AN564

Using the PWM

Author: *Mark Palmer*
Microchip Technology Inc.

INTRODUCTION

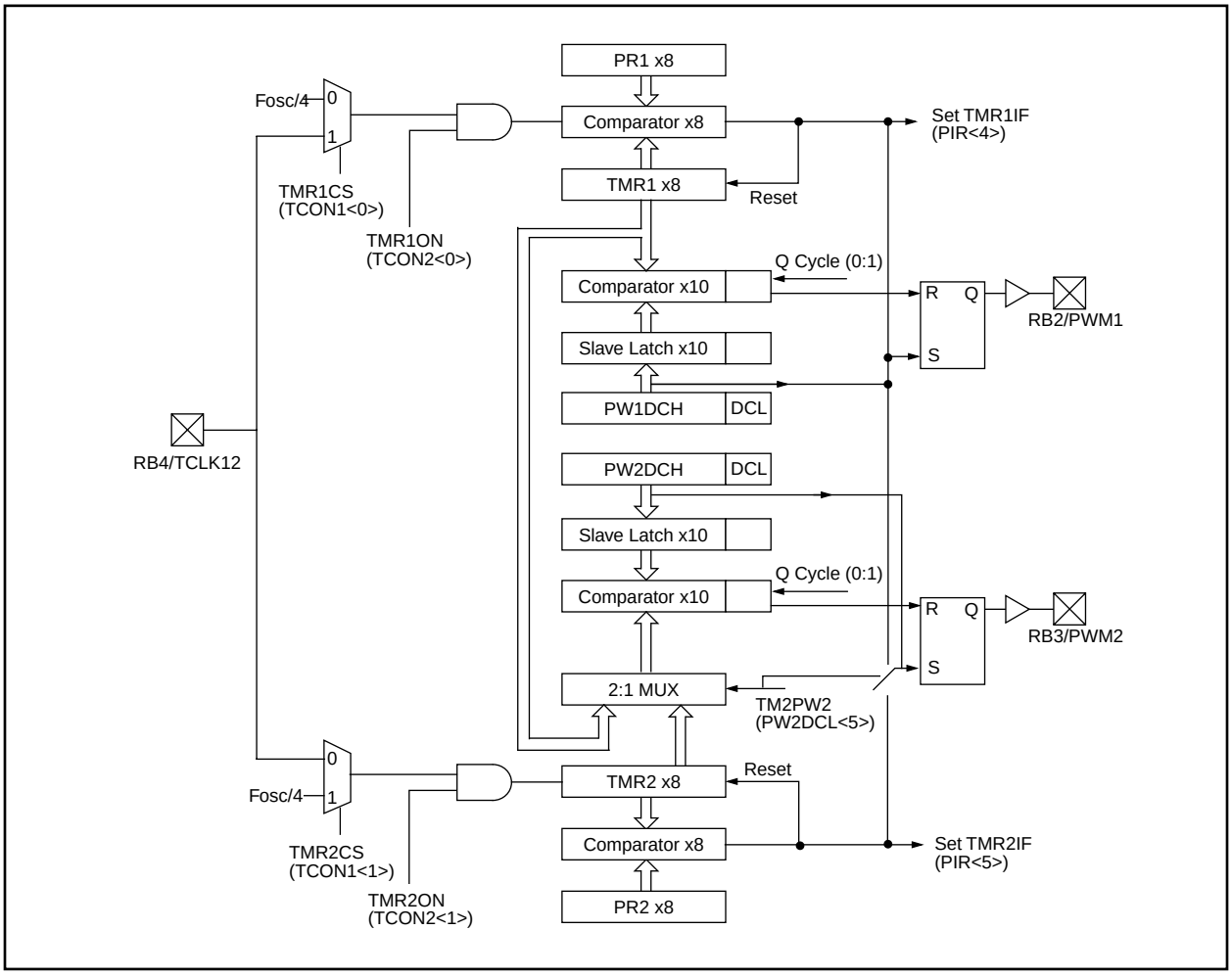
The PICmico™ family of RISC microcontrollers hax been designed to provide advanced performance and a cost-effective solution for a variety of applications. This application note provides examples which illustrate some uses of Pulse Width Modulation (PWM) using the PIC17C42's Timer1 or Timer2 module. These examples may be modified to suit the specific needs of your application.

This Application Note describes the operation of the PWM. They include the following topics:

1. Simple PWM Operation
2. Variable Period / Variable Duty Cycle PWM
3. External Clock for Timer Time-base (ramifications/issues)

The listing file for the Variable Period / Variable Duty Cycle example can be found in Appendix A. The source files can be found on the Microchip BBS. On directions on how to access the Microchip BBS please refer to DS30128, which can also be found in the Microchip Embedded Control Handbook (Literature Number DS00092).

FIGURE 1: TIMER1 AND TIMER2 BLOCK DIAGRAM WITH PWM



Control registers that are used by Timer1 and Timer2 are shown in Table 1. Shaded Boxes are control bits that are not used by Timer1 or Timer2.

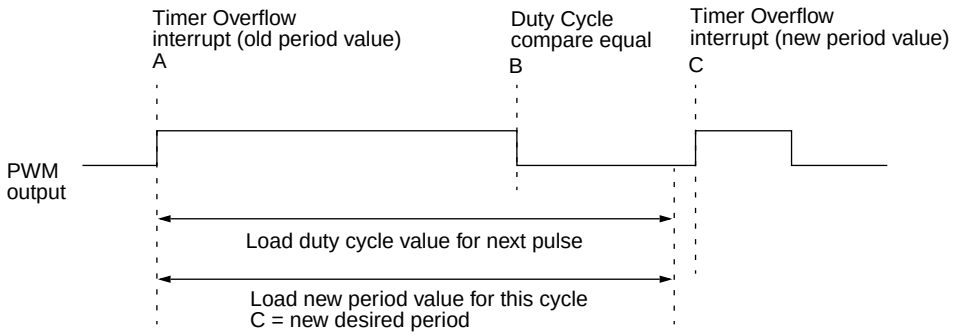
TABLE 1: REGISTERS ASSOCIATED WITH TIMER3 AND CAPTURE

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-On Reset	Value on all other resets (Note1)
16h, Bank 3	TCON1	CA2ED1	CA2ED0	CA1ED1	CA1ED0	T16	TMR3CS	TMR2CS	TMR1CS	0000 0000	0000 0000
17h, Bank 3	TCON2	CA2OVF	CA1OVF	PWM2ON	PWM1ON	CA1/$\overline{PR3}$	TMR3ON	TMR2ON	TMR1ON	0000 0000	0000 0000
10h, Bank 2	TMR1	Timer1								xxxx xxxx	uuuu uuuu
11h, Bank 2	TMR2	Timer2								xxxx xxxx	uuuu uuuu
12h, Bank 2	TMR3L	Timer3 low byte								xxxx xxxx	uuuu uuuu
13h, Bank 2	TMR3H	Timer3 high byte								xxxx xxxx	uuuu uuuu
16h, Bank 1	PIR	RBIF	TMR3IF	TMR2IF	TMR1IF	CA2IF	CA1IF	TXIF	RCIF	0000 0010	0000 0010
17h, Bank 1	PIE	RBIE	TMR3IE	TMR2IE	TMR1IE	CA2IE	CA1IE	TXIE	RCIE	0000 0000	0000 0000
07h, Unbanked	INTSTA	PEIF	T0CKIF	T0IF	INTF	PEIE	T0CKIE	T0IE	INTE	0000 0000	0000 0000
06h, Unbanked	CPUSTA	—	—	STKAV	GLINTD	$\overline{TO}$	$\overline{PD}$	—	—	--11 11--	--11 ??--
14h, Bank 2	PR1	Timer1 period register								xxxx xxxx	uuuu uuuu
15h, Bank 2	PR2	Timer2 period register								xxxx xxxx	uuuu uuuu
16h, Bank 2	PR3L/CA1L	Timer3 period register, low byte/capture1 register, low byte								xxxx xxxx	uuuu uuuu
17h, Bank 2	PR3H/CA1H	Timer3 period register, high byte/capture1 register, high byte								xxxx xxxx	uuuu uuuu
10h, Bank 3	PW1DCL	DC1	DC0	—	—	—	—	—	—	xx-- ----	uu-- ----
11h, Bank 3	PW2DCL	DC1	DC0	TM2PW2	—	—	—	—	—	xx0- ----	uu0- ----
12h, Bank 3	PW1DCH	DC9	DC8	DC7	DC6	DC5	DC4	DC3	DC2	xxxx xxxx	uuuu uuuu
13h, Bank 3	PW2DCH	DC9	DC8	DC7	DC6	DC5	DC4	DC3	DC2	xxxx xxxx	uuuu uuuu
14h, Bank 3	CA2L	Capture2 low byte								xxxx xxxx	uuuu uuuu
15h, Bank 3	CA2H	Capture2 high byte								xxxx xxxx	uuuu uuuu

Legend: x = unknown, u = unchanged, - = unimplemented, reads as '0'. ? - Value depends on condition.
Note 1: Shaded cells are not used by TMR1, TMR2,TMR3 or Capture.

Care must be taken when loading values into the PWM registers. These registers are the duty cycle registers (PWxDCH:PWxDCL) and the period register (PRx). Figure 2 shows proper update timing of these values.

FIGURE 2: TIMING FOR UPDATING THE DUTY CYCLE REGISTERS AND PERIOD REGISTER



Note: When updating the period register, the value loaded must be greater than the timer value. If the period value is less than the timer value, the duty cycle value is not latched and the timer is not reset to 00h until the next Timer = Period match. This causes the current cycle to not output as expected and cause a “glitch”.

Note: It is generally good practice to load the new period value into the PRx register as soon as possible after Point A.

SIMPLE PWM OPERATION

Simple PWM operation is where the period of the PWM output remains constant, and only the duty cycle is modified. The PWM can operate in either of two modes:

- Hi-resolution mode: the PWxDCL register is modified
- Standard resolution mode: the PWxDCL register is not modified

When operating in the standard-resolution mode, only the PWDCH register is ever modified. Since the modification takes only a single cycle and can be done at any time. Also since the period is remaining constant this may be done without any PWM interrupt software overhead.

When operating in the high-resolution mode both the PWxDCH:PWxDCL register pair is modified. Since this is a multicycle update, care needs to be taken that the "new" PWM duty cycle value is not latched until the update is complete. If the duty cycle is latched before this update is complete, the duty cycle will display a "glitch". If the PWxDCH is written first, the maximum error is 3 Q-cycles (187.5 ns @ 16 MHz). If the PWxDCL is written first, the maximum error is also 3 Q-cycles (187.5 ns @ 16 MHz), with the PWxDCH delayed by one PWM period. This may be acceptable for some applications. If this is not acceptable for your application then a subroutine can be written to ensure that these duty cycle writes are not done when the timer will equal the period. One implementation of this subroutine (`PWM_UD`) is used in the Variable Period / Variable Duty Cycle PWM example. This is discussed in the following section, with the listing in Appendix A.

Additional code examples can be found in application note AN539 in the Embedded Control Handbook.

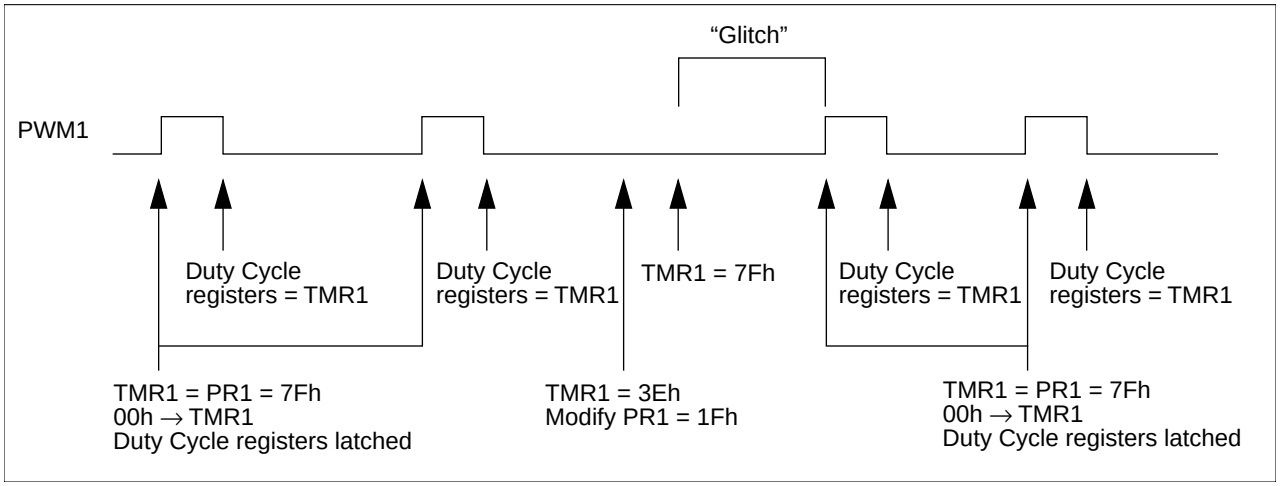
VARIABLE PERIOD / VARIABLE DUTY CYCLE PWM

In a variable period / variable duty cycle PWM both the duty cycle of the PWM as well as the frequency (period) of the PWM are modified.

The PIC17C42's hardware double buffers the duty cycle registers, but the period registers are not double buffered. What this means is that you can modify the duty cycle registers, but the value will only be latched when the timer register equals the period register. Since the period register is not buffered, as the period register is modified this becomes the "new" period. This means that care must be taken when modifying the period register. The most common problem would be to modify the period register resulting in a "glitch." This "glitch" occurs when the period register is modified with a value that is less than the present timer value. The timer does not have a match with the old period value, and continues to count until the timer register equals the period register.

Figure 3, shows an example where PR1 the register period = 7Fh. Then the period is modified to a smaller value (PR1 = 1Fh) without checking that the value in Timer1 (TMR1) register = 3Eh. Since the new period (PR1) value is less then the present timer (TMR1) value, a glitch has occurred.

FIGURE 3: MODIFYING PERIOD REGISTER "GLITCH"



Care must be taken when writing a 10-bit duty cycle value. Since this requires two register writes, the “Timer Equals Period” could occur between these two writes, which would give a duty cycle that was not as expected. The cases are as follows:

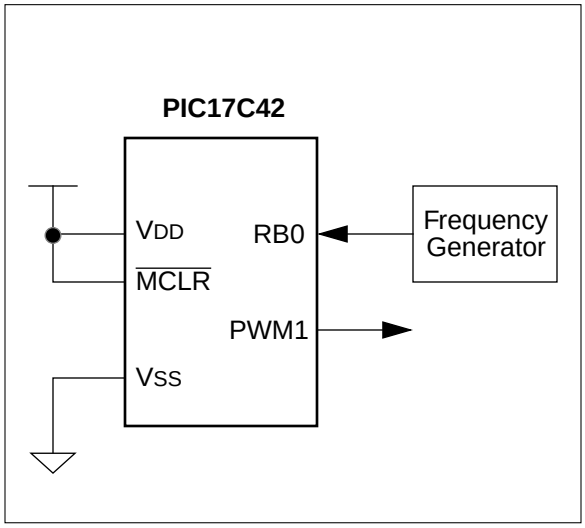
a) If the duty cycle low register (DCL) is written, and then the Timer equals period. The old DCH register and the new DCL register becomes the duty cycle.

b) If the duty cycle high register (DCH) is written, and then the Timer equals period. The new DCH register and the old DCL register becomes the duty cycle

At the following occurrence of the timer equaling the period, the second register written would be updated. The subroutine `PWM_UD` (Appendix A) ensures that these duty cycle writes are not done when the timer will equal the period.

A software example of a variable period / variable duty cycle is shown in Appendix A. In this example the period is double buffered in software, and the new period value is loaded in the timer overflow interrupt service routine. When the new duty cycle needs to be loaded. The device connections are shown in Figure 4. This program has two PWM settings (period / duty cycle combinations) that are switched between depending on the level on pin RB0. A frequency generator was used to give a low frequency signal on the RB0 pin. Figure 5 shows an example of the input and output waveforms.

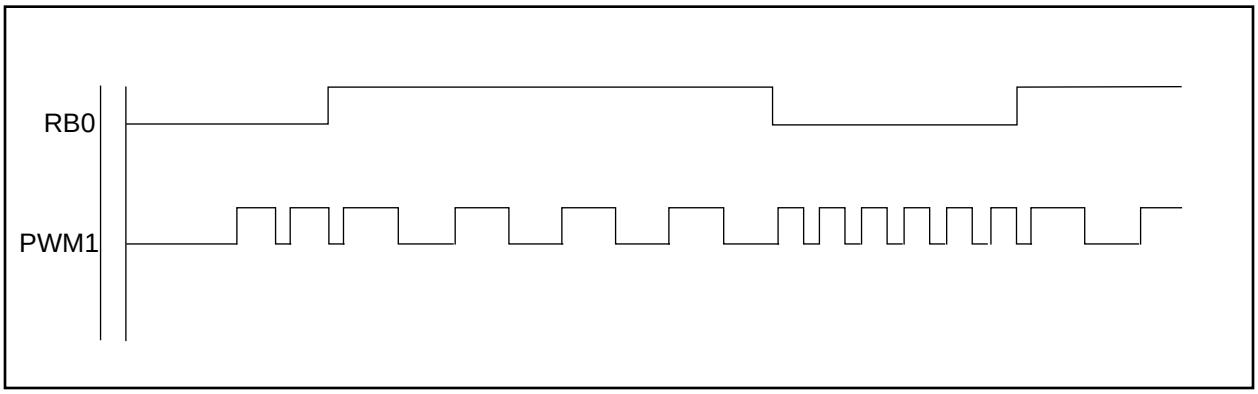
FIGURE 4: APPLICATION HARDWARE SETUP



The program listing in Appendix A implements this example, Figure 8 is the hardware function. This example may be modified to suit the particular needs of your application. The following table is a summary of the requirements for this program (@ 16 MHz):

Code Size:	52 Words
RAM used:	11 Bytes
Interrupt Service Routine time	3.0 µs
Subroutine time	4.5 µs 6.0 µs
Maximum PWM frequency:	200 kHz
PWM Accuracy:	62.5 µs

FIGURE 5: EXAMPLE APPLICATION WAVEFORMS



EXTERNAL CLOCK FOR TIMER TIMEBASE

The counters used for the time-base of the PWM outputs can be software selected to operate from an external clock source. This allows a lower frequency PWM to be achieved. This brings up new issues that must be understood for the application.

One of these issues is clock synchronization. All external clocks must be synchronized to the internal operating speed of the microcontroller, as shown in Figure 6. When this synchronization occurs the PWM output is not truly operating from the external clock, but actually the internal synchronized clock. This leads to a "jitter" of the output to the clock. This jitter is caused from the delta time between the external clock and the synchronized clock not being constant. The synchronization errors are:

Duty cycle error = $\pm T_{CY}$

Period error = $\pm T_{CY}$

If you needed to run the PWM at a low frequency, and also want to reduce the "jitter" from the use of an external asynchronous clock, a PWM output could be used as the synchronous clock source. When the clock is synchronized to the device the clock error is always constant, so there is no jitter. Figure 7 shows this example.

FIGURE 7: PWM OUTPUT TO GENERATE A SYNCHRONOUS CLOCK

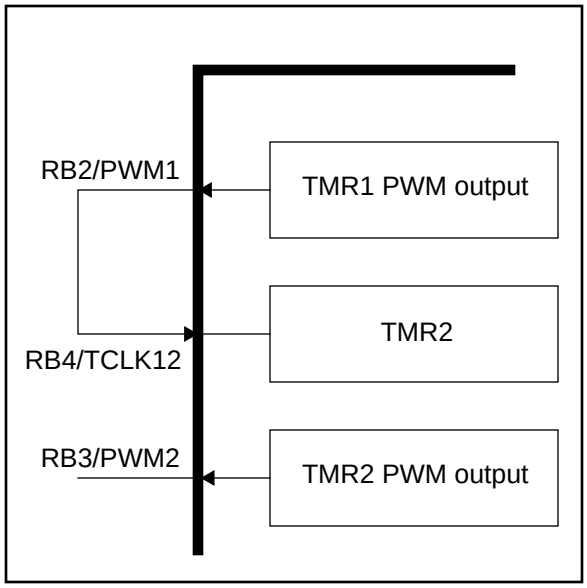
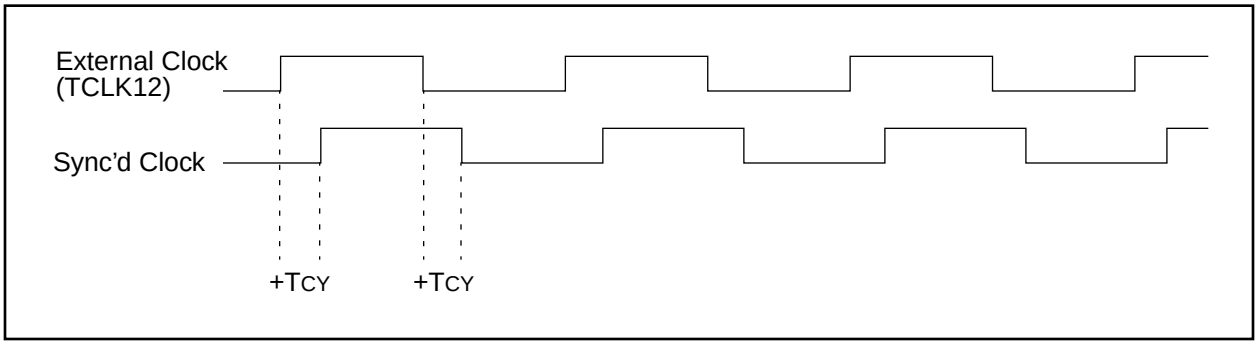


FIGURE 6: EXTERNAL CLOCK SYNCHRONIZATION



Another use is where precise timing of updates need to be done, but not at the frequency of the PWM output. In this discussion, TMR1 is used as the time-base of a constant frequency PWM output. TMR1 uses the internal clock of the device and TMR2 uses the external clock input. TMR2 will get the clock input from the PWM2 output.

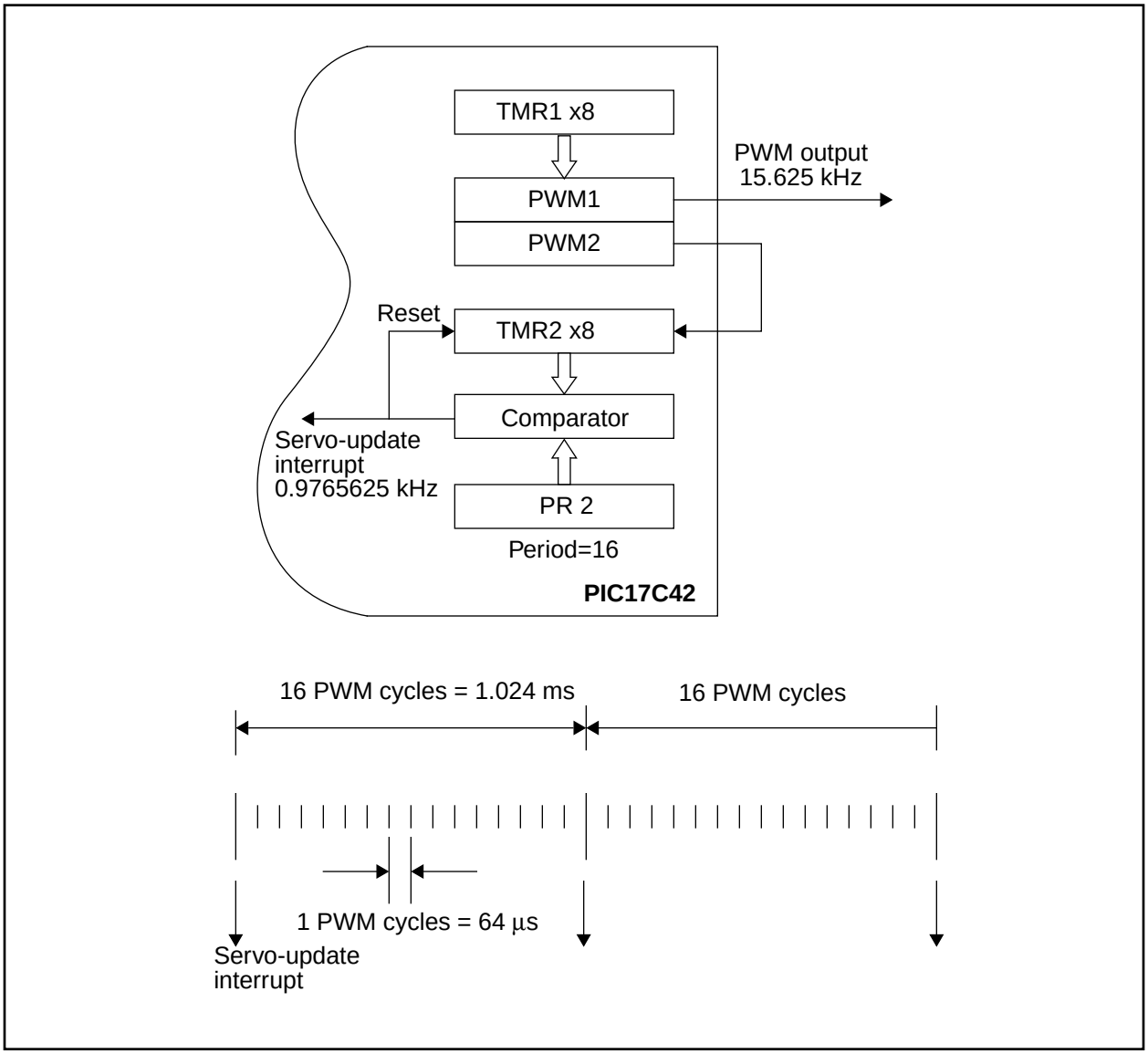
The PWM output is a constant frequency variable duty cycle output. The PW1DCH:PW1DCL register pair contain the variable duty cycle value of PWM1 output. The PW2DCH:PW2DCL register pair is set for a fixed duty cycle (50%) for the PWM2 output.

The PWM outputs could be programmed to have a frequency of 20 kHz, so to reduce audible noise. The PWM2 signal is connected to the RB4/TCLK12, as shown in Figure 8. The PR2 register could be loaded with 14h (20), to give an interrupt every 1 kHz. This interrupt can then trigger tasks, such as updating the duty cycle of PWM1. This is useful in motor control as well as other applications where the update rate is less then the PWM frequency.

CONCLUSION

The PIC17C42's PWM features offer a high performance solution at a lower system cost than previously available. The versatility of PWMs make the PIC17C42 ideal for motor control applications (ses AN532) and many industrial control applications.

FIGURE 8: SAMPLING SCHEME



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: LISTING FILE

```
MPASM 01.40 Released         AN564_P1.ASM   1-16-1997  16:47:52         PAGE  1


LOC  OBJECT CODE     LINE SOURCE TEXT
  VALUE

             00001             PROCESSOR               17C42
             00002 ;
             00003 ;*********************************************************
             00004 ;
             00005 ;       Program:          AN564-P1.ASM
             00006 ;       Revision Date:
             00007 ;                         1-15-97  Compatibility with MPASMWIN 1.40
             00008 ;
             00009 ;*********************************************************
             00010 ;
             00011
             00012 ; This is the basic outline for a program that generates a
             00013 ; variable PWM output. The PWM's period and duty cycle can
             00014 ; be varied. The new period (NEW_PR1) and the new duty cycle
             00015 ; (NEW_DC1 and NEW_DC1Q) are loaded by the user program.
             00016 ; The peripheral interrupt routine loads the new period value
             00017 ; (frequency) into the PR1 register. A subroutine (PWM_UD)
             00018 ; is also used to ensure that the 10-bit duty cycle registers
             00019 ; are updated in the same PWM cycle, i.e. the timer match does not
             00020 ; occur between two duty cycle register writes.
             00021 ;
             00022 ; The duty cycle value gets latched on the overflow (Period match)
             00023 ; of the timer. The period value gets modified as soon as the period
             00024 ; register is changed. Therefore care must be taken in updating
             00025 ; the period register. In cases where the period value is modified
             00026 ; to a smaller value, we must ensure that the Timer counter is less
             00027 ; then this value when the period register is updated (TMR1 < new PR1).
             00028 ; If TMR1 is greater then PR1, the counter will count to FFh, rollover
             00029 ; to 00H, and only cause the overflow interrupt when it then reaches
             00030 ; the period value. This would give a wrong PWM output.
             00031 ;
             00032 ; In this example the event which cause the PWM to be updated
             00033 ; is an asynchronous event. A low frequency signal was placed on
             00034 ; port pin RB0.
             00035 ; For a high level the PWM registers are updated as follows:
             00036 ;    PR1 = 7Fh, PW1DCH = 3Fh, and PW1DCL = 40h
             00037 ; For a low level the PWM registers are updated as follows:
             00038 ;    PR1 = 1Fh, PW1DCH = 07h, and PW1DCL = 80h
             00039 ;
             00040 ; Do the EQUate table
             00041 ;
  00000020   00042 NEW_DC1           EQU  0x20 ; New PWM1 duty cycle value
  00000021   00043 NEW_DC1Q          EQU  0x21 ;
  00000022   00044 NEW_PR1           EQU  0x22 ; New PWM1 period value
  00000025   00045 PWM_WIN           EQU  0x25 ; Register for the PWM window cycle count
  00000026   00046 CALC_PR           EQU  0x26 ; Calculated period value
  00000027   00047 FLAG_REG          EQU  0x27 ; Register for flag bits
             00048 ;
  0000001A   00049 DC1H              EQU  0x1A ; PWM registers for high time
  0000001B   00050 DC1QH             EQU  0x1B
  0000001C   00051 PR1H              EQU  0x1C
             00052 ;
  0000001D   00053 DC1L              EQU  0x1D  ; PWM registers for low time
```

```
  0000001E    00054 DC1QL             EQU  0x1E
  0000001F    00055 PR1L              EQU  0x1F
              00056 ;
              00057 ;
  000007FF    00058 END_OF_PROG_MEM   EQU  0x07FF
              00059 ;
  00000004    00060 ALUSTA            EQU  0x04
  00000006    00061 CPUSTA            EQU  0x06
  00000007    00062 INTSTA            EQU  0x07
  0000000A    00063 W                 EQU  0x0A
              00064 ;
  00000011    00065 DDRB              EQU  0x11  ; Bank 0
  00000012    00066 PORTB             EQU  0x12
              00067 ;
  00000016    00068 PIR               EQU  0x16  ; Bank 1
  00000017    00069 PIE               EQU  0x17
              00070 ;
  00000010    00071 TMR1              EQU  0x10  ; Bank 2
  00000011    00072 TMR2              EQU  0x11
  00000012    00073 TMR3l             EQU  0x12
  00000013    00074 TMR3h             EQU  0x13
  00000014    00075 PR1               EQU  0x14
  00000015    00076 PR2               EQU  0x15
  00000016    00077 PR3L              EQU  0x16
  00000017    00078 PR3h              EQU  0x17
              00079 ;
  00000010    00080 PW1DCL            EQU  0x10  ; Bank 3
  00000011    00081 PW2DCL            EQU  0x11
  00000012    00082 PW1DCH            EQU  0x12
  00000013    00083 PW2DCH            EQU  0x13
  00000016    00084 TCON1             EQU  0x16
  00000017    00085 TCON2             EQU  0x17
              00086  PAGE
0000          00087                   ORG  0x0000   ; Origin for the RESET vector
0000 C02B     00088                   GOTO START    ; On reset, go to the start of
              00089                                 ;   the program
0008          00090                   ORG  0x0008   ; Origin for the external RA0/INT
              00091                                 ;   interrupt vector
0008 C07C     00092                   GOTO EXT_INT  ; Goto the ext. interrupt
              00093                                 ;   on RA0/INT routine
0010          00094                   ORG  0x0010   ; Origin for the TMR0
              00095                                 ;   overflow interrupt vector
0010 C07D     00096                   GOTO TMR0INT  ; Goto the TMR0 overflow interrupt
              00097                                 ;   routine
0018          00098                   ORG  0x0018   ; Origin for the external
              00099                                 ;   RA1/T0CKI interrupt vector
0018 C07E     00100                   GOTO T0_INT   ; Goto the ext. interrupt on
              00101                                 ;   RA1/T0CKI routine
0020          00102                   ORG  0x0020   ; Origin for the interrupt vector
              00103                                 ;   of any enabled peripheral
              00104 ;
              00105 ; The interrupt routine for any peripheral interrupt, This routine
              00106 ; only deals with Timer1 interrupt.
              00107 ;
              00108 ; Time required to execute interrupt routine. Not including
              00109 ; interrupt latency (time to enter into the interrupt routine)
              00110 ;
              00111 ;       case1 - only T1 overflow = 12 cycles
              00112 ;       case2 - Other            = Infinite Loop
              00113 ;
              00114 ;
0020 B801     00115 PER_INT MOVLB   1                     ; Select register Bank 1
0021 9416     00116                 BTFSS   PIR,4         ; Did Timer1 overflow?
0022 C022     00117 ERR1    GOTO    ERR1                  ; Not a Timer1 overflow.
              00118                                       ;   No other interrupts should
              00119                                       ;   be enabled, so error.
```

```
                00120 ;
                00121 ; Once the enabled Timer1 overflow occurs, the period register
                00122 ; is loaded. This PWM waveform will remain until the PWM duty
                00123 ; cycle and / or period is updated. Until such update, there is no
                00124 ; S/W overhead from T1 interrupts (T1 interrupts can be disabled).
                00125 ;
                00126 ; NOTE: If PW1DCH >= PR1, then the duty cycle of this PWM output
                00127 ;       is 100%.
                00128 ;
                00129 ; NOTE: The new Period register (PR1) value, must always be greater
                00130 ;       than the value in the Timer1 register (TMR1). If a PR1 value
                00131 ;       is loaded that is less then the TMR1 value, the timer will
                00132 ;       continue to count until it reaches the PR1 value. I.E. TMR1
                00133 ;       will overflow at FFh and the count to the new PR1 value.
                00134 ;       Minimum PR1 value is 0Ah, due to time to load new values and
                00135 ;       execute the peripheral interrupt service routine.
                00136 ;
0023 8C16       00137 T1OVFL      BCF      PIR,4       ; Clear Overflow interrupt flag
0024 B802       00138             MOVLB   2            ; Bank2
0025 7422       00139             MOVFP   NEW_PR1,PR1  ; Load this period value
0026 B801       00140             MOVLB   1            ; Bank 0
0027 8C17       00141             BCF     PIE, 4       ; Disable T1 interrupt
                00142                                  ;   (until transition on PORTB0)
0028 B800       00143             MOVLB   0            ; Bank 0
0029 3F12       00144             BTG     PORTB, 7  ; Transition PortB 7 pin (H->L, or L->H)
002A 0005       00145             RETFIE               ; Return from Interrupt
                00146  PAGE
                00147 ;
                00148 ; This is the start of the program.
                00149 ;
002B 8406       00150 START       BSF     CPUSTA,4     ; Disable ALL interrupts via the
                00151                                  ;   Global Interrupt Disable
                00152                                  ;   (GLINTD) bit.
                00153                                  ;
002C            00154 MAIN                             ; Place Main program here
002C B803       00155             MOVLB   3            ; Select register Bank 3
002D 2817       00156             CLRF    TCON2,0      ; Stop the timers, Single Capture
002E B070       00157             MOVLW   0x070        ; Initalize TCON1 so that
002F 0116       00158             MOVWF   TCON1        ;   T1 (8-bit), T2 (8-bit),
                00159                                  ;   and T3 run off the internal
                00160                                  ;   system clock. Timer3 uses
                00161                                  ;   period register
0030 B00D       00162             MOVLW   0x0D         ; Load the PWM window cycle value
0031 0125       00163             MOVWF   PWM_WIN      ;
                00164 ;
0032 B800       00165             MOVLB   0            ; Select register Bank 0
0033 2B11       00166             SETF    DDRB, 1      ; Port B is an input
0034 2912       00167             CLRF    PORTB, 1     ; Set output values to 0 (for PORTB)
0035 8F11       00168             BCF     DDRB, 7 ; PORTB7 is an output used to trigger a scope
0036 2927       00169             CLRF    FLAG_REG, 1  ; Clear the Flag registers
                00170 ;
                00171 ; Load registers with the PWM values that we will switch between. One
                00172 ; set for the time PORTB0 is high and another set for when low.
                00173 ;
                00174 ; For a high level the PWM registers are updated as follows:
                00175 ;    PR1 = 7Fh, PW1DCH = 3Fh, and PW1DCL = 40h
                00176 ; 16Mhz gives a period of 31.75 us and a duty cycle of 16.625 us
                00177 ; For a low level the PWM registers are updated as follows:
                00178 ;   PR1 = 1Fh, PW1DCH = 07h, and PW1DCL = 80h
                00179 ; At 16Mhz this gives a period of 7.75 us, and a duty cycle of 6.00 us
                00180 ;
0037 B803       00181             MOVLB   3            ; Bank 3
0038 B03F       00182             MOVLW   0x3F         ; The Duty Cycle initial value is
0039 4A1A       00183             MOVPF   W, DC1H      ;   50% of the initial period
003A B040       00184             MOVLW   0x40         ;
003B 4A1B       00185             MOVPF   W, DC1QH     ; Duty Cycle low = 01
```

```
003C B007     00186          MOVLW  0x07         ; The Duty Cycle initial value is
003D 4A1D     00187          MOVPF  W, DC1L      ;   25% of the initial period
003E B080     00188          MOVLW  0x80         ;
003F 4A1E     00189          MOVPF  W, DC1QL     ; Duty Cycle low = 10
              00190 ;
0040 B802     00191          MOVLB  2            ; Bank 2
0041 B07F     00192          MOVLW  0x7F         ;
0042 4A1C     00193          MOVPF  W, PR1H      ; The initial period value is 50%
              00194                              ;   of full scale (for High)
0043 B01F     00195          MOVLW  0x1F
0044 4A1F     00196          MOVPF  W, PR1L      ; The initial period value is 12.5%
              00197                              ;   of full scale (for Low)
              00198 ;
              00199 ;
              00200 ; Default PWM values should be set, and the timer should be started
              00201 ; and the interrupts enabled.
              00202 ;
0045 B0F0     00203          MOVLW  0xF0         ; Load the Period register
0046 0114     00204          MOVWF  PR1          ;
0047 B803     00205          MOVLB  3            ; Select register Bank 3
0048 B0C0     00206          MOVLW  0xC0         ; Load the T1 duty cycle register
0049 0112     00207          MOVWF  PW1DCH       ;
004A 0110     00208          MOVWF  PW1DCL       ; effectively loaded with 0
004B B031     00209          MOVLW  0x31         ;** Enable PWM1 and PWM2 outputs
004C 0117     00210          MOVWF  TCON2        ;**   and turn on Timer1.
              00211
004D 8307     00212          BSF    INTSTA,3     ; Turn on Peripheral Interrupts
004E B801     00213          MOVLB  1            ; Select register Bank 1
004F B010     00214          MOVLW  0x10         ; Enable Timer1 overflow
0050 0117     00215          MOVWF  PI           ;   Interrupts (when GLINTD = 0)
0051 8C06     00216          BCF    CPUSTA,4     ; Enable ALL interrupts
0052 B800     00217          MOVLB  0            ; Bank 0
              00218  PAGE
              00219 ;
              00220 ; Only need to update PWM values on the first occurance of a new level
              00221 ; on RB0, Else loop waiting for level to change.
              00222 ;
0053 8827     00223 HIGH1ST BCF     FLAG_REG, 0 ; First time in loop (this cycle)= True
0054 9012     00224 HIGHCYC BTFSS   PORTB, 0    ; Is PortB0 low
0055 C05F     00225         GOTO    LOW1ST      ; PORTB0 = L
0056 9827     00226         BTFSC   FLAG_REG, 0 ; Is this the First High time (this cycle)?
0057 C054     00227         GOTO    HIGHCYC     ; Loop looking for low signal on PortB0
0058 8027     00228         BSF     FLAG_REG, 0 ; Set First time in loop (this cycle)=False
              00229 ;
              00230 ; Here is where we update the PWM values (period and Duty cycle)
              00231 ; for high level.
0059 B803     00232         MOVLB   3               ; Bank 3
005A 5A20     00233         MOVPF   DC1H, NEW_DC1   ;
005B 5B21     00234         MOVPF   DC1QH, NEW_DC1Q ;
005C 5C22     00235         MOVPF   PR1H, NEW_PR1   ;
005D E06B     00236         CALL    PWM1_UD         ;
005E C054     00237         GOTO    HIGHCYC         ; Loop looking for low signal on PortB0
              00238 ;
              00239 ;
005F 8827     00240 LOW1ST   BCF    FLAG_REG, 0    ; First time in loop (this cycle)=True
0060 9812     00241 LOWCYC   BTFSC  PORTB, 0       ; Is PortB0 high
0061 C053     00242          GOTO   HIGH1ST        ; PORTB0 = H
0062 9827     00243          BTFSC  FLAG_REG, 0  ; Is this the First Low time (this cycle)?
0063 C060     00244          GOTO   LOWCYC         ; Loop looking for high signal on PortB0
0064 8027     00245          BSF    FLAG_REG, 0  ; First time in loop (this cycle) = False
              00246 ;
              00247 ; Here is where we update the PWM values (period & Duty cycle) for low level.
              00248 ;
0065 B803     00249          MOVLB  3
0066 5D20     00250          MOVPF  DC1L, NEW_DC1   ;
0067 5E21     00251          MOVPF  DC1QL, NEW_DC1Q ;
```

```
0068 5F22     00252          MOVPF  PR1L, NEW_PR1   ;
0069 E06B     00253          CALL   PWM1_UD         ;
006A C060     00254          GOTO   LOWCYC        ; Loop looking for high signal on PortB0
              00255
              00256  PAGE
              00257 ;
              00258 ; This code segment ensure that all PWM values (period and duty cycle)
              00259 ; are updated at the same time. This is done by ensuring that the Timer
              00260 ; is at least PWM_WIN (0Dh) cycles before the PR1 value
              00261 ; (PR1 - PWM_WIN > TMR1).If not a “glitch” could occur in the PWM wave
              00262 ; form. When only the 1st duty cycle register is latched for this PWM
              00263 ; cycle, and the following PWM periodwill latch the 2nd duty cycle
              00264 ; register.
006B 8406     00265 PWM1_UD BSF CPUSTA, 4          ; Disable Global Interrupts
006C B802     00266             MOVLB   2          ; Bank 2
006D 6A10     00267             MOVFP   TMR1, W    ; Load W reg. with Timer1 value
006E 0414     00268             SUBWF   PR1, 0     ; PR1 - TMR1 -> W reg.
006F 3025     00269             CPFSLT  PWM_WIN    ; Check if Timer1 is about to overflow
0070 C06B     00270             GOTO    PWM1_UD    ; Overflow would have occurred during
              00271                                ;   PWM updates, Delay a few cycles
0071 B803     00272             MOVLB   3          ; Bank 3
0072 6A20     00273             MOVFP   NEW_DC1, W ; Your New PWM MSB
0073 0112     00274             MOVWF   PW1DCH     ; Loaded in duty cycle buffer
0074 6A21     00275             MOVFP   NEW_DC1Q, W; Your New PWM LSB
0075 0110     00276             MOVWF   PW1DCL     ; Loaded in duty cycle buffer
0076 B801     00277             MOVLB   1          ; Back to Bank 1
0077 8C16     00278             BCF     PIR, 4     ; Clear T1 Overflow interrupt flag
0078 8417     00279             BSF     PIE, 4     ; Enable T1 int
0079 8C06     00280             BCF     CPUSTA, 4  ; Enable Global Interrupts
007A B800     00281             MOVLB   0          ; Bank 0
007B 0002     00282             RETURN             ;** this does not need to be implemented
              00283                                ;**  as a subroutine.
              00284 ;
              00285 ; Other Interrupt routines. (Not utilized in this example)
              00286 ;
007C 0005     00287 EXT_INT RETFIE                 ; RA0/INT interrupt routine
              00288                                ;   (NOT used in this program)
007D 0005     00289 TMR0INT RETFIE                 ; TMR0 overflow interrupt routine
              00290                                ;   (NOT used in this program)
007E 0005     00291 T0_INT  RETFIE                 ; RA1/T0CKI interrupt routine
              00292                                ;   (NOT used in this program)
              00293                                ;
007F C02B     00294 SRESET  GOTO START             ; If program became lost, goto
              00295                                ;   START and reinitalize.
              00296 ;
              00297 ;
              00298 ; When the executed address is NOT in the program range, the
              00299 ; 16-bit address should contain all 1’s (a CALL 0x1FFF). At
              00300 ; this location you could branch to a routine to recover or
              00301 ; shut down from the invalid program execution.
              00302 ;
07FF          00303            ORG    END_OF_PROG_MEM ;
07FF C07F     00304            GOTO   SRESET        ; The program has lost it’s mind,
              00305                                ;   do a system reset
              00306                 END
MEMORY USAGE MAP (‘X’ = Used,  ‘-’ = Unused)


0000 : X-------X------- X-------X------- XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
0040 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
07C0 : ---------------- ---------------- ---------------- ---------------X

All other memory blocks unused.

Program Memory Words Used:   101
```

```
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