btfss   status, z
0019 0A17           00155         goto    Timer_Check
                    00156
001A 0C60           00157         movlw   FIVEMSEC
001B 0021           00158         movwf   TMR0
                    00159
001C 02E9           00160         decfsz  msTimer, f
001D 0A21           00161         goto    Update_Backplane
                    00162
001E 03EA           00163         incfsz  sTimerLow, f      ; Update second counter
001F 0A21           00164         goto    Update_Backplane
0020 02AB           00165         incf    sTimerHigh, f
                    00166
                    00167 ; RA0 and RA1 are used to control voltage level for common plane 0.
                    00168 ; RA2 and RA3 are used to control voltage level for common plane 1.
                    00169 ; There are four possible states with different voltage levels as
                    00170 ; follows:
                    00171 ;
                    00172 ; State 0 - cp0 = +5v           ra0=1, ra1=x
                    00173 ;           cp1 = +2.5v         ra2=1, ra3=0
                    00174 ; State 1 - cp0 = +2.5v         ra0=1, ra1=0
                    00175 ;           cp1 = +5v           ra2=1, ra3=x
                    00176 ; State 2 - cp0 =  0v           ra0=0, ra1=x
                    00177 ;           cp1 = +2.5v         ra2=1, ra3=0
                    00178 ; State 3 - cp0 = +2.5v         ra0=1, ra1=0
                    00179 ;           cp1 =  0v           ra2=0, ra3=x
                    00180
                    00181
0021                00182 Update_Backplane
0021 0004           00183         clrwdt                    ; Reset watchdog timer
                    00184
0022 00C8           00185         decf    currentState, w   ; Update w register
0023 0E03           00186         andlw   0x03              ; Use only bit0/1
```

```
0024 0028           00187         movwf   currentState      ; Update currentState
0025 01E2           00188         addwf   pc, f
                    00189
0026 0AAD           00190         goto    State3
0027 0A81           00191         goto    State2
0028 0A55           00192         goto    State1
                    00193 ;       goto    State0
                    00194
                    00195 ; State 0
                    00196
0029                00197 State0
                    00198         UpdateState     State0, S0_Table
                        M
0029 038A               M         swapf   sTimerLow, w
002A 0E0F               M         andlw   0xf               ; Isolate digit 5 (offset)
002B 0944               M         call    S0_Table
002C 002C               M         movwf   digit56
002D 03AC               M         swapf   digit56, f
                        M
002E 020A               M         movf    sTimerLow, w
002F 0E0F               M         andlw   0xf               ; Isolate digit 6 (offset)
0030 0944               M         call    S0_Table
0031 012C               M         iorwf   digit56, f
                        M
0032 038B               M         swapf   sTimerHigh, w
0033 0E0F               M         andlw   0xf               ; Isolate digit 5 (offset)
0034 0944               M         call    S0_Table
0035 002D               M         movwf   digit34
0036 03AD               M         swapf   digit34, f
                        M
0037 020B               M         movf    sTimerHigh, w
0038 0E0F               M         andlw   0xf               ; Isolate digit 6 (offset)
0039 0944               M         call    S0_Table
003A 012D               M         iorwf   digit34, f
                        M
003B 020D               M         movf    digit34, w        ; Display digits 3 & 4
003C 0027               M         movwf   portc
003D 020C               M         movf    digit56, w
003E 0026               M         movwf   portb             ; Display digits 5 & 6
                    00199
003F 0C05           00200         movlw   00000101b
0040 0025           00201         movwf   porta
0041 0C02           00202         movlw   00000010b
0042 0005           00203         tris    porta
                    00204
0043 0800           00205         retlw   0
                    00206
0044                00207 S0_Table
0044 01E2           00208         addwf   pc, f             ; Add offset to pc
                    00209
0045 0804           00210         retlw   0100b             ; 0
0046 080C           00211         retlw   1100b             ; 1
0047 0802           00212         retlw   0010b             ; 2
0048 0800           00213         retlw   0000b             ; 3
0049 0808           00214         retlw   1000b             ; 4
004A 0801           00215         retlw   0001b             ; 5
004B 080F           00216         retlw   1111b             ; 6
004C 0804           00217         retlw   0100b             ; 7
004D 0800           00218         retlw   0000b             ; 8
004E 0800           00219         retlw   0000b             ; 9
004F 0800           00220         retlw   0000b             ; a
0050 0809           00221         retlw   1001b             ; b
0051 080B           00222         retlw   1011b             ; c
0052 0808           00223         retlw   1000b             ; d
0053 0803           00224         retlw   0011b             ; e
0054 0803           00225         retlw   0011b             ; f
```

```
                  00226
                  00227 ; State 1
                  00228
0055              00229 State1
                  00230         UpdateState     State1, S1_Table
                      M
0055 038A             M         swapf   sTimerLow, w
0056 0E0F             M         andlw   0xf               ; Isolate digit 5 (offset)
0057 0970             M         call    S1_Table
0058 002C             M         movwf   digit56
0059 03AC             M         swapf   digit56, f
                      M
005A 020A             M         movf    sTimerLow, w
005B 0E0F             M         andlw   0xf               ; Isolate digit 6 (offset)
005C 0970             M         call    S1_Table
005D 012C             M         iorwf   digit56, f
                      M
005E 038B             M         swapf   sTimerHigh, w
005F 0E0F             M         andlw   0xf               ; Isolate digit 5 (offset)
0060 0970             M         call    S1_Table
0061 002D             M         movwf   digit34
0062 03AD             M         swapf   digit34, f
                      M
0063 020B             M         movf    sTimerHigh, w
0064 0E0F             M         andlw   0xf               ; Isolate digit 6 (offset)
0065 0970             M         call    S1_Table
0066 012D             M         iorwf   digit34, f
                      M
0067 020D             M         movf    digit34, w        ; Display digits 3 & 4
0068 0027             M         movwf   portc
0069 020C             M         movf    digit56, w
006A 0026             M         movwf   portb             ; Display digits 5 & 6
                  00231
                  00232
006B 0C05         00233         movlw   00000101b
006C 0025         00234         movwf   porta
006D 0C08         00235         movlw   00001000b
006E 0005         00236         tris    porta
                  00237
006F 0800         00238         retlw   0
                  00239
0070              00240 S1_Table
0070 01E2         00241         addwf   pc, f
                  00242
0071 0801         00243         retlw   0001b             ; 0
0072 080F         00244         retlw   1111b             ; 1
0073 0809         00245         retlw   1001b             ; 2
0074 080D         00246         retlw   1101b             ; 3
0075 0807         00247         retlw   0111b             ; 4
0076 0805         00248         retlw   0101b             ; 5
0077 080F         00249         retlw   1111b             ; 6
0078 080F         00250         retlw   1111b             ; 7
0079 0801         00251         retlw   0001b             ; 8
007A 0805         00252         retlw   0101b             ; 9
007B 0803         00253         retlw   0011b             ; a
007C 0801         00254         retlw   0001b             ; b
007D 0809         00255         retlw   1001b             ; c
007E 0809         00256         retlw   1001b             ; d
007F 0801         00257         retlw   0001b             ; e
0080 0803         00258         retlw   0011b             ; f
                  00259
                  00260 ; State 2
                  00261
0081              00262 State2
                  00263         UpdateState     State2, S2_Table
                      M
```

```
0081 038A               M         swapf   sTimerLow, w
0082 0E0F               M         andlw   0xf               ; Isolate digit 5 (offset)
0083 099C               M         call    S2_Table
0084 002C               M         movwf   digit56
0085 03AC               M         swapf   digit56, f
                        M
0086 020A               M         movf    sTimerLow, w
0087 0E0F               M         andlw   0xf               ; Isolate digit 6 (offset)
0088 099C               M         call    S2_Table
0089 012C               M         iorwf   digit56, f
                        M
008A 038B               M         swapf   sTimerHigh, w
008B 0E0F               M         andlw   0xf               ; Isolate digit 5 (offset)
008C 099C               M         call    S2_Table
008D 002D               M         movwf   digit34
008E 03AD               M         swapf   digit34, f
                        M
008F 020B               M         movf    sTimerHigh, w
0090 0E0F               M         andlw   0xf               ; Isolate digit 6 (offset)
0091 099C               M         call    S2_Table
0092 012D               M         iorwf   digit34, f
                        M
0093 020D               M         movf    digit34, w        ; Display digits 3 & 4
0094 0027               M         movwf   portc
0095 020C               M         movf    digit56, w
0096 0026               M         movwf   portb             ; Display digits 5 & 6
                     00264
0097 0C04            00265         movlw   00000100b
0098 0025            00266         movwf   porta
0099 0C02            00267         movlw   00000010b
009A 0005            00268         tris    porta
                     00269
009B 0800            00270         retlw   0
                     00271
009C                 00272 S2_Table
009C 01E2            00273         addwf   pc, f
                     00274
009D 080B            00275         retlw   1011b             ; 0
009E 0803            00276         retlw   0011b             ; 1
009F 080D            00277         retlw   1101b             ; 2
00A0 080F            00278         retlw   1111b             ; 3
00A1 0807            00279         retlw   0111b             ; 4
00A2 080E            00280         retlw   1110b             ; 5
00A3 080E            00281         retlw   1110b             ; 6
00A4 080B            00282         retlw   1011b             ; 7
00A5 080F            00283         retlw   1111b             ; 8
00A6 080F            00284         retlw   1111b             ; 9
00A7 080F            00285         retlw   1111b             ; a
00A8 0806            00286         retlw   0110b             ; b
00A9 0804            00287         retlw   0100b             ; c
00AA 0807            00288         retlw   0111b             ; d
00AB 080C            00289         retlw   1100b             ; e
00AC 080C            00290         retlw   1100b             ; f
                     00291
                     00292 ; State 3
                     00293
00AD                 00294 State3
                     00295         UpdateState     State3, S3_Table
                         M
00AD 038A               M         swapf   sTimerLow, w
00AE 0E0F               M         andlw   0xf                     ; Isolate digit 5 (offset)
00AF 09C8               M         call    S3_Table
00B0 002C               M         movwf   digit56
00B1 03AC               M         swapf   digit56, f
                        M
00B2 020A               M         movf    sTimerLow, w
```

```
00B3 0E0F               M         andlw   0xf                   ; Isolate digit 6 (offset)
00B4 09C8               M         call    S3_Table
00B5 012C               M         iorwf   digit56, f
                        M
00B6 038B               M         swapf   sTimerHigh, w
00B7 0E0F               M         andlw   0xf                   ; Isolate digit 5 (offset)
00B8 09C8               M         call    S3_Table
00B9 002D               M         movwf   digit34
00BA 03AD               M         swapf   digit34, f
                        M
00BB 020B               M         movf    sTimerHigh, w
00BC 0E0F               M         andlw   0xf                   ; Isolate digit 6 (offset)
00BD 09C8               M         call    S3_Table
00BE 012D               M         iorwf   digit34, f
                        M
00BF 020D               M         movf    digit34, w            ; Display digits 3 & 4
00C0 0027               M         movwf   portc
00C1 020C               M         movf    digit56, w
00C2 0026               M         movwf   portb                 ; Display digits 5 & 6
                     00296
00C3 0C01            00297         movlw   00000001b
00C4 0025            00298         movwf   porta
00C5 0C08            00299         movlw   00001000b
00C6 0005            00300         tris    porta
                     00301
00C7 0800            00302         retlw   0
                     00303
00C8                 00304 S3_Table
00C8 01E2            00305         addwf   pc, f
                     00306
00C9 080E            00307         retlw   1110b                 ; 0
00CA 0800            00308         retlw   0000b                 ; 1
00CB 0806            00309         retlw   0110b                 ; 2
00CC 0802            00310         retlw   0010b                 ; 3
00CD 0808            00311         retlw   1000b                 ; 4
00CE 080A            00312         retlw   1010b                 ; 5
00CF 080E            00313         retlw   1110b                 ; 6
00D0 0800            00314         retlw   0000b                 ; 7
00D1 080E            00315         retlw   1110b                 ; 8
00D2 080A            00316         retlw   1010b                 ; 9
00D3 080C            00317         retlw   1100b                 ; a
00D4 080E            00318         retlw   1110b                 ; b
00D5 0806            00319         retlw   0110b                 ; c
00D6 0806            00320         retlw   0110b                 ; d
00D7 080E            00321         retlw   1110b                 ; e
00D8 080C            00322         retlw   1100b                 ; f
                     00323
                     00324 ; Main code
                     00325
00D9                 00326 Start
00D9 0900            00327         call    Initialize
00DA                 00328 Repeat
00DA 0917            00329         call    Timer_Check
00DB 0ADA            00330         goto    Repeat
                     00331
01FF                 00332         org     pic55
                     00333
01FF                 00334 System_Reset
01FF 0AD9            00335         goto    Start
                     00336
                     00337         END

MEMORY USAGE MAP ('X' = Used,  '-' = Unused)

0000 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
0040 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
```

```
0080 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
00C0 : XXXXXXXXXXXXXXXX XXXXXXXXXXXX---- ---------------- ----------------
01C0 : ---------------- ---------------- ---------------- ---------------X

All other memory blocks unused.

Program Memory Words Used:   221
Program Memory Words Free:   291


Errors   :     0
Warnings :     0 reported,     0 suppressed
Messages :     0 reported,     0 suppressed
```

Note the following details of the code protection feature on PICmicro® MCUs.

- The PICmicro family meets the specifications contained in the Microchip Data Sheet.
- Microchip believes that its family of PICmicro microcontrollers is one of the most secure products of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the PICmicro microcontroller in a manner outside the operating specifications contained in the data sheet. The person doing so may be engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable”.
- Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our product.

If you have any further questions about this matter, please contact the local sales office nearest to you.

DNV Certification, Inc. USA — ANSI-RAB QMS ACCREDITED — DNV MSC The Netherlands Accredited by the RvA — DNV — ISO 9001 / QS-9000 REGISTERED FIRM

Microchip received QS-9000 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona in July 1999. The Company’s quality system processes and procedures are QS-9000 compliant for its PICmicro® 8-bit MCUs, KEELOQ® code hopping devices, Serial EEPROMs and microperipheral products. In addition, Microchip’s quality system for the design and manufacture of development systems is ISO 9001 certified.

WORLDWIDE SALES AND SERVICE

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200 Fax: 480-792-7277
Technical Support: 480-792-7627
Web Address: http://www.microchip.com

Rocky Mountain
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7966 Fax: 480-792-7456

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

Boston
2 Lan Drive, Suite 120
Westford, MA 01886
Tel: 978-692-3848 Fax: 978-692-3821

Chicago
333 Pierce Road, Suite 180
Itasca, IL 60143
Tel: 630-285-0071 Fax: 630-285-0075

Dallas
4570 Westgrove Drive, Suite 160
Addison, TX 75001
Tel: 972-818-7423 Fax: 972-818-2924

Detroit
Tri-Atria Office Building
32255 Northwestern Highway, Suite 190
Farmington Hills, MI 48334
Tel: 248-538-2250 Fax: 248-538-2260

Kokomo
2767 S. Albright Road
Kokomo, Indiana 46902
Tel: 765-864-8360 Fax: 765-864-8387

Los Angeles
18201 Von Karman, Suite 1090
Irvine, CA 92612
Tel: 949-263-1888 Fax: 949-263-1338

New York
150 Motor Parkway, Suite 202
Hauppauge, NY 11788
Tel: 631-273-5305 Fax: 631-273-5335

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

Toronto
6285 Northam Drive, Suite 108
Mississauga, Ontario L4V 1X5, Canada
Tel: 905-673-0699 Fax: 905-673-6509

ASIA/PACIFIC

Australia
Microchip Technology Australia Pty Ltd
Suite 22, 41 Rawson Street
Epping 2121, NSW
Australia
Tel: 61-2-9868-6733 Fax: 61-2-9868-6755

China - Beijing
Microchip Technology Consulting (Shanghai) Co., Ltd., Beijing Liaison Office
Unit 915
Bei Hai Wan Tai Bldg.
No. 6 Chaoyangmen Beidajie
Beijing, 100027, No. China
Tel: 86-10-85282100 Fax: 86-10-85282104

China - Chengdu
Microchip Technology Consulting (Shanghai) Co., Ltd., Chengdu Liaison Office
Rm. 2401, 24th Floor,
Ming Xing Financial Tower
No. 88 TIDU Street
Chengdu 610016, China
Tel: 86-28-6766200 Fax: 86-28-6766599

China - Fuzhou
Microchip Technology Consulting (Shanghai) Co., Ltd., Fuzhou Liaison Office
Unit 28F, World Trade Plaza
No. 71 Wusi Road
Fuzhou 350001, China
Tel: 86-591-7503506 Fax: 86-591-7503521

China - Shanghai
Microchip Technology Consulting (Shanghai) Co., Ltd.
Room 701, Bldg. B
Far East International Plaza
No. 317 Xian Xia Road
Shanghai, 200051
Tel: 86-21-6275-5700 Fax: 86-21-6275-5060

China - Shenzhen
Microchip Technology Consulting (Shanghai) Co., Ltd., Shenzhen Liaison Office
Rm. 1315, 13/F, Shenzhen Kerry Centre,
Renminnan Lu
Shenzhen 518001, China
Tel: 86-755-2350361 Fax: 86-755-2366086

Hong Kong
Microchip Technology Hongkong Ltd.
Unit 901-6, Tower 2, Metroplaza
223 Hing Fong Road
Kwai Fong, N.T., Hong Kong
Tel: 852-2401-1200 Fax: 852-2401-3431

India
Microchip Technology Inc.
India Liaison Office
Divyasree Chambers
1 Floor, Wing A (A3/A4)
No. 11, O'Shaugnessey Road
Bangalore, 560 025, India
Tel: 91-80-2290061 Fax: 91-80-2290062

Japan
Microchip Technology Japan K.K.
Benex S-1 6F
3-18-20, Shinyokohama
Kohoku-Ku, Yokohama-shi
Kanagawa, 222-0033, Japan
Tel: 81-45-471- 6166 Fax: 81-45-471-6122

Korea
Microchip Technology Korea
168-1, Youngbo Bldg. 3 Floor
Samsung-Dong, Kangnam-Ku
Seoul, Korea 135-882
Tel: 82-2-554-7200 Fax: 82-2-558-5934

Singapore
Microchip Technology Singapore Pte Ltd.
200 Middle Road
#07-02 Prime Centre
Singapore, 188980
Tel: 65-334-8870 Fax: 65-334-8850

Taiwan
Microchip Technology Taiwan
11F-3, No. 207
Tung Hua North Road
Taipei, 105, Taiwan
Tel: 886-2-2717-7175 Fax: 886-2-2545-0139

EUROPE

Denmark
Microchip Technology Nordic ApS
Regus Business Centre
Lautrup hoj 1-3
Ballerup DK-2750 Denmark
Tel: 45 4420 9895 Fax: 45 4420 9910

France
Microchip Technology SARL
Parc d'Activite du Moulin de Massy
43 Rue du Saule Trapu
Batiment A - ler Etage
91300 Massy, France
Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79

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

Italy
Microchip Technology SRL
Centro Direzionale Colleoni
Palazzo Taurus 1 V. Le Colleoni 1
20041 Agrate Brianza
Milan, Italy
Tel: 39-039-65791-1 Fax: 39-039-6899883

United Kingdom
Arizona Microchip Technology Ltd.
505 Eskdale Road
Winnersh Triangle
Wokingham
Berkshire, England RG41 5TU
Tel: 44 118 921 5869 Fax: 44-118 921-5820

01/18/02