2

PIC18 MICROCONTROLLER { Timers}

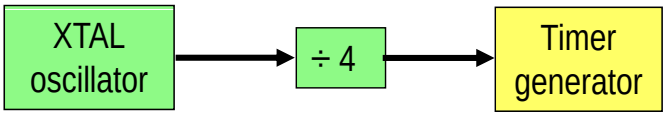
UTHM JOHOR

Mohamad Fauzi Zakaria
http://fkee.uthm.edu.my/mfauzi

Timers in PIC18F4550

Timer	8-bit	16-bit
0	Yes	Yes
1	No	Yes
2	Yes	No
3	No	Yes

- Clock source can be internal or external
- Internal clock: Fosc/4 is feed for generating Timer



- External clock: used as Counter (Only Timer2 cannot be used as counter).

- Basic registers of the timer
  - PIC18 has 16 bits wide timer: TMRxL and TMRxH
  - Each timer has the TCON (timer control)

Timer0

3

REGISTER 11-1: T0CON: TIMER0 CONTROL REGISTER

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **TMR0ON:** Timer0 On/Off Control bit
1 = Enables Timer0
0 = Stops Timer0

bit 6 **T08BIT**: Timer0 8-Bit/16-Bit Control bit
1 = Timer0 is configured as an 8-bit timer/counter
0 = Timer0 is configured as a 16-bit timer/counter

bit 5 **T0CS**: Timer0 Clock Source Select bit
1 = Transition on T0CKI pin
0 = Internal instruction cycle clock (CLKO)

If T0CS = 1, the clock source is external signal from RA4/T0CK1.

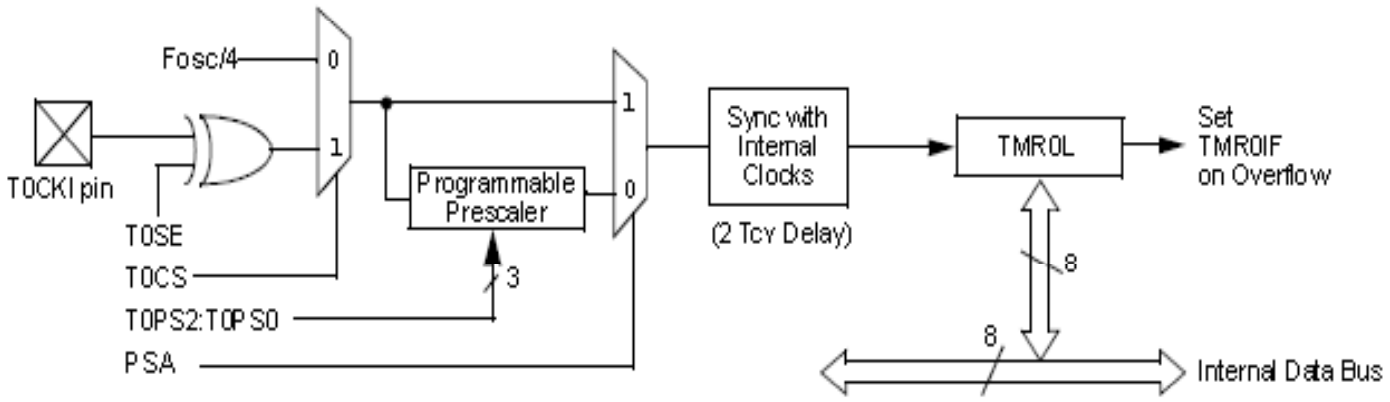
bit 4 **T0SE**: Timer0 Source Edge Select bit
1 = Increment on high-to-low transition on T0CKI pin
0 = Increment on low-to-high transition on T0CKI pin

bit 3 **PSA**: Timer0 Prescaler Assignment bit
1 = TImer0 prescaler is NOT assigned. Timer0 clock input bypasses prescaler.
0 = Timer0 prescaler is assigned. Timer0 clock input comes from prescaler output.

bit 2-0 **T0PS2:T0PS0**: Timer0 Prescaler Select bits
111 = 1:256 Prescale value
110 = 1:128 Prescale value
101 = 1:64 Prescale value
100 = 1:32 Prescale value
011 = 1:16 Prescale value
010 = 1:8 Prescale value
001 = 1:4 Prescale value
000 = 1:2 Prescale value

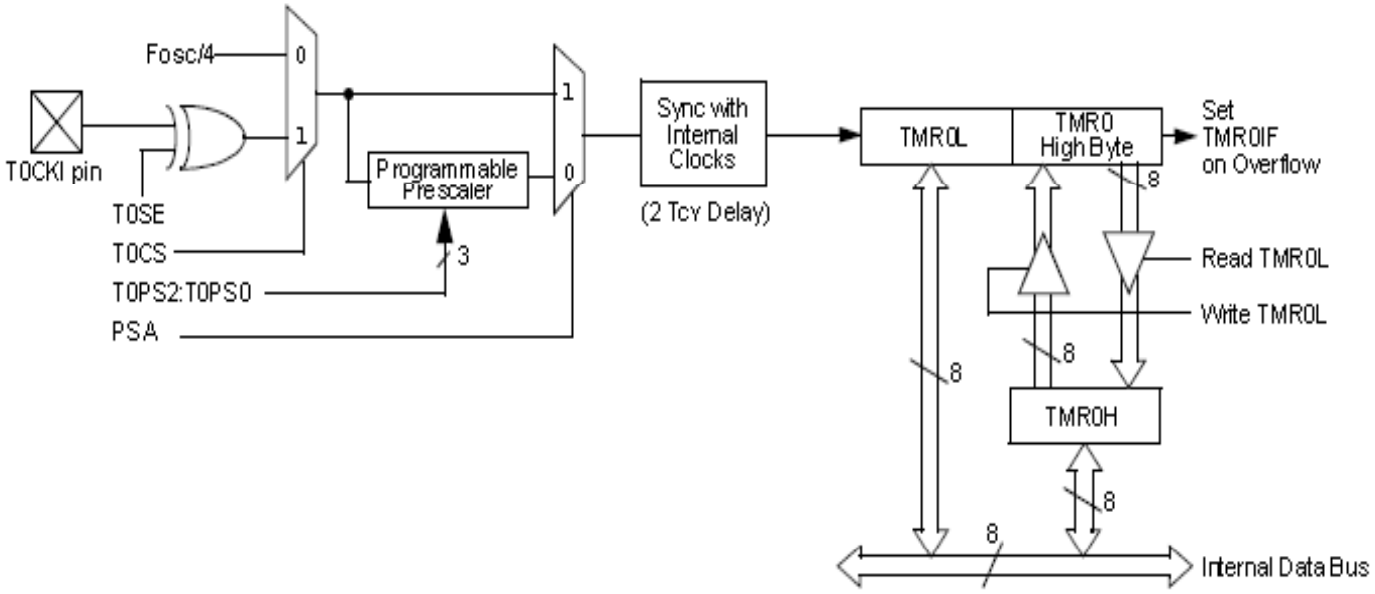
TIMER0 BLOCK DIAGRAM (8-BIT MODE)

4



Note: Upon Reset, Timer0 is enabled in 8-bit mode with clock input from T0CKI maximum prescale.

TIMER0 BLOCK DIAGRAM (16-BIT MODE)



Note: Upon Reset, Timer0 is enabled in 8-bit mode with clock input from T0CKI maximum prescale.

Steps to program Timer0 (16-bit mode)

1. Configure the T0CON register indicating which mode (8-bit or 16- bit) to be used and the selected prescaler option.
2. Load register TMR0H followed by register TMR0L with initial count values.
3. Start the timer with the instruction “T0CONbits.TMR0ON = 1”
4. Keep monitoring the timer flag (T0CONbits.TMR0IF) to see if it is raised. Get out of the loop when TMR0IF becomes high.
5. Stop the timer with the instruction “T0CONbits.TMR0ON = 0”.
6. Clear the TMR0IF flag for the next round (T0CONbits.TMR0IF = 0).
7. Go back to Step 2 to load TMR0H and TMR0L again.

Values for TMR0H and TMR0L

1. Divide the desired time delay by 0.4 μs.
2. Perform *65,536 — n*, where *n* is the decimal value we got in Step 1.
3. Convert the result of Step 2 to hex, where *yyxx* is the initial hex value to be loaded into the timer's registers.
4. Set TMR0H = *yy* and TMR0L = *xx*.

Steps to program Timer0 (8-bit mode)

1. Load the T0CON value register indicating 8-bit mode is selected.
2. Load the TMR0L registers with the initial count value.
3. Start the timer.
4. Keep monitoring the TMR0IF to see if it is raised. Get out of the loop when TMR0IF becomes HIGH.
5. Clear the TMR0IF flag for the next round.
6. Start the timer.
7. Go back to Step 2 to load TMR0L again.

Notice that when we choose the 8-bit option, only the TMR0L register is used and the TMR0H has a zero value during the count up.

Example 1

Write a C18 program to toggle all the bits of PORTB continuously with some delay. Use Timer0, 16-bit mode, and no prescaler options to generate the delay.

```
#include <p18f4550.h>
void T0Delay(void);
void main(void)
{
TRISB=0; //PORTB output port
while (1)   //repeat forever
    {
        PORTB=0x55;     //toggle all bits of Port B
        T0Delay();      //delay size unknown
        PORTB 0xAA; //toggle all bits of Port B
        T0Delay();
    }
}
void T0Delay()
{
    T0CON=0x08; //Timer0, 16-bit mode, no prescaler
    TMR0H=0x35; //load TH0
    TMR0L=0x00; //load TL0
    T0CONbits.TMR0ON = 1; //turn on T0
    while (INTCONbits.TMR0IF == 0); //wait for TF0 to roll over
    T0CONbits TMR0ON = 0; //turn off T0
    INTCONbits.TMR0IF = 0; //clear TF0
}
```

Example 2

Write a C18 program to toggle only the PORTB.4 bit continuously every 50 ms. Use Timer0, 16-bit mode, the 1:4 prescaler to create the delay. Assume XTAL = 10 MHz.

FFFFh - 85EEH = 7A11H
= 31249 + 1 = *31250*

Timer delay =
31250 x 4 x 0.4 ts = 50 ms

```
#include <p18f4550.h>
void T0Delay(void);
#define  mybit     PORTBbits.RB4
void main(void)
{
TRISBbits.TRISB4 = 0;
while (1)   //repeat forever
    {
        mybit ^ = 1;     //toggle using ex-or (^)
        T0Delay();
    }
}
void T0Delay()
{
    T0CON=0x01;     //Timer0, 16-bit mode, 1:4 prescaler
    TMR0H=0x85;     //load TH0
    TMR0L=0xEE;     //load TL0
    T0CONbits.TMR0ON = 1;   //turn on T0
    while (INTCONbits.TMR0IF == 0);    //wait for TF0 to roll over
    T0CONbits TMR0ON = 0;    //turn off T0
    INTCONbits.TMR0IF = 0;    //clear TF0
}
```

Example 3 (Counter)

Assume that a 1-Hz external clock is being fed into pin T0CKI (RA4). Write a C18 program for Counter0 in 8-bit mode to count up and display the state of the TMR0L count on PORTB. Start the count at 0H,

```
#include <p18f4550.h>
void main(void)
{
TRISAbits.TRISA4 = 1; //make T0CKI as input
TRISB = 0;
T0CON = 0x68; //Counter 0, 8-bit mode, no prescaler
TMR0L = 0; //set counter to 0
while (1)   //repeat forever
   {
      do
      {
         T0CONbits.TMR0ON = 1;   //Turn on Timer0
         PORTB = TMR0L;
      {
      while (INTCONbits.TMR0IF == 0); //wait TF0 overflow
      T0CONbits.TMR0ON = 0;   //Turn off Timer0
      INTCONbits.TMR0IF = 0; //Clear TF0
   }
}
```

Timer1

The Timer1 timer/counter module incorporates these features:

- Software selectable operation as a 16-bit timer or counter
- Readable and writable 8-bit registers (TMR1H and TMR1L)
- Selectable clock source (internal or external) with device clock or Timer1 oscillator internal options
- Interrupt on overflow
- Module Reset on CCP Special Event Trigger
- Device clock status flag (T1RUN)

Timer1

REGISTER 12-1: T1CON: TIMER1 CONTROL REGISTER

R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **RD16:** 16-Bit Read/Write Mode Enable bit
1 = Enables register read/write of Timer1 in one 16-bit operation
0 = Enables register read/write of Timer1 in two 8-bit operations

bit 6 **T1RUN:** Timer1 System Clock Status bit
1 = Device clock is derived from Timer1 oscillator
0 = Device clock is derived from another source

bit 5-4 **T1CKPS1:T1CKPS0:** Timer1 Input Clock Prescale Select bits
11 = 1:8 Prescale value
10 = 1:4 Prescale value
01 = 1:2 Prescale value
00 = 1:1 Prescale value

bit 3 **T1OSCEN:** Timer1 Oscillator Enable bit
1 = Timer1 oscillator is enabled
0 = Timer1 oscillator is shut off
The oscillator inverter and feedback resistor are turned off to eliminate power drain.

bit 2 **$\overline{\text{T1SYNC}}$:** Timer1 External Clock Input Synchronization Select bit
When TMR1CS = 1:
1 = Do not synchronize external clock input
0 = Synchronize external clock input
When TMR1CS = 0:
This bit is ignored. Timer1 uses the internal clock when TMR1CS = 0.

bit 1 **TMR1CS:** Timer1 Clock Source Select bit
1 = External clock from RC0/T1OSO/T13CKI pin (on the rising edge)
0 = Internal clock (Fosc/4)

bit 0 **TMR1ON:** Timer1 On bit
1 = Enables Timer1
0 = Stops Timer1

Timer2

The Timer2 module timer incorporates the following features:

- 8-bit timer and period registers (TMR2 and PR2, respectively).
- Readable and writable (both registers).
- Software programmable prescaler (1:1, 1:4 and1:16).
- Software programmable postscaler (1:1 through 1:16).
- Interrupt on TMR2 to PR2 match.
- Optional use as the shift clock for the MSSP module.

Timer2

REGISTER 13-1: T2CON: TIMER2 CONTROL REGISTER

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **Unimplemented:** Read as 'o'

bit 6-3 **T2OUTPS3:T2OUTPS0**: Timer2 Output Postscale Select bits
oooo = 1:1 Postscale
oool = 1:2 Postscale
•
•
•
1111 = 1:16 Postscale

bit 2 **TMR2ON**: Timer2 On bit
1 = Timer2 is on
o = Timer2 is off

bit 1-0 **T2CKPS1:T2CKPS0**: Timer2 Clock Prescale Select bits
oo = Prescaler is 1
o1 = Prescaler is 4
1x = Prescaler is 16

Timer3

The Timer3 module timer/counter incorporates these features:

- Software selectable operation as a 16-bit timer or counter
- Readable and writable 8-bit registers (TMR3H and TMR3L)
- Selectable clock source (internal or external) with device clock or Timer1 oscillator internal options
- Interrupt on overflow
- Module Reset on CCP Special Event Trigger

Timer3

REGISTER 14-1: T3CON: TIMER3 CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{\text{T3SYNC}}$	TMR3CS	TMR3ON
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **RD16:** 16-Bit Read/Write Mode Enable bit
1 = Enables register read/write of Timer3 in one 16-bit operation
0 = Enables register read/write of Timer3 in two 8-bit operations

bit 6, 3 **T3CCP2:T3CCP1:** Timer3 and Timer1 to CCPx Enable bits
1x = Timer3 is the capture/compare clock source for both CCP modules
01 = Timer3 is the capture/compare clock source for CCP2;
Timer1 is the capture/compare clock source for CCP1
00 = Timer1 is the capture/compare clock source for both CCP modules

bit 5-4 **T3CKPS1:T3CKPS0:** Timer3 Input Clock Prescale Select bits
11 = 1:8 Prescale value
10 = 1:4 Prescale value
01 = 1:2 Prescale value
00 = 1:1 Prescale value

bit 2 **$\overline{\text{T3SYNC}}$:** Timer3 External Clock Input Synchronization Control bit
(Not usable if the device clock comes from Timer1/Timer3.)
When TMR3CS = 1:
1 = Do not synchronize external clock input
0 = Synchronize external clock input
When TMR3CS = 0:
This bit is ignored. Timer3 uses the internal clock when TMR3CS = 0.

bit 1 **TMR3CS:** Timer3 Clock Source Select bit
1 = External clock input from Timer1 oscillator or T13CKI (on the rising edge after the first falling edge)
0 = Internal clock (Fosc/4)

bit 0 **TMR3ON:** Timer3 On bit
1 = Enables Timer3
0 = Stops Timer3

C18 Function Library for Timers

Function	Description
CloseTimer*x*	Disable timer *x*.
OpenTimer*x*	Configure and enable timer *x*.
ReadTimer*x*	Read the value of timer *x*.
WriteTimer*x*	Write a value into timer *x*.

OpenTimer0

Arguments: *config*

Enable Timer0 Interrupt:

TIMER_INT_ON	Interrupt enabled
TIMER_INT_OFF	Interrupt disabled

Timer Width:

T0_8BIT	8-bit mode
T0_16BIT	16-bit mode

Clock Source:

T0_SOURCE_EXT	External clock source (I/O pin)
T0_SOURCE_INT	Internal clock source (TOSC)

External Clock Trigger (for T0_SOURCE_EXT):

T0_EDGE_FALL	External clock on falling edge
T0_EDGE_RISE	External clock on rising edge

Prescale Value:

T0_PS_1_1	1:1 prescale
T0_PS_1_2	1:2 prescale
T0_PS_1_4	1:4 prescale
T0_PS_1_8	1:8 prescale
T0_PS_1_16	1:16 prescale
T0_PS_1_32	1:32 prescale
T0_PS_1_64	1:64 prescale
T0_PS_1_128	1:128 prescale
T0_PS_1_256	1:256 prescale

Code Example:

```
OpenTimer0( TIMER_INT_OFF &
            T0_8BIT &
            T0_SOURCE_INT &
            T0_PS_1_32 );
```

Example 4

Write a C18 program using Timer Function Library to toggle all the bits of PORTB continuously with some delay. Use Timer0, 16-bit mode, and no prescaler options to generate the delay.

```
#include <p18f4550.h>
#include <timersh>

void main(void)
{
TRISB=0; //PORTB output port
while (1)   //repeat forever
   {
      PORTB=0x55;     //toggle all bits of Port B
      T0Delay();      //delay size unknown
      PORTB 0xAA; //toggle all bits of Port B
      T0Delay();
   }
}
void T0Delay()
{   // configure timer0
  OpenTimer0( TIMER_INT_OFF & T0_SOURCE_INT &
   T0_PS_1_32 );
    WriteTimer0( 0x3500);
    while (INTCONbits.TMR0IF == 0); //wait for TF0 to roll over
    CloseTimer0();
}
```