

<u>AN539</u>

Frequency and Resolution Options for PWM Outputs

INTRODUCTION

The PIC17C42 is equipped with two high frequency Pulse Width Modulation (PWM) outputs. In a pulse width modulated signal the period of the signal is (usually) kept fixed, while the duty cycle is varied. In this application note, we will discuss options in selecting their frequency and resolution.

This application brief assumes that internal clock is used for the time-base, which is typically the preferred set-up. Also, throughout this application brief, PWM1 output is used in examples, timer1 is assumed to be the timebase.

Definition of terms:

<u>Period</u> of a PWM output is the duration after which the PWM pattern will repeat itself.

Frequency of a PWM output is = 1/Period.

<u>Resolution</u> of a PWM output is the granularity with which the duty cycle can be modulated.

In the case of the PIC17C42, when using PWM1 with timer1 as time-base the:

PWM1 period = $[(PR1) + 1] \times 4$ tosc

PWM1 duty cycle = (DC1) x tosc

where PR1 = period register for timer1

DC1 = PW1DCH, PW2DCL concantenated (10-bit value)

tosc = oscillator period

At 16 MHz oscillator frequency, tosc = 62.5 ns. The user can control the frequency of the PWM output by altering the 'period' value of the time-base. For example, if period is chosen to be 100 tosc (PR1 = 18h), then PWM frequency is $1/(100 \times 62.5)$ ns = 160 KHz. Note however that duty cycle resolution is a little less than 7-bits.

FIGURE 1 - PWM OUTPUT

Useful and Common PWM Modes

While a variety of period values can be selected, the following modes would be most commonly used:

<u>10-Bit Mode</u>: In this mode PWM duty cycle has full 10-bit resolution (maximum offered by the PIC17C42). The period register PR1 is set at FFh. PWM period is 1024tosc = 64 µs. PWM frequency is 15.625 KHz. The user must write both PW1DCH and PW1DCL to update PWM output. See Appendix A for an example that code modules 10-bit resolution PWM output (PWM10.LST).

<u>8-Bit Hi-Resolution Mode</u>: In this mode, the user has only an 8-bit quantity to write to the duty-cycle register. Period register is set at 3Fh (63 decimal), such that PWM period is 256 tosc. To write the 8-bit duty-cycle value, first the 8-bit is right shifted two bits. The upper six bits are written to PW1DCH and the lower two bits are written to PW1DCL as follows:

;8-D11	t auty-cycle	value is in W reg
CLRF	TEMP	;
RRCF	WREG	;
RRCF	TEMP	;
RRCF	WREG	;
RRCF	TEMP	;Shift right twice
ANDLW	00111111b	;Mask off two-high bits
MOVPF	WREG, PW1DCH	;Write duty-cycle val-
ues		
MOVED	TEMP. PW1DCL	;

Note that in 8-bit, hi-resolution mode, maximum PWM frequency is attained. For example, at 16 MHz clock, PWM period = $256 \text{ tosc} = 16 \,\mu\text{s}$; PWM frequency = $62.5 \,\text{KHz}$. See appendix B for an example code that generates 8-bit low high resolution PWM output (PWM8HI.LST).



© 1993 Microchip Technology Inc.

FIGURE 2 - VARIOUS PWM MODES



8-Bit Low Resolution Mode

In this mode, the user still has only an 8-bit quantity to write to duty cycle register. However, the desired frequency of the PWM output is less, due to the nature of the application. For example, if the PWM output is being used to drive a motor through a power stage, the power transistors (or devices) due to their switching time will prefer PWM frequency not to exceed certain frequency. In the previous section, we derived an 8-bit resolution PWM output at 62.5 KHz.

To attain a low-resolution PWM output, the PW1DCL is always kept at zero. The 8-bit value is written to PW1DCH. The period (PR1) is set at FFh, i.e. 256 Tcy equals 1024 tosc (15.625 KHz). See Appendix C for an example code that produces 8-bit low resolution PWM output (PWM8LO.LST).

Choosing Resolution and Frequency of PWM Output

Actually, the resolution and the frequency of the PWM output is selectable within certain limits. The user will need to first define the requirements based on the application. There may be an upper limit to the frequency if the PWM is being used to drive motors. On the

other hand, if the PWM is being filtered to generate an analog signal, higher frequency may be desirable. In any case, the lowest frequency achievable (using internal clock for the timer) is (OSC freq/1024). At 16 MHz oscillator input, the lowest PWM frequency possible is 15.625 KHz. At resolutions less than 10-bit, higher frequencies are possible (see Figure 3). For example, if 7-bit resolution is chosen, then the PWM frequencies can be 15-625 KHz, 31.25 KHz, 62.5 KHz or 125 KHz. The reader will note that it's how the 7-bits are placed within the 10-bit possible duty cycle value.

Conversely, if a certain frequency is desired, such as 44 KHz, then referring to Figure 1, resolution can be 8.5-bit or 7.5-bit or 6.5-bit etc.

Summary

The frequency and resolution of the PWM outputs of the PIC17C42 can be traded off against each other to best suit the application. The oscillator frequency can also be varied to adjust PWM frequency, if necessary. External clock should be used as timer time-base to generate very low frequency PWM output.



FIGURE 3 - PWM FREQ VS RESOLUTION



APPENDIX A: PWM10.LST

MPASM	в0.54					PAGE 1
PULSE	WIDTH	MODULATION 3	10 BIT	RESOLUI	ION	
				TITLE LIST	"PULSE WIDTH MOI P=17C42, C=80,	DULATION 10 BIT RESOLUTION" T=ON
				include	"17c42.h"	,
0021 0020 0022		PWI PWI TEI	M_HI M_LO MP	equ equ equ	0x21 0x20 0x22	
		; T ; 1 ; 1 ; t ; ; g ; ;	he use ocatio ransfe enerat	r would ns PWM_H rs these e the re	generate a 16 bi II and PWM_LO byt values directly equired 10 bit PW	it value which is saved in r te. In 10 bit mode, the prog 7 to the Duty Cycle (DC) reg MM.
		;				
		;11 ;11 ;T1 ;k4 ;t. ;T1 ;i: ;T1 ;r2	D bit he max eeps t b the his pr s done he per amps u weep t	resoluti c. period he period most sig ogram is in the iod upda p from 0 akes app	on. Since a 10MH i = 1024x100nS = od constant and w mificant 10 bits a interrupt drive rtcc interrupt, ite is done durin % to 100% duty o prox. 52 secs.	his crystal was used in the t 102.4 uS or 9.8 KHz. This p varies the duty cycle (which s of the 16 bit value PWM_LO en, i.e. the update to the D which then enables the pwm og the pwm interrupt. The pw cycle and then repeats. The
		;				
				ORG	0	
0000	C058			goto	start	
		; rta	c int	ORG	0x10	;vector for rtcc interrupt
0010	C04C		00_1110	goto	service_rtcc	;service rtcc
		,	n int	ORG	0x0020	;vector for pwm interrupt
0020	C03E	: :		goto	service_pwm	;service pwm only
		,		ORG	0x0030	
		; ;i:; ;fc in:	nitial or 10 it pwm	ize inte bit resc 10	ernal hardware to olution pwm.	o generate the output
0030	B802		- <u>-</u> 1	movlb	2	
0031	2910			clrf	tmrl	;clear timer 1 ;used to "drive" pwml
0032	2B14			setf	prl	;set period=9.8 khz
0033	B803			movlb	3	
0034	7221			movfp	PWM_HI,pwldch	;load duty cyl. hi byte
0035	7020			movfp	PWM_LO,pw1dcl	;load duty cycle lo byte
0036	2916			clrf	tcon1	<pre>;tmr1 inc. internally ;as 8 bit counter</pre>
0037	B01B			movlw	00010001B	;start tmrl and
0038	4A17			movpf	wreg,tcon2	;enable pwm1
0039	B801			movlb	1	
003A	2917			clrf	pie	;clr all int. enables
003B	2916			clrf	pir	;clear all interrupts
003C	8307			bsf	_peie	;except peripheral int.
003D	0005			retfie		
		;				
		,				

© 1993 Microchip Technology Inc.

	;every	time a ne	w value is writt	en to the PWM_HI, PWM_LO reg		
	/tmrl	itmrl interrupts is enabled. The DC value are written just				
	, the "	pwm inter	rupt" is enabled	. Here the new period regist		
	, updat	ea. In tr	iis example, peri	od is kept constant at UXII		
	servic	e_pwm				
0000 0000		; if the	e period changed,	write new value here.		
003E B802		movid	2	Select Dank 2		
003F 2B14		seti	pri	/period <- Uxii		
0040 B801		movib	1	disable tmrl int		
0041 8C17		bci	_tmlie	; /		
0042 0005		retfie				
	<i>i</i>		, ,			
	/This	part of t	ne program is ba	sically used to simmulate a		
	, which	would be	e usea to arive t	ne pwm output.		
	, ,			t		
	ithe r	tcc is se	et up to interrup	t every 52 ms.		
0042 5005	init_r	LCC	0.01.00.000			
0043 8008		moviw	001000000	set up rtcc timer		
0044 650A		movip	wreg,rtcsta	, , ,		
0045 290B		ciri	rtcci	Clear rtcc		
0046 290C		clrt	rtech	; /		
0047 B080		movlw	0x80	start pwm at 50%		
0048 0121		movwf	PWM_HI	; /		
0049 2920		clrf	PWM_LO	; /		
004A 8107		bsf	_rtcie	;enable rtcc int.		
004B 0002		return				
	;					
	;Every	rtcc int	errupt, the PWM_	HI&PWM_LO bytes are incremen		
	;Only	the 10 mc	ost significant b	oits are incremented. Once th		
	; .					
	servic	e_rtcc				
004C 8D07		bcf	_rtcir	;reset int flag		
		;do a p	seudo inc of the	10 bit PWM_HI, PWM_LO.		
004D 8804		bcf	_carry	;clear carry		
004E B00B		movlw	0100000b	;load lsb for 10 bit		
004F 0F20		addwf	PWM_LO,1	;add to LSB		
0050 9804		btfsc	_carry	;carry?		
0051 1521		incf	PWM_HI	;yes then inc PWM_HI		
		;now lo	ad the values in	to the Duty Cycle registers		
0052 B803		movlb	3	;bank 3		
0053 7020		movfp	PWM_LO,pw1dcl	;load lo value		
0054 7221		movfp	PWM_HI,pwldch	;load hi value		
0055 B801		movlb	1			
0056 8417		bsf	_tmlie	;enable tmr1 int		
0057 0005		retfie				
	;					
	;					
	start					
0058 8406		bsf	_glintd	;disable interrupts		
0059 E043		call	init_rtcc	;initailize the RTCC tmr		
				;for test purposes		
005A E030		call	init_pwm10	;initialize pwm		
005B C05B	loop	goto	loop	;spin wheels		
	;					
		END				
Errors : 0						

Warnings : 0

AppendixB:PWM8HI.LST

MPASI	4 B0.54						PAGE 1
PULSI	E WIDTH MODULATI	ON 8 BIT	HIGH RE	SOLUTION	1		
			TITLE LIST	"PULSE	WIDTH MOI P=17C42,	DULATION 8 BIT HIGH C=80, T=ON	RESOLUTION"
			include		"17c42.h		
0021		PWM_HI PWM_LO	equ equ	0x21 0x20			
0022		TEMP	equ	0x22			
		;The us ;locati ;transf ;genera ;	er would ons PWM_1 ers the s te the re	generat HI and H 8 bit va equired	te a 16 bi PWM_LO by alues to 4 8 bit hi	it value which is s te. In 8 bit hi-res the lo Duty Cycle (-res PWM.	aved in r mode, th DC) regis
		; ;This i :8 bit	s a shor	t progra	am to demo	onstrate how to gen	erate PWM
		; The ma ; keeps ; to the ; This p ; is don ; The pe ; ramps ; sweep	x. period the period most sign rogram is e in the riod upday up from takes app	d = 2562 od const gnificar s intern rtcc in ate is c 0% to 10 prox. 13	cloons = 3 cant and y nt 10 bits rupt drive nterrupt, done durin 00% duty o 3.3 secs.	25.6uS or 39KHz. Th varies the duty cyc s of the 16 bit val en, i.e. the update which then enables and the pwm interrup cycle and then repe	In the te is progra le (which ue PWM_LO to the D to the pwm t. The pw ats. The
		;					
		;	ORG	0			
0000	C063		goto	start			
		;	ORG	0x10		;vector for rtcc i	nterrupt
		rtcc_in	t				
0010	C054		goto	service	e_rtcc	;service rtcc	
		1	ORG	0x0020		;vector for pwm in	terrupt
0020	0046	pwm_int	asta	aomia		· acentiac num en lu	
0020	CU46	;	goto	service	e_pwm	,service pwm only	
			ORG	0x0030			
		; ;initia	lize int	ernal ha	ardware to	o generate the pwm	output
		init_pw	m8hi			5	*
0030	B802		movlb	2		·aloon timon 1	
0031	2910		CILL			;used to "drive" p	wml
0032	B062		movlw	62		;set period=39khz	
0033	0114		movwf	pr1		; /	
0034	B803		movlb	3			
0035	2922		cirt	TEMP DUM UT		;TEMP = mask for p	widci
0030	6AZI 1907		movip	PWM_HI,	,wreg	rotate bi through	byte
0037	1922		rrcf	TEMP		;rotate into lo by	te
0039	190A		rrcf	wreq		repeat for 2nd 1s	 b
003A	1922		rrcf	TEMP		; /	
003B	B53F		andlw	b'00111	1111'	;mask hi bits	
003C	4012		movpf	W,pwldd	ch	;save in high	
003D	7022		movfp	TEMP, pv	vldcl	;save in low	
003E	2916		cirf	tconl		<pre>;tmrl inc. interna ;as 8 bit counter</pre>	117
003F	B011		movlw	b'00010	001'	;start tmr1 and	
0040	4017		movpf	W,tcon2	2	;enable pwm1	
0041	B801		movlb	1			

© 1993 Microchip Technology Inc.

PWM Frequency and Resolution

0042 2917 0043 2916 0044 8307 0045 0005	clrf clrf bsf retf	pie pir _peie ie	<pre>;clr all int. enables ;clear all interrupts ;except peripheral int.</pre>
	;		
	; ;everytime a ;tmrl interr ;the "pwm in ;updated. In	new value is wri upts is enabled. ' terrupt" is enable this example, pe	tten to the PWM_HI, PWM_LO reg The DC value are written just ed. Here the new period regist riod is kept constant at 62 Tc
	jif	the period change	d, write new value here.
0046 B802	movl	b 2	;select bank 2
0047 B062	movl	w 62	;period = 62 Tcyl.
0048 0114	movw	f prl	; /
0049 B801	movl	b 1	;disable tmr1 int
004A 8C17	bcf	_tmlie	; /
004B 0005	. reti	10	
	, ;This part o	f the program is l	basically used to simmulate a
	;which would	be used to drive	the pwm output.
	;the rtcc is	set up to interro	upt every 52 mS.
004C B020	movl	w b'00100000'	set up rtcc timer
004D 650A	movf	p wreg,rtcsta	; /
004E 290B	clrf	rtccl	;clear rtcc
004F 290C	clrf	rtcch	; /
0050 B031	movl	w 31	;init pwm at 50%
0051 0121	movw	f PWM_HI	;save in high
0052 8107	bsf	_rtcie	;enable rtcc int.
0053 0002	. retu	rn	
	, Every rtcc	interrupt, the PW	M HI&PWM LO bytes are incremen
	/Every itee	inceriape, che rwi	"_IIIdrwh_LO Dyces are Incremen
	;Only the 8 ;	most significant l	bits are incremented.
	;Only the 8 ; service_rtcc	most significant l	bits are incremented.
0054 8D07	;Only the 8 ; service_rtcc bcf	most significant] _rtcir	<pre>int flag</pre>
0054 8D07	;Only the 8 ; service_rtcc bcf	most significant] _rtcir	<pre>its are incremented. ;reset int flag ;do a pseudo inc of the 8 bit PWM_HI. ;ice pum UI</pre>
0054 8D07 0055 1521	;Only the 8 ; service_rtcc bcf incf	most significant] _rtcir PWM_HI	<pre>bits are incremented. ;reset int flag ;do a pseudo inc of the 8 bit PWM_HI. ;inc PWM_HI ;now load the values into the Duty Cycle</pre>
0054 8D07 0055 1521 0056 B803	;Only the 8 ; service_rtcc bcf incf	most significant] _rtcir PWM_HI b 3	<pre>bits are incremented. ;reset int flag ;do a pseudo inc of the 8 bit PWM_HI. ;inc PWM_HI ;now load the values into the Duty Cycle ;bank 3</pre>
0054 8D07 0055 1521 0056 B803 0057 2922	;Only the 8 ; service_rtcc bcf incf movl clrf	most significant] _rtcir PWM_HI b 3 TEMP	<pre>bits are incremented. ;reset int flag ;do a pseudo inc of the 8 bit PWM_HI. ;inc PWM_HI ;now load the values into the Duty Cycle ;bank 3 ;TEMP = mask for pwldcl</pre>
0054 8D07 0055 1521 0056 B803 0057 2922 0058 6A21	;Only the 8 ; service_rtcc bcf incf movl clrf movf	most significant _rtcir PWM_HI b 3 TEMP p PWM_HI,wreg	<pre>bits are incremented. ;reset int flag ;do a pseudo inc of the 8 bit PWM_HI. ;inc PWM_HI ;now load the values into the Duty Cycle ;bank 3 ;TEMP = mask for pwldcl ;get duty cyl. hi byte</pre>
0054 8D07 0055 1521 0056 B803 0057 2922 0058 6A21 0059 190A	;Only the 8 ; service_rtcc bcf incf movl clrf movf rrcf	most significant _rtcir PWM_HI b 3 TEMP p PWM_HI,wreg wreg	<pre>bits are incremented. ;reset int flag ;do a pseudo inc of the 8 bit PWM_HI. ;inc PWM_HI ;now load the values into the Duty Cycle ;bank 3 ;TEMP = mask for pwldcl ;get duty cyl. hi byte ;rotate hi through carry</pre>
0054 8D07 0055 1521 0056 B803 0057 2922 0058 6A21 0059 190A 0054 1922	;Only the 8 ; service_rtcc bcf incf movl clrf movf rrcf	most significant _rtcir PWM_HI b 3 TEMP p PWM_HI,wreg wreg TEMP	<pre>bits are incremented. ;reset int flag ;do a pseudo inc of the 8 bit PWM_HI. ;inc PMM_HI ;now load the values into the Duty Cycle ;bank 3 ;TEMP = mask for pwldcl ;get duty cyl. hi byte ;rotate hi through carry ;rotate into lo byte</pre>
0054 8D07 0055 1521 0056 B803 0057 2922 0058 6A21 0059 190A 005A 1922 005B 190A	;Only the 8 ; service_rtcc bcf incf movl clrf movf rrcf rrcf rrcf	most significant _rtcir PWM_HI b 3 TEMP p PWM_HI,wreg wreg TEMP wreg	<pre>bits are incremented. ;reset int flag ;do a pseudo inc of the 8 bit PWM_HI. ;inc PWM_HI ;now load the values into the Duty Cycle ;bank 3 ;TEMP = mask for pwldcl ;get duty cyl. hi byte ;rotate hi through carry ;rotate into lo byte ;repeat for 2nd lsb</pre>
0054 8D07 0055 1521 0056 B803 0057 2922 0058 6A21 0059 190A 005A 1922 005B 190A 005C 1922	;Only the 8 ; service_rtcc bcf incf movl clrf movf rrcf rrcf rrcf	most significant _rtcir PWM_HI b 3 TEMP p PWM_HI,wreg wreg TEMP wreg TEMP	<pre>bits are incremented. ;reset int flag ;do a pseudo inc of the 8 bit PWM_HI. ;inc PWM_HI ;now load the values into the Duty Cycle ;bank 3 ;TEMP = mask for pwldcl ;get duty cyl. hi byte ;rotate hi through carry ;rotate into lo byte ;repeat for 2nd lsb ;</pre>
0054 8D07 0055 1521 0056 B803 0057 2922 0058 6A21 0059 190A 005A 1922 005B 190A 005C 1922 005D B53F	;Only the 8 ; service_rtcc bcf incf movl clrf movf rrcf rrcf rrcf rrcf andl	<pre>most significant lrtcir PWM_HI b 3 TEMP P PWM_HI,wreg wreg TEMP wreg TEMP w b'00111111' f proproduct</pre>	<pre>bits are incremented. ;reset int flag ;do a pseudo inc of the 8 bit PWM_HI. ;inc PWM_HI ;now load the values into the Duty Cycle ;bank 3 ;TEMP = mask for pwldcl ;get duty cyl. hi byte ;rotate hi through carry ;rotate into lo byte ;repeat for 2nd lsb ; / ;mask hi bits ; /;;mask hi bits ; /;;mask hi bits</pre>
0054 8D07 0055 1521 0056 B803 0057 2922 0058 6A21 0059 190A 005A 1922 005B 190A 005C 1922 005D B53F 005E 4A12	;Only the 8 ; service_rtcc bcf incf movl clrf movf rrcf rrcf rrcf andl movp	<pre>most significant lrtcir PWM_HI b 3 TEMP p PWM_HI,wreg wreg TEMP wreg TEMP w b'0011111' f wreg,pwldch TEMP pwldch TEMP pwldch</pre>	<pre>bits are incremented. ;reset int flag ;do a pseudo inc of the 8 bit PWM_HI. ;inc PWM_HI ;now load the values into the Duty Cycle ;bank 3 ;TEMP = mask for pwldcl ;get duty cyl. hi byte ;rotate hi through carry ;rotate into lo byte ;repeat for 2nd lsb ;</pre>
0054 8D07 0055 1521 0056 B803 0057 2922 0058 6A21 0059 190A 005A 1922 005B 190A 005C 1922 005D B53F 005E 4A12 005F 7022	;Only the 8 ; service_rtcc bcf incf movl clrf movf rrcf rrcf rrcf andl movp movf	<pre>most significant lrtcir PWM_HI b 3 TEMP p PWM_HI,wreg wreg TEMP wreg TEMP w b'0011111' f wreg,pwldch p TEMP,pwldcl b 1</pre>	<pre>bits are incremented. ;reset int flag ;do a pseudo inc of the 8 bit PWM_HI. ;inc PWM_HI ;now load the values into the Duty Cycle ;bank 3 ;TEMP = mask for pwldcl ;get duty cyl. hi byte ;rotate hi through carry ;rotate into lo byte ;repeat for 2nd lsb ;</pre>
0054 8D07 0055 1521 0056 B803 0057 2922 0058 6A21 0059 190A 005A 1922 005B 190A 005C 1922 005D B53F 005E 4A12 005F 7022 006D B801 0061 8417	;Only the 8 ; service_rtcc bcf incf movl clrf movf rrcf rrcf rrcf rrcf andl movp movf movf bsf	<pre>most significant l _rtcir PWM_HI b 3 TEMP p PWM_HI,wreg wreg TEMP wreg TEMP w b'0011111' f wreg,pwldch p TEMP,pwldcl b 1 tmlie</pre>	<pre>bits are incremented. ;reset int flag ;do a pseudo inc of the 8 bit PWM_HI. ;inc PWM_HI ;now load the values into the Duty Cycle ;bank 3 ;TEMP = mask for pwldcl ;get duty cyl. hi byte ;rotate hi through carry ;rotate into lo byte ;repeat for 2nd lsb ;</pre>
0054 8D07 0055 1521 0056 B803 0057 2922 0058 6A21 0059 190A 005A 1922 005B 190A 005C 1922 005D B53F 005E 4A12 005F 7022 006D B801 0061 8417 0062 0005	;Only the 8 ; service_rtcc bcf incf movl clrf movf rrcf rrcf rrcf rrcf andl movp movf movl bsf retf	<pre>most significant l _rtcir PWM_HI b 3 TEMP p PWM_HI,wreg wreg TEMP wreg TEMP w b'0011111' f wreg,pwldch p TEMP,pwldcl b 1 _tmlie ie</pre>	<pre>bits are incremented. ;reset int flag ;do a pseudo inc of the 8 bit PWM_HI. ;inc PMM_HI ;now load the values into the Duty Cycle ;bank 3 ;TEMP = mask for pwldcl ;get duty cyl. hi byte ;rotate hi through carry ;rotate into lo byte ;repeat for 2nd lsb ;</pre>
0054 8D07 0055 1521 0056 B803 0057 2922 0058 6A21 0059 190A 005A 1922 005B 190A 005C 1922 005D B53F 005E 4A12 005F 7022 0060 B801 0061 8417 0062 0005	;Only the 8 ; service_rtcc bcf incf movl clrf movf rrcf rrcf rrcf andl movp movf movl sf ;	<pre>most significant 1 _rtcir PWM_HI b 3 TEMP p PWM_HI,wreg wreg TEMP wreg TEMP w b'00111111' f wreg,pwldch p TEMP,pwldcl b 1 _tmlie ie</pre>	<pre>bits are incremented. ;reset int flag ;do a pseudo inc of the 8 bit PWM_HI. ;inc PWM_HI ;now load the values into the Duty Cycle ;bank 3 ;TEMP = mask for pwldcl ;get duty cyl. hi byte ;rotate hi through carry ;rotate into lo byte ;repeat for 2nd lsb ;</pre>
0054 8D07 0055 1521 0056 B803 0057 2922 0058 6A21 0059 190A 005A 1922 005B 190A 005C 1922 005D B53F 005E 4A12 005F 7022 005F 7022 005G B801 0061 8417 0062 0005	;Only the 8 ; service_rtcc bcf incf movl clrf movf rrcf rrcf rrcf andl movp movf wovf sf retf ;	<pre>most significant lrtcir PWM_HI b 3 TEMP P PWM_HI,wreg wreg TEMP w b'0011111' f wreg,pwldch p TEMP,pwldcl b 1tmlie ie</pre>	<pre>bits are incremented. ;reset int flag ;do a pseudo inc of the 8 bit PWM_HI. ;inc PWM_HI ;now load the values into the Duty Cycle ;bank 3 ;TEMP = mask for pwldcl ;get duty cyl. hi byte ;rotate hi through carry ;rotate into lo byte ;repeat for 2nd lsb ;</pre>
0054 8D07 0055 1521 0056 B803 0057 2922 0058 6A21 0059 190A 005A 1922 005B 190A 005C 1922 005D B53F 005E 4A12 005F 7022 0060 B801 0061 8417 0062 0005	; Only the 8 ; service_rtcc bcf incf movl clrf movf rrcf rrcf rrcf andl movp movf ssf ; ; start	<pre>most significant lrtcir PWM_HI b 3 TEMP p PWM_HI,wreg wreg TEMP w b'0011111' f wreg,pwldch p TEMP,pwldcl b 1tmlie ie</pre>	<pre>bits are incremented. ;reset int flag ;do a pseudo inc of the 8 bit PWM_HI. ;inc PWM_HI ;now load the values into the Duty Cycle ;bank 3 ;TEMP = mask for pwldcl ;get duty cyl. hi byte ;rotate hi through carry ;rotate into lo byte ;repeat for 2nd lsb ;</pre>
0054 8D07 0055 1521 0056 B803 0057 2922 0058 6A21 0059 190A 005A 1922 005B 190A 005C 1922 005D B53F 005E 4A12 005F 7022 0060 B801 0061 8417 0062 0005	; Only the 8 ; service_rtcc bcf incf mov1 clrf mov1 rrcf rrcf rrcf rrcf rrcf ; ; start bsf call	<pre>most significant l _rtcir PWM_HI b 3 TEMP p PWM_HI,wreg wreg TEMP wreg,pwldch f wreg,pwldch p TEMP,pwldcl b 1 _tmlie ie _glintd init_rtcc</pre>	<pre>bits are incremented. ;reset int flag ;do a pseudo inc of the 8 bit PWM_HI. ;inc PWM_HI ;now load the values into the Duty Cycle ;bank 3 ;TEMP = mask for pwldcl ;get duty cyl. hi byte ;rotate hi through carry ;rotate into lo byte ;repeat for 2nd lsb ;</pre>
0054 8D07 0055 1521 0056 B803 0057 2922 0058 6A21 0059 190A 005A 1922 005B 190A 005C 1922 005D B53F 005E 4A12 005F 7022 0060 B801 005F 7022 0060 B801 0061 8417 0062 0005	; Only the 8 ; service_rtcc bcf incf movl clrf movf rrcf rrcf rrcf rrcf ; ; start bsf call	<pre>most significant l _rtcir PWM_HI b 3 TEMP p PWM_HI,wreg TEMP w b'0011111' f wreg,pwldch p TEMP,pwldcl b 1 _tmlie ie _glintd init_rtcc </pre>	<pre>bits are incremented. ;reset int flag ;do a pseudo inc of the 8 bit PWM_HI. ;inc PMM_HI ;now load the values into the Duty Cycle ;bank 3 ;TEMP = mask for pwldcl ;get duty cyl. hi byte ;rotate hi through carry ;rotate into lo byte ;repeat for 2nd lsb ;</pre>
0054 8D07 0055 1521 0056 B803 0057 2922 0058 6A21 0059 190A 005A 1922 005B 190A 005C 1922 005D B53F 005E 4A12 005F 7022 0060 B801 0061 8417 0062 0005 0063 8406 0064 E04C	;Only the 8 ; service_rtcc bcf incf movl clrf movf rrcf rrcf rrcf andl movp movf bsf retf ; ; start bsf call	<pre>most significant lrtcir PWM_HI b 3 TEMP P PWM_HI,wreg wreg TEMP w b'0011111' f wreg,pwldch p TEMP,pwldcl b 1tmlie ieglintd init_rtcc init_pwm8hi </pre>	<pre>bits are incremented. ;reset int flag ;do a pseudo inc of the 8 bit PWM_HI. ;inc PWM_HI ;now load the values into the Duty Cycle ;bank 3 ;TEMP = mask for pwldcl ;get duty cyl. hi byte ;rotate hi through carry ;rotate into lo byte ;repeat for 2nd lsb ;</pre>
0054 8D07 0055 1521 0056 B803 0057 2922 0058 6A21 0059 190A 005A 1922 005B 190A 005C 1922 005D B53F 005E 4A12 005F 7022 005D 8801 0061 8417 0062 0005	; Only the 8 ; service_rtcc bcf incf movl clrf movf rrcf rrcf rrcf andl movp movf sf retf ; ; start bsf call call loop goto	<pre>most significant lrtcir PWM_HI b 3 TEMP p PWM_HI,wreg wreg TEMP w b'0011111' f wreg,pwldch p TEMP,pwldcl b 1tmlie ieglintd init_rtcc init_pwm8hi loop</pre>	<pre>bits are incremented. ;reset int flag ;do a pseudo inc of the 8 bit PWM_HI. ;inc PWM_HI ;now load the values into the Duty Cycle ;bank 3 ;TEMP = mask for pwldcl ;get duty cyl. hi byte ;rotate hi through carry ;rotate into lo byte ;repeat for 2nd lsb ; / ;mask hi bits ;save in high ;save in low ;enable tmr1 int ;disable interrupts ;initialize the RTCC tmr ;for test purposes ;initialize 8 bit pwm ;spin wheels.</pre>
0054 8D07 0055 1521 0056 B803 0057 2922 0058 6A21 0059 190A 005A 1922 005B 190A 005C 1922 005D B53F 005E 4A12 005F 7022 0060 B801 0061 8417 0062 0005 0063 8406 0064 E04C	; Only the 8 ; service_rtcc bcf incf movl clrf movf rrcf rrcf rrcf andl movp movf start bsf call loop goto ;	<pre>most significant l _rtcir PWM_HI b 3 TEMP p PWM_HI,wreg wreg TEMP w b'0011111' f wreg,pwldch p TEMP,pwldcl b 1 _tmlie ie _glintd init_rtcc init_pwm8hi loop</pre>	<pre>bits are incremented. ;reset int flag ;do a pseudo inc of the 8 bit PWM_HI. ;inc PWM_HI ;now load the values into the Duty Cycle ;bank 3 ;TEMP = mask for pwldcl ;get duty cyl. hi byte ;rotate hi through carry ;rotate into lo byte ;repeat for 2nd lsb ;</pre>
0054 8D07 0055 1521 0056 8803 0057 2922 0058 6A21 0059 190A 005A 1922 005B 190A 005C 1922 005D B53F 005E 4A12 006F 7022 0060 8801 0061 8417 0062 0005 0063 8406 0064 E04C 0065 E030 0066 C066 Errors : 0	; Only the 8 ; service_rtcc bcf incf mov1 clrf movf rrcf rrcf rrcf rrcf ; ; start bsf call bsf call loop goto ; END	<pre>most significant l _rtcir PWM_HI b 3 TEMP p PWM_HI,wreg wreg TEMP w b'0011111' f wreg,pwldch p TEMP,pwldcl b 1 _tmlie ie _glintd init_rtcc init_pwm8hi loop</pre>	<pre>bits are incremented. ;reset int flag ;do a pseudo inc of the 8 bit PWM_HI. ;inc PWM_HI ;now load the values into the Duty Cycle ;bank 3 ;TEMP = mask for pwldcl ;get duty cyl. hi byte ;rotate hi through carry ;rotate into lo byte ;repeat for 2nd lsb ;</pre>

AppendixC:PWM8LO.LST

MPASM B0.54 PULSE WIDTH MODULAT	ION 8 BIT	LOW RES	DLUTION	PAGE 1
	T	ITLE " IST P	PULSE WIDTH MODU =17C42, T=ON, C=	LATION 8 BIT LOW RESOLUTION" 80
	includ	e "17c42	.h″	
0021 0020 0022	<pre>PWM_HI PWM_LO TEMP ;The usen ;location ;transfer ; ;</pre>	equ equ equ would s PWM_H s the 8	0x21 0x20 0x22 generate a 16 bit I and PWM_LO byte hi-byte value di	t value which is saved in r e. In 8 bit lo-res mode, th irectly to the PW1DCH regis
	;This is ;8 bit 1d ;The max ;keeps th ;to the r ;This pro; ;is done ;The per: ;ramps up ;sweep ta ; ;	a short by resolu- . period he period nost sign ogram is in the m iod updato from 05 akes approximation	program to demor ution. Since a 5. = 1024x100nS = 1 d constant and va inficant 8 bits of interrupt driver etcc interrupt, w te is done during & to 100% duty cy cox. 52 secs.	Astrate how to generate PWM 068Mhz crystal was used in 102.4 uS or 9.8 KHz. This p aries the duty cycle (which of the 16 bit value PWM_LO& h, i.e. the update to the D which then enables the pwm g the pwm interrupt. The pw ycle and then repeats. The
0000 C053		ORG goto	0 start	
	;	ORG	0x10	;vector for rtcc interrupt
	rtcc_int			
0010 C04C	;	goto	service_rtcc	;service rtcc
	num int	ORG	0x0020	;vector for pwm interrupt
0020 C03E	;	goto	service_pwm	;service pwm only
	;	ORG	0x0030	
	;initial: ;for 8 b: init_pwm8	ize inte it low re 3lo	rnal hardware to esolution pwm	generate the output
0030 B802		movlb	2 tmr1	aloar timor 1
0031 2910		citt	curri	;used to "drive" pwml
0032 2B14 0033 B803		movlb	3	/set period=9.8 kiiz
0034 7221 0035 2910 0036 2916		movfp clrf clrf	PWM_HI,pwldch pwldcl tconl	<pre>;load duty cyl. hi byte ;clear lo byte ;tmr1 inc. internally ;as 8 bit counter</pre>
0037 B011 0038 4A17 0039 B801		movlw movpf movlb	b'00010001' wreg,tcon2	<pre>;start tmr1 and ;enable pwm1</pre>
003A 2917 003B 2916 003C 8307 003D 0005		clrf clrf bsf retfie	pie pir _peie	<pre>;clr all int. enables ;clear all interrupts ;except peripheral int.</pre>
	; ;			

4

			;everyti ;tmr1 in ;the "pw ;updated service_	ime a new nterrupts vm interr d. In thi _pwm	value is writte is enabled. The upt" is enabled. s example, peric	en to the PWM_HI, PWM_LO reg DC value are written just Here the new period regist d is kept constant at Oxff
000-				, II LINE F	period changed, v	write new value nere.
003E	8802			movid	2	/select bank 2
003F	2B14			setf	prl	<pre>;period <- 0xff</pre>
0040	B801			movlb	1	disable tmrl int
0041	8C17			bcf	_tmlie	; /
0042	0005			retfie		
			;			
			;This pa ;which v ;	art of th would be	e program is bas used to drive th	ically used to simmulate a le pwm output.
			init rto	70 13 300	up co inceriape	every 52 mo.
0043	B020		11110_100	movlw	b'00100000'	set up rtcc timer
0043	6500			movfp	W rtcsta	; /
0045	2000			alrf	rtaal	alear rtag
0045	2000			clrf	rtach	; /
0040	200C 2080			mowlw	0~80	begin DWM at 50% dg
0047	0121			movwf	DWM HT	: /
0010	2020			alrf	DWM LO	; /
0045	2J20 9107			baf	rtaio	ionable rtag int
004A	0002			roturn	_rtcre	fenable flee filt.
			; ;Every n ;Only th ;	rtcc inte 1e 8 most	rrupt, the PWM_H significant bit	U&PWM_LO bytes are incremen s are incremented.
			service_	_rtcc		
004C	8D07			bcf	_rtcir	;reset int flag ;do a inc of the 8 bit PWM_HI
004D	1521			incf	PWM_HI	
						;now load the values into the Duty Cycle
004E	B803			movlb	3	
004F	7221			movfp	PWM_HI,pwldch	;load hi byte
0050	B801			movlb	1	
0051	8417			bsf	_tmlie	;enable tmr1 int
0052	0005			retfie		
			; ;			
			start			
0053	8406			bsf	qlintd	disable interrupts
0054	E043			call	init_rtcc	;initailize the RTCC tmr ;for test purposes
0055	E030			call	init_pwm8lo	;initialize pwm
0056	C056		loop	goto	loop	;spin wheels.
Error Warni	s : ngs :	0	;	END		

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

All rights reserved. © 1995, Microchip Technology Incorporated, USA.

Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip's products as critical components in life support systems is not authorized except with express written approval by Microchip No licenses are conveyed, implicitly or otherwise, under any intellectual property rights. The Microchip logo and name are registered trademarks of Microchip Technology Inc. All rights reserved. All other trademarks mentioned herein are the property of their respective companies.