Having a good time

Exploring PIC timers

Part 1 : TIMER0

Not all timers are the same. Some are only 8 bit, some have a pre-scaler only, some are 8 or 16 bit and while Microchip seems to try and keep the various timers the same across the devices, there are some differences. I hope that these notes will help with the understanding of all timers across all devices.

I will be using the PIC16F1779

TIMER0 on this device :

21.0 TIMER0 MODULE

The Timer0 module is an 8-bit timer/counter with the following features:

- 8-bit timer/counter register (TMR0)
- 8-bit prescaler (independent of Watchdog Timer)
- Programmable internal or external clock source
- Programmable external clock edge selection
- Interrupt-on-overflow
- TMR0 can be used to gate Timer1

When TIMER0 is in timer mode, it will increment every instruction cycle(which is Fosc/4) if no pre scaler is used. It is not explicitly obvious by looking at the registers where to select between timer or counter mode. Selecting between timer or counter mode is in the Option register(OPTION_REG) shown on next page. Bit 5 of OPTION_REG.

Timer mode : OPTION_REG→TMR0CS = 0

Counter mode : OPTION_REG→TMR0CS = 1

REGISTER 21-1: OPTION_REG: OPTION REGISTER

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
$\overline{\text{WPUEN}}$	INTEDG	TMR0CS	TMR0SE	PSA		PS[2:0]	
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	

bit 7 **$\overline{\text{WPUEN}}$**: Weak Pull-Up Enable bit
1 = All weak pull-ups are disabled (except $\overline{\text{MCLR}}$, if it is enabled)
0 = Weak pull-ups are enabled by individual WPUx latch values

bit 6 **INTEDG**: Interrupt Edge Select bit
1 = Interrupt on rising edge of INT pin
0 = Interrupt on falling edge of INT pin

bit 5 **TMR0CS**: Timer0 Clock Source Select bit
1 = Transition on T0CKI pin
0 = Internal instruction cycle clock (FOSC/4)

bit 4 **TMR0SE**: Timer0 Source Edge Select bit
1 = Increment on high-to-low transition on T0CKI pin
0 = Increment on low-to-high transition on T0CKI pin

bit 3 **PSA**: Prescaler Assignment bit
1 = Prescaler is not assigned to the Timer0 module
0 = Prescaler is assigned to the Timer0 module

bit 2-0 **PS[2:0]**: Prescaler Rate Select bits

Bit Value	Timer0 Rate
000	1 : 2
001	1 : 4
010	1 : 8
011	1 : 16
100	1 : 32
101	1 : 64
110	1 : 128
111	1 : 256

Not all of these bits are for TIMER0.

TIMER0 will increment on every instruction cycle(if no pre scaler) and then interrupt on overflow from 0xFF->0x00. If a pre scaler is used, the increment per instruction cycle will be divided by the selected pre scaler amount. EG. If you select a pre scaler of 4(PS=001) it will take 4 instruction cycles for the TIMER0 to increment, similarly, if you select 64(PS=101) it will take 64 instruction cycles for one increment of TIMER0.

Math :

If Fosc = 4MHz then Fosc/4 = 1MHz. The timer will increment every 1/1000000 = 1us. With no pre scaler we will get a 255us timer. If we use our two example pre scalers our timer will give us the following

Pre scaler 1:4 = 4 * 255us = 1.020ms

Pre scaler 1:64 = 64 * 255 = 16.320ms

To assign the pre scaler to TIMER0, clear the PSA bit of the OPTION_REG(3)

To select the pre scaler value, set the PS bits of the OPTION_REG(0-2) according to the values in the register definition.

To get more precise timing the TIMER0 register can be pre loaded with a value to start counting up from. Using our 64 pre scaler example above, if we want to get 16ms, we will load the TIMER0 register with 5(16320-16000=320, 320/64=5) The timer will then count up to 255 and since we already have 5 in the register we will get 250 counts.

250 * 64 = 16ms.

Please note that these numbers only work for this chosen Fosc.

When the timer overflows and interrupts, you will need to clear the flag in the interrupt service routine as well as write the '5' back into the timer as it does not auto load this value.

To enable the TIMER0 interrupt

INTCONbits.TMR0IF = 0;//clear the flag first

INTCONbits.TMR0IE = 1;//enable timer 0 interrupt

To re write the TIMER0 offset inside the ISR

TMR0 = 0x05;//write the offset into the timer 0 register to achieve 16ms.

PLEASE TAKE NOTE OF THE FOLLOWING

When TMR0 is written, the increment is inhibited for two instruction cycles immediately following the write.

Note:	The value written to the TMR0 register can be adjusted, in order to account for the two instruction cycle delay when TMR0 is written.

TIMER0 counter mode

In counter mode, the TIMER0 register will increment on every falling or rising edge of T0CKI pin. The pin allocation table shows us that T0CKI is on RA4. If you want to use this pin for TIMER0 counter, you will need to set it as an input using TRISA register.

TRISAbits.RA4 = 1;

selecting to count on either rising or falling edge is bit 4 of OPTION_REG(TMR0SE)

increment on falling edge OPTION_REG -> TMR0SE = 1

increment on rising edge OPTION_REG -> TMR0SE = 0

interrogate the TIMER0 register to retrieve the counter value.

Take note of the following note from the data sheet

21.1.6 OPERATION DURING SLEEP

Timer0 cannot operate while the processor is in Sleep mode. The contents of the TMR0 register will remain unchanged while the processor is in Sleep mode.