

---

---

## Section 11. Timers

---

---

### HIGHLIGHTS

This section of the manual contains the following major topics:

11.1	Introduction .....	11-2
11.2	Timer Variants .....	11-3
11.3	Control Registers .....	11-6
11.4	Modes of Operation .....	11-10
11.5	Timer Interrupts .....	11-15
11.6	32-bit Timer Configuration.....	11-15
11.7	32-bit Timer Modes of Operation .....	11-18
11.8	Timer Operation in Power-Saving Modes .....	11-20
11.9	Peripherals Using Timer Modules .....	11-21
11.10	Register Map.....	11-22
11.11	Related Application Notes.....	11-23
11.12	Revision History .....	11-24

**Note:** This family reference manual section is meant to serve as a complement to device data sheets. Depending on the device variant, this manual section may not apply to all dsPIC33F/PIC24H devices.

Please consult the note at the beginning of the “**Timers**” chapter in the current device data sheet to check whether this document supports the device you are using.

Device data sheets and family reference manual sections are available for download from the Microchip Worldwide Web site at: <http://www.microchip.com>.

## 11.1 INTRODUCTION

The dsPIC33F/PIC24H device family offers several 16-bit Timer modules. With certain exceptions, all 16-bit timers have the same functional circuitry, and are classified into three types according to their functional differences:

- Type A timer (Timer1)
- Type B timer (Timer2, Timer4, Timer6 and Timer8), which is referred to as Timer<sub>x</sub>
- Type C timer (Timer3, Timer5, Timer7 and Timer9), which is referred to as Timer<sub>y</sub>

The Type B and Type C timers can be combined to form a 32-bit timer.

Each Timer module is a 16-bit timer/counter consisting of the following readable/writable registers:

- TMR<sub>n</sub>: 16-bit Timer Count register
- PR<sub>n</sub>: 16-bit Timer Period register associated with the timer
- TnCON: 16-bit Timer Control register associated with the timer

Each Timer module also has these associated bits for interrupt control:

- Interrupt Enable Control bit (TnIE)
- Interrupt Flag Status bit (TnIF)
- Interrupt Priority Control bits (TnIP<2:0>)

**Note 1:** Each dsPIC33F/PIC24H device variant can have one or more Timer modules. For more details, refer to the specific device data sheets.

**2:** An ‘n’ used in the names of pins, control/status bits and registers denotes all timers (n = 1 through 9).

**3:** An ‘x’ used in the names of pins, control/status bits and registers denotes the particular Type B timer number (x = 2, 4, 6 and 8).

**4:** A ‘y’ used in the names of pins, control/status bits and registers denotes the particular Type C timer number (y = 3, 5, 7 and 9).

## 11.2 TIMER VARIANTS

This section describes the different types of timers available on the dsPIC33F/PIC24H family of devices.

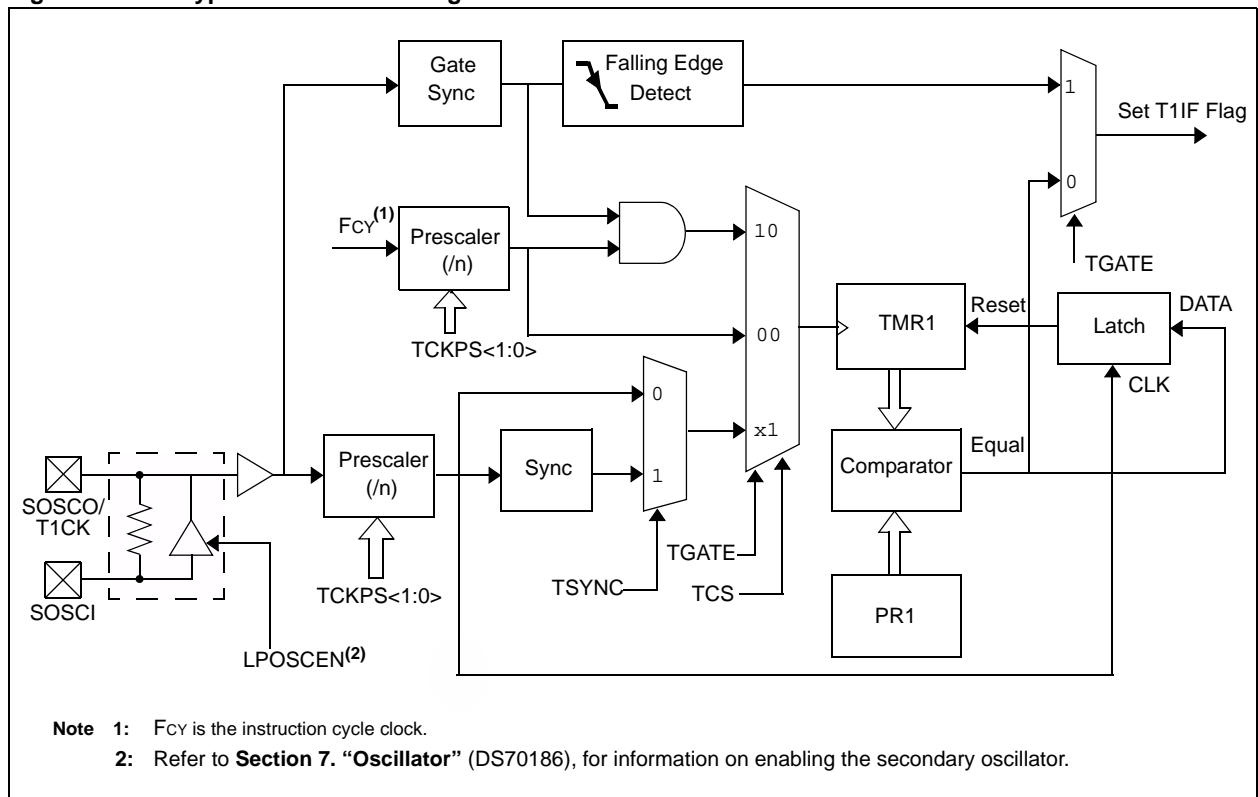
### 11.2.1 Type A Timer

Timer1 is a Type A timer. A Type A timer has the following unique features over other types of timers:

- Can be operated from the low-power 32 kHz crystal oscillator available on the device
- Can be operated in Asynchronous Counter mode from an external clock source
- Optionally, the external clock input (T1CK) can be synchronized to the internal device clock and clock synchronization is performed after T1CK is divided by the prescaler. The advantage of clock synchronization after division by the prescaler is explained in [11.4.3 “Synchronous Counter Mode”](#).

The unique features of Type A timer allows it to be used for Real-Time Clock (RTC) applications. [Figure 11-1](#) shows the block diagram of a Type A timer.

**Figure 11-1: Type A Timer Block Diagram**



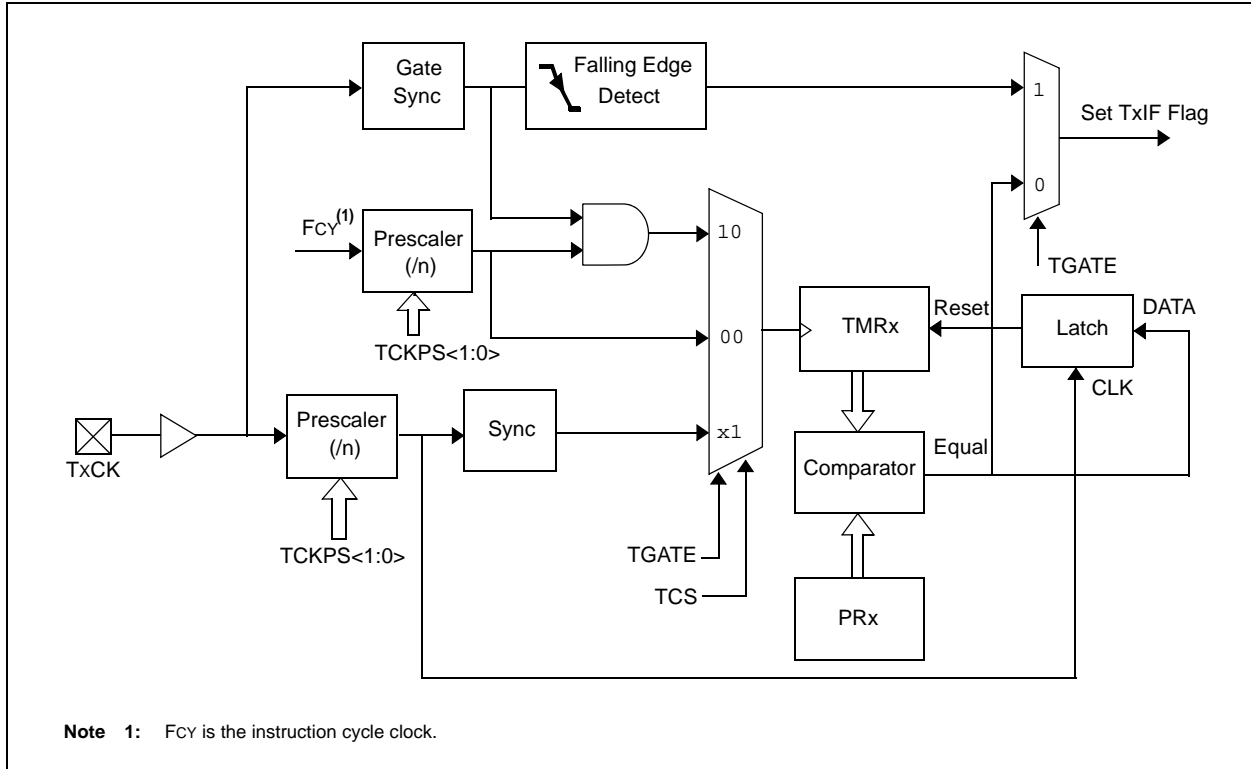
## 11.2.2 Type B Timer

Timer2, Timer4, Timer6 and Timer8, if present, are Type B timers. A Type B timer consists of the following specific features:

- It can be concatenated with a Type C timer to form a 32-bit timer
- The external clock input (TxCK) is always synchronized to the internal device clock and clock synchronization is performed after TxCK is divided by the prescaler. The advantage of clock synchronization after division by the prescaler is explained in [11.4.3 “Synchronous Counter Mode”](#).

Figure 11-2 shows a block diagram of the Type B timer.

**Figure 11-2: Type B Timer Block Diagram**



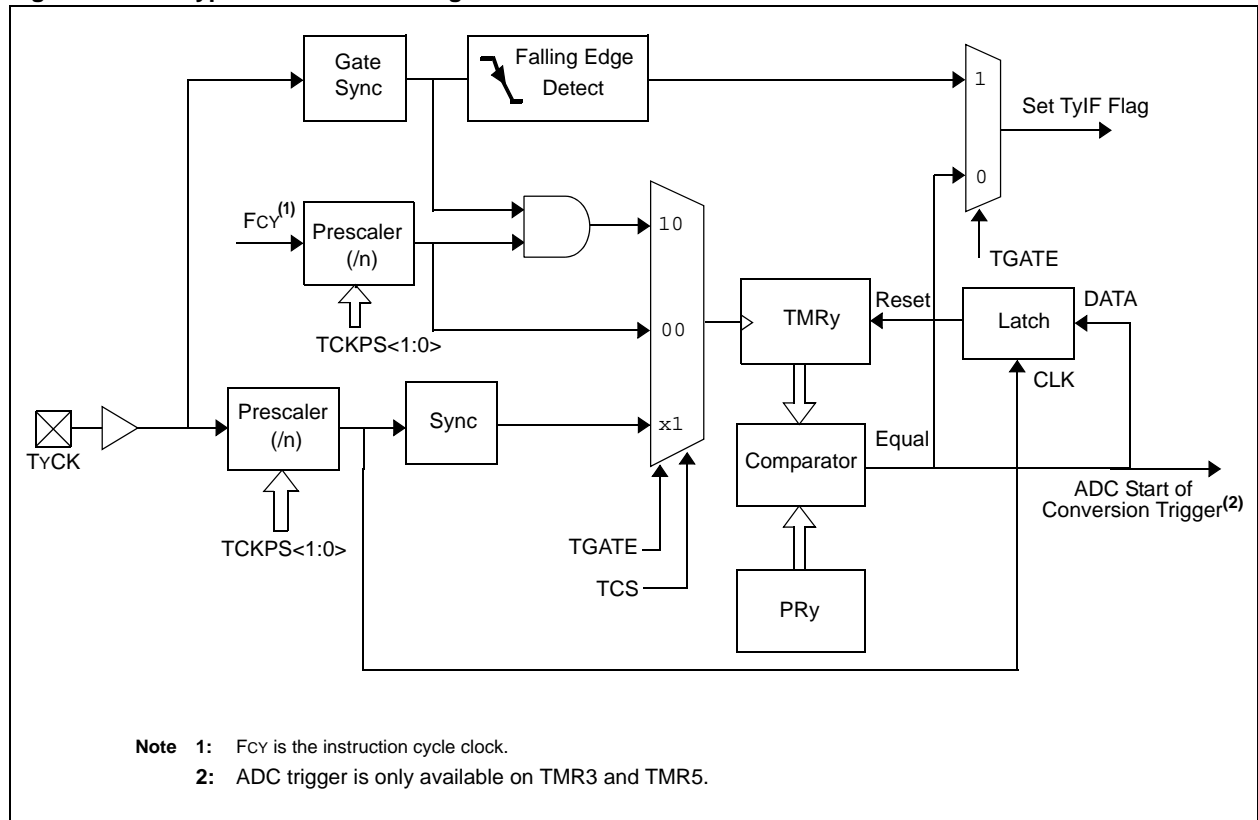
### 11.2.3 Type C Timer

Timer3, Timer5, Timer7 and Timer9, if present, are Type C timers. A Type C timer has the following specific features:

- It can be concatenated with a Type B timer to form a 32-bit timer
- At least one Type C timer has the ability to trigger an analog-to-digital conversion
- The external clock input (TyCK) is always synchronized to the internal device clock. The clock synchronization is performed using TyCK, after which this synchronized clock is divided by the prescaler.

Figure 11-3 shows a block diagram of the Type C timer.

Figure 11-3: Type C Timer Block Diagram



## 11.3 CONTROL REGISTERS

This section outlines the functions of specific timer control registers:

- **T1CON: Type A Timer Control Register**

This register controls the configuration of a Type A timer.

- **TxCON: Type B Timer Control Register (x = 2, 4, 6, 8)**

This register controls the configuration of a Type B timer.

- **TyCON: Type C Timer Control Register (y = 3, 5, 7, 9)**

This register controls the configuration of a Type C timer.

In addition to the above registers, each timer has the following associated 16-bit registers:

- PRn: Timer Period Register (n = 1 through 9)

This is the 16-bit timer period register.

- TMRn: Timer Count Register (n = 1 through 9)

This is the 16-bit timer count register.

Register 11-1: T1CON: Type A Timer Control Register

R/W-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
TON	—	TSIDL	—	—	—	—	—
bit 15							bit 8
U-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	U-0
—	TGATE	TCKPS<1:0>		—	TSYNC	TCS	—
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 15      **TON:** Timer On bit  
1 = Starts the timer  
0 = Stops the timer
- bit 14      **Unimplemented:** Read as '0'
- bit 13      **TSIDL:** Stop in Idle Mode bit  
1 = Discontinue timer operation when device enters Idle mode  
0 = Continue timer operation in Idle mode
- bit 12-7    **Unimplemented:** Read as '0'
- bit 6      **TGATE:** Timer Gated Time Accumulation Enable bit  
When TCS = 1:  
This bit is ignored  
When TCS = 0:  
1 = Gated time accumulation enabled  
0 = Gated time accumulation disabled
- bit 5-4    **TCKPS<1:0>:** Timer Input Clock Prescale Select bits  
11 = 1:256 prescale value  
10 = 1:64 prescale value  
01 = 1:8 prescale value  
00 = 1:1 prescale value
- bit 3      **Unimplemented:** Read as '0'
- bit 2      **TSYNC:** Timer External Clock Input Synchronization Select bit  
When TCS = 1:  
1 = Synchronize external clock input  
0 = Do not synchronize external clock input  
When TCS = 0:  
This bit is ignored. Read as '0'. Timerx uses the internal clock when TCS = 0
- bit 1      **TCS:** Timer Clock Source Select bit  
1 = External clock from TxCK pin  
0 = Internal clock (FOSC/2)
- bit 0      **Unimplemented:** Read as '0'

# dsPIC33F/PIC24H Family Reference Manual

## Register 11-2: TxCON: Type B Timer Control Register (x = 2, 4, 6, 8)

R/W-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
TON	—	TSIDL	—	—	—	—	—
bit 15						bit 8	

U-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	U-0
—	TGATE	TCKPS<1:0>		T32	—	TCS	—
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 15      **TON:** Timerx On bit  
When T32 = 1 (in 32-bit Timer mode):  
 1 = Starts 32-bit TMRx:TMRy<sup>(1)</sup> timer pair  
 0 = Stops 32-bit TMRx:TMRy<sup>(1)</sup> timer pair  
When T32 = 0 (in 16-bit Timer mode):  
 1 = Starts 16-bit timer  
 0 = Stops 16-bit timer
- bit 14      **Unimplemented:** Read as '0'
- bit 13      **TSIDL:** Stop in Idle Mode bit  
 1 = Discontinue timer operation when device enters Idle mode  
 0 = Continue timer operation in Idle mode
- bit 12-7    **Unimplemented:** Read as '0'
- bit 6        **TGATE:** Timerx Gated Time Accumulation Enable bit  
When TCS = 1:  
 This bit is ignored  
When TCS = 0:  
 1 = Gated time accumulation enabled  
 0 = Gated time accumulation disabled
- bit 5-4      **TCKPS<1:0>:** Timerx Input Clock Prescale Select bits  
 11 = 1:256 prescale value  
 10 = 1:64 prescale value  
 01 = 1:8 prescale value  
 00 = 1:1 prescale value
- bit 3        **T32:** 32-bit Timerx Mode Select bit  
 1 = TMRx and TMRy<sup>(1)</sup> form a 32-bit timer  
 0 = TMRx and TMRy<sup>(1)</sup> form separate 16-bit timers
- bit 2        **Unimplemented:** Read as '0'
- bit 1        **TCS:** Timerx Clock Source Select bit  
 1 = External clock from TxCK pin  
 0 = Internal clock (Fosc/2)
- bit 0        **Unimplemented:** Read as '0'

**Note 1:** TMRy is a Type C timer (y = 3, 5, 7 and 9).



Register 11-3: TyCON: Type C Timer Control Register (y = 3, 5, 7, 9)

R/W-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
TON <sup>(2)</sup>	—	TSIDL <sup>(1)</sup>	—	—	—	—	—
bit 15							bit 8
U-0	R/W-0	R/W-0	R/W-0	U-0	U-0	R/W-0	U-0
—	TGATE <sup>(2)</sup>	TCKPS<1:0> <sup>(2)</sup>		—	—	TCS <sup>(2)</sup>	—
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 15      **TON:** Timery On bit<sup>(2)</sup>  
             1 = Starts 16-bit Timery  
             0 = Stops 16-bit Timery
- bit 14      **Unimplemented:** Read as '0'
- bit 13      **TSIDL:** Stop in Idle Mode bit<sup>(1)</sup>  
             1 = Discontinue timer operation when device enters Idle mode  
             0 = Continue timer operation in Idle mode
- bit 12-7    **Unimplemented:** Read as '0'
- bit 6        **TGATE:** Timery Gated Time Accumulation Enable bit<sup>(2)</sup>  
             When TCS = 1:  
             This bit is ignored  
             When TCS = 0:  
             1 = Gated time accumulation enabled  
             0 = Gated time accumulation disabled
- bit 5-4     **TCKPS<1:0>:** Timery Input Clock Prescale Select bits<sup>(2)</sup>  
             11 = 1:256 prescale value  
             10 = 1:64 prescale value  
             01 = 1:8 prescale value  
             00 = 1:1 prescale value
- bit 3-2     **Unimplemented:** Read as '0'
- bit 1        **TCS:** Timery Clock Source Select bit<sup>(2)</sup>  
             1 = External clock from TyCK pin  
             0 = Internal clock (Fosc/2)
- bit 0        **Unimplemented:** Read as '0'

**Note 1:** When 32-bit timer operation is enabled (T32 = 1) in the Type B Timer Control register (TxCON<3>), the TSIDL bit must be cleared to operate the 32-bit timer in Idle mode.

**2:** These bits have no effect when the 32-bit timer operation is enabled (T32 = 1) in the Type B Timer Control register (TxCON<3>).

## 11.4 MODES OF OPERATION

The Timer module can operate in one of the following modes:

- Timer mode
- Gated Timer mode
- Synchronous Counter mode
- Asynchronous Counter mode (Type A timer only)

In Timer and Gated Timer modes, the input clock is derived from the internal instruction cycle clock (FCY). In Synchronous and Asynchronous Counter modes, the input clock is derived from the external clock input at the TxCK pin.

The Timer modes are determined by the following bits:

- TCS (TnCON<1>): Timer Clock Source Control bit
- TSYNC (T1CON<2>): Timer Synchronization Control bit (Type A timer only)
- TGATE (TnCON<6>): Timer Gate Control bit

Timer control bit settings for different operating modes are provided in [Table 11-1](#), as follows:

**Table 11-1: Timer Modes Configuration**

Mode	Bit Setting		
	TCS	TGATE <sup>(2)</sup>	TSYNC <sup>(1)</sup>
Timer	0	0	x
Gated Timer	0	1	x
Synchronous Counter	1	x	1
Asynchronous Counter <sup>(3)</sup>	1	x	0

**Note 1:** TSYNC bit is available for the Type A timer only and is ignored for both of the timer modes.

**2:** TGATE bit is ignored for both of the counter modes.

**3:** Asynchronous Counter mode is supported by the Type A timer only.

The input clock (FCY or TnCK) to all 16-bit timers has prescale options of 1:1, 1:8, 1:64 and 1:256. The clock prescaler is selected using the Timer Clock Prescaler bits (TCKPS) in the Timer Control register (TnCON<5:4>). The prescaler counter is cleared when any of the following occurs:

- A write to the Timer register (TMRn) or Timer Control register (TnCON)
- Clearing the Timer Enable bit (TON) in the Timer Control register (TnCON<15>)
- Any device Reset

The Timer module is enabled or disabled using the TON bit (TnCON<15>).

### 11.4.1 Timer Mode

In Timer mode, the input clock to the timer is derived from the internal clock (FCY), divided by a programmable prescaler. When the timer is enabled, it increments by one on every rising edge of the input clock and generates an interrupt on a period match. [Figure 11-4](#) illustrates the timer operation.

To configure Timer mode:

- Clear the TCS control bit (TnCON<1>) to select the internal clock source
- Clear the TGATE control bit (TnCON<6>) to disable Gated Timer mode operation

Setting the TSYNC bit (TnCON<2>) has no effect since the internal clock is always synchronized.

**Note 1:** The PRn register resets one timer clock period only after the TnIF bit is set.

**2:** The TnIF bit is set one instruction cycle after a period match.

[Example 11-1](#) illustrates the code sequence to set up Timer1 in 16-bit Timer mode. This code generates an interrupt on every 10 instruction cycles.

**Example 11-1: Initialization Code for 16-bit Timer Mode**

```

TlCONbits.TON = 0;           // Disable Timer
TlCONbits.TCS = 0;          // Select internal instruction cycle clock
TlCONbits.TGATE = 0;        // Disable Gated Timer mode
TlCONbits.TCKPS = 0b00;     // Select 1:1 Prescaler
TMR1 = 0x00;                // Clear timer register
PR1 = 9;                    // Load the period value

IPC0bits.T1IP = 0x01;       // Set Timer1 Interrupt Priority Level
IFS0bits.T1IF = 0;         // Clear Timer1 Interrupt Flag
IEC0bits.T1IE = 1;         // Enable Timer1 interrupt

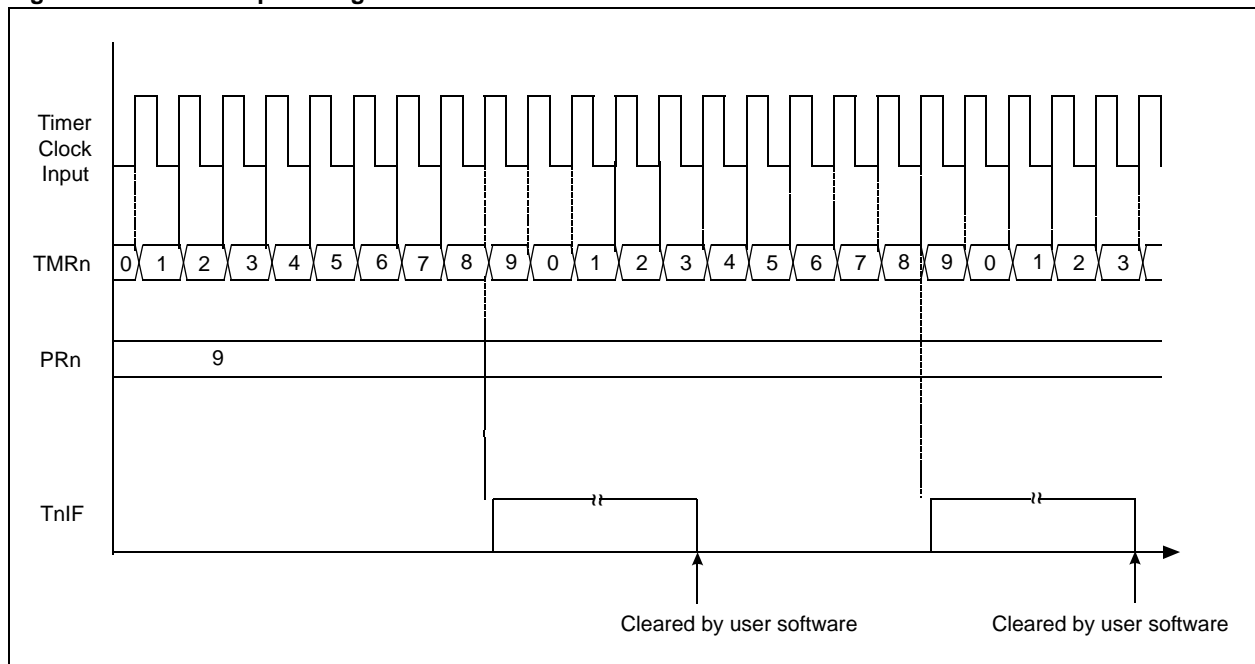
TlCONbits.TON = 1;         // Start Timer

/* Example code for Timer1 ISR */
void __attribute__((__interrupt__, no_auto_psv)) _T1Interrupt(void)
{
    /* Interrupt Service Routine code goes here */

    IFS0bits.T1IF = 0;      // Clear Timer1 Interrupt Flag
}

```

**Note:** The timer counts PRn times for the first TnIF event and (PRn + 1) times for all subsequent TnIF events. For applications in which the asymmetry in interrupt timing is not acceptable, it is recommended to ignore the first TnIF event after enabling the timer.

**Figure 11-4: Interrupt Timing for Timer Period Match**

## 11.4.2 Gated Timer Mode

When the Timer module operates with the internal clock (TCS = 0), Gated Timer mode can be used to measure the duration of an external gate signal. In this mode, the timer increments by one on every rising edge of the input clock as long as the external gate signal at the TnCK pin is high. The timer interrupt is generated on the falling edge of the TnCK pin. Figure 11-5 illustrates Gated Timer mode operation.

To configure the Gated Timer mode:

- Set the TGATE control bit (TnCON<6>) to enable gated timer operation
- Clear the TCS control bit (TnCON<1>) to select the internal clock source

Setting the TSYNC bit (TnCON<2>) has no effect because the internal clock is always synchronized.

Example 11-2 illustrates the code sequence to measure the Timer 1 pulse-width (T1CK) in Gated Timer mode.

### Example 11-2: Initialization Code for 16-bit Gated Timer Mode

```

T1CONbits.TON = 0;           // Disable Timer
T1CONbits.TCS = 0;          // Select internal instruction cycle clock
T1CONbits.TGATE = 1;        // Enable Gated Timer mode
T1CONbits.TCKPS = 0b00;     // Select 1:1 Prescaler
TMR1 = 0x00;                // Clear timer register
PR1 = 9;                    // Load the period value

IPC0bits.T1IP = 0x01;       // Set Timer1 Interrupt Priority Level
IFS0bits.T1IF = 0;         // Clear Timer1 Interrupt Flag
IEC0bits.T1IE = 1;         // Enable Timer1 interrupt

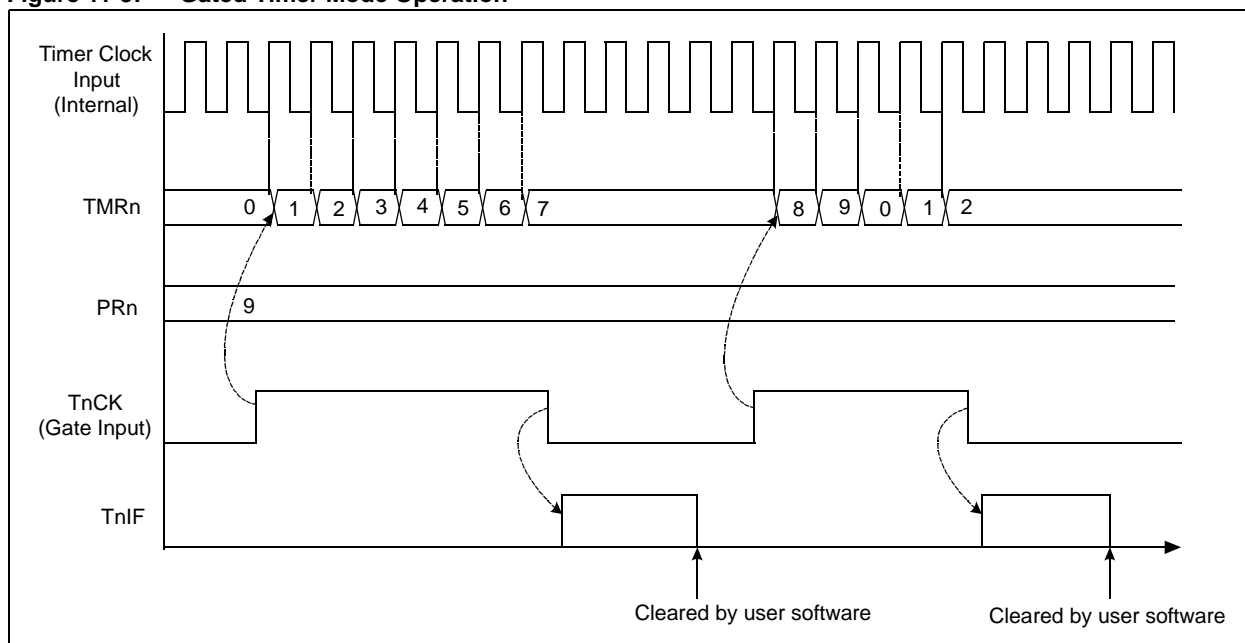
T1CONbits.TON = 1;          // Start Timer

/* Example code for Timer1 ISR */
void __attribute__((__interrupt__, no_auto_psv)) _T1Interrupt(void)
{
    /* Interrupt Service Routine code goes here */

    IFS0bits.T1IF = 0;       // Clear Timer1 Interrupt Flag
}

```

Figure 11-5: Gated Timer Mode Operation



### 11.4.3 Synchronous Counter Mode

In Synchronous Counter mode, the input clock to the timer is derived from the external clock input divided by a programmable prescaler. In this mode, the external clock input is synchronized with the internal device clock. When the timer is enabled, it increments by one on every rising edge of the input clock, and generates an interrupt on a period match.

To configure Synchronous Counter mode:

- Set the TSYNC control bit (T1CON<2>) for a Type A timer to enable clock synchronization. For a Type B or Type C timer, the external clock input is always synchronized.
- Set the TCS control bit (TnCON<1>) to select the external clock source

A timer operating from a synchronized external clock source does not operate in Sleep mode, because the synchronization circuit is shut off during Sleep mode.

For Type C timers when configured for external mode, it is necessary for the external clock input period to meet minimum high and low clock input times for proper synchronization. Refer to the specific device data sheet for timing information.

The clock synchronization for a Type A and Type B timer is performed after the prescaler and the prescaler output changes on the rising edge of the input. Therefore, for a Type A and Type B timer, the external clock input period is a function of T<sub>CY</sub> and an additional input buffer delay divided by the prescaler value. Refer to the specific device data sheet for timing information. In addition to the minimum period, the high and low time of the external clock input must not violate the minimum pulse-width requirements as defined in the data sheet.

- Note 1:** For the external clock timing requirement in Synchronous Counter mode, refer to the “**Electrical Characteristics**” chapter of the specific device data sheet.
- 2: Timers, when configured for the External Counter mode (TCS = 1), operate as follows: Type A and Type B timers start counting from the second rising edge, while Type C timers start counting from the first rising edge.
  - 3: The PR<sub>n</sub> register resets on the subsequent rising edge of the timer clock input.
  - 4: The TnIF bit is set one instruction cycle after a period match.

[Example 11-3](#) illustrates the code sequence to set up the Timer1 module in Synchronous Counter mode. This code generates an interrupt after counting 1000 rising edges in the T1CK pin.

#### Example 11-3: Initialization Code for 16-bit Synchronous Counter Mode

```
T1CONbits.TON = 0;           // Disable Timer
T1CONbits.TCS = 1;           // Select external clock source
T1CONbits.TSYNC = 1;        // Enable Synchronization
T1CONbits.TCKPS = 0b00;     // Select 1:1 Prescaler
TMR1 = 0x00;                // Clear timer register
PR1 = 999;                  // Load the period value

IPC0bits.T1IP = 0x01;       // Set Timer1 Interrupt Priority Level
IFS0bits.T1IF = 0;          // Clear Timer1 Interrupt Flag
IEC0bits.T1IE = 1;         // Enable Timer1 interrupt

T1CONbits.TON = 1;         // Start Timer

/* Example code for Timer1 ISR */
void __attribute__((__interrupt__, no_auto_psv)) _T1Interrupt(void)
{
    /* Interrupt Service Routine code goes here */

    IFS0bits.T1IF = 0;       // Clear Timer1 Interrupt Flag
}
```

- Note:** The timer counts PR<sub>n</sub> times for the first TnIF event and (PR<sub>n</sub> + 1) times for all subsequent TnIF events. For applications in which the asymmetry in interrupt timing is not acceptable, it is recommended to ignore the first TnIF event after enabling the timer.

## 11.4.4 Asynchronous Counter Mode (Type A Timer only)

A Type A timer can operate in an asynchronous counting mode. In Asynchronous Counter mode, the input clock to the timer is derived from the external clock input (T1CK) divided by a programmable prescaler. In this mode, the external clock input is not synchronized with the internal device clock. When enabled, the timer increments by one on every rising edge of the input clock and generates an interrupt on a period match.

To configure the Asynchronous Counter mode:

- Clear the TSYNC control bit (T1CON<2>) to disable clock synchronization
- Set the TCS control bit (T1CON<1>) to select the external clock source

In Asynchronous Counter mode:

- The timer can be clocked from the low-power 32 kHz secondary crystal oscillator for RTC applications by setting the Secondary Oscillator Enable bit (LPOSCEN) in the Oscillator Control register (OSCCON<1>). For more information, refer to **Section 7. “Oscillator”** (DS70186).
- The timer can operate during Sleep mode, if the external clock input is active or the secondary oscillator is enabled. The timer can generate an interrupt (if enabled) on a period register match to wake-up the processor from Sleep mode.
- The high and low time of the external clock input must not violate the minimum pulse-width requirement of 10 ns nominal (or 50 MHz nominal frequency)

**Note 1:** For the external clock timing requirement in Asynchronous Counter mode, refer to the “**Electrical Characteristics**” chapter of the specific device data sheet.

**2:** The PR1 register resets on the subsequent rising edge of the timer clock input.

**3:** The T1IF bit is set one instruction cycle after a period match.

**Example 11-4** illustrates the code sequence to set up the Timer1 module in Asynchronous Counter mode. This code generates an interrupt on every second when running on 32 kHz clock input.

### Example 11-4: Initialization Code for 16-bit Asynchronous Counter Mode

```
T1CONbits.TON = 0;           // Disable Timer
T1CONbits.TCS = 1;           // Select external clock
T1CONbits.TSYNC = 0;         // Disable Synchronization
T1CONbits.TCKPS = 0b00;      // Select 1:1 Prescaler
TMR1 = 0x00;                 // Clear timer register
PR1 = 32767;                 // Load the period value

IPC0bits.T1IP = 0x01;        // Set Timer1 Interrupt Priority Level
IFS0bits.T1IF = 0;           // Clear Timer1 Interrupt Flag
IEC0bits.T1IE = 1;           // Enable Timer1 interrupt

T1CONbits.TON = 1;           // Start Timer

/* Example code for Timer1 ISR */
void __attribute__((__interrupt__, no_auto_psv)) _T1Interrupt(void)
{
    /* Interrupt Service Routine code goes here */

    IFS0bits.T1IF = 0;         // Clear Timer1 Interrupt Flag
}
```

**Note:** The timer counts PR1 times for the first T1IF event and (PR1 + 1) times for all subsequent T1IF events. For applications in which the asymmetry in interrupt timing is not acceptable, it is recommended to ignore the first T1IF event after enabling the timer.

## 11.5 TIMER INTERRUPTS

A timer interrupt is generated:

- On a period match for Timer mode or Synchronous/Asynchronous Counter mode (refer to [Figure 11-4](#))
- On falling edge of the “gate” signal at the TnCK pin for Gated Timer mode (refer to [Figure 11-5](#))

The Timer Interrupt Flag bit (TnIF) must be cleared in software.

A timer is enabled as a source of interrupt via the respective Timer Interrupt Enable bit (TnIE). The Interrupt Priority Level bits (TnIP<2:0>) must be written with a non-zero value for the timer to be a source of interrupt. For more information, refer to **Section 6. “Interrupts”** (DS70184).

**Note:** A special case occurs when the period register, PRn, is loaded with 0x0000 and the timer is enabled. No timer interrupts are generated for this configuration.

## 11.6 32-BIT TIMER CONFIGURATION

A 32-bit Timer module can be formed by combining Type B and Type C 16-bit timers. For 32-bit timer operation, the T32 control bit in the Type B Timer Control register (TxCON<3>) must be set. The Type C timer holds the most significant word (msw) and the Type B timer holds the least significant word (lsw) for 32-bit operation.

When configured for 32-bit operation, only the Type B Timer Control register bits are required for setup and control. With the exception of the TSIDL bit, all Type C Timer Control register bits are ignored. For more information, refer to [11.8.2 “Timer Operation in Idle Mode”](#).

For interrupt control, the combined 32-bit timer uses the interrupt enable, interrupt flag, and interrupt priority control bits of the Type C timer. The interrupt control and status bits of the Type B timer are ignored during 32-bit timer operation.

[Table 11-2](#) lists the Type B and Type C timers that can be combined to form a 32-bit timer.

**Table 11-2: 32-bit Timer Combinations**

Type B Timer (lsw)	Type C Timer (msw)
Timer2	Timer3
Timer4	Timer5
Timer6	Timer7
Timer8	Timer9

A block diagram representation of the 32-bit Timer module is shown in [Figure 11-6](#). The 32-bit Timer module can operate in any of the following modes:

- Timer
- Gated Timer
- Synchronous Counter

In Timer and Gated Timer modes, the input clock is derived from the internal instruction cycle clock (Fcy). In Synchronous Counter mode, the input clock is derived from the Type B timer external clock input at the TxCK pin.

The 32-bit Timer modes are determined by the following bits in the Type B Timer Control registers:

- TCS (TxCON<1>): Timer Clock Source control bit
- TGATE (TxCON<6>): Timer Gate control bit

Timer control bit settings for different operating modes are provided in [Table 11-3](#).

**Table 11-3: Timer Mode Configuration**

Mode	Bit Setting	
	TCS	TGATE
Timer	0	0
Gated Timer	0	1
Synchronous Counter	1	x

**Note 1:** Type B and Type C timers do not support the Asynchronous External Clock mode; therefore, 32-bit Asynchronous Counter mode is not supported.

**2:** The PRx register resets on the subsequent rising edge of the timer clock input.

**3:** The TxIF bit is set one instruction cycle after a period match.

The input clock (FCY or TxCK) to all 32-bit timers has prescale options of 1:1, 1:8, 1:64 and 1:256. The clock prescaler is selected using the Timer Clock Prescaler bits (TCKPS<1:0>) in the Type B Timer Control register (TxCON<5:4>). The prescaler counter is cleared when any of the following occurs:

- A write to the Type B Timer register (TMRx) or Type B Timer Control register (TxCON)
- Clearing the Timer Enable bit (TON) in the Type B Timer Control register (TxCON<15>)
- Any device Reset

The 32-bit Timer module is enabled or disabled using the TON bit in the Type B Timer Control registers (TxCON<15>).

For 32-bit read/write operations to be synchronized between the lsw and msw of the 32-bit timer, additional control logic and holding registers are used (refer to [Figure 11-6](#)). Each Type C timer has a register called TMRyHLD, which is used when reading or writing the timer register pair. The TMRyHLD registers are used only when the respective timers are configured for 32-bit operation.

Assuming TMR3:TMR2 form a 32-bit timer pair, the user application must first read the lsw of the timer value from the TMR2 register. The read of the lsw automatically transfers the contents of TMR3 into the TMR3HLD register. The user application can then read TMR3HLD to get the msw of the timer value.

To write a value to the TMR3:TMR2 register pair, the user application must first write the msw to the TMR3HLD register. When the lsw of the timer value is written to TMR2, the contents of TMR3HLD is automatically transferred to the TMR3 register.

The code for accessing the 32-bit timer is shown in [Example 11-5](#).

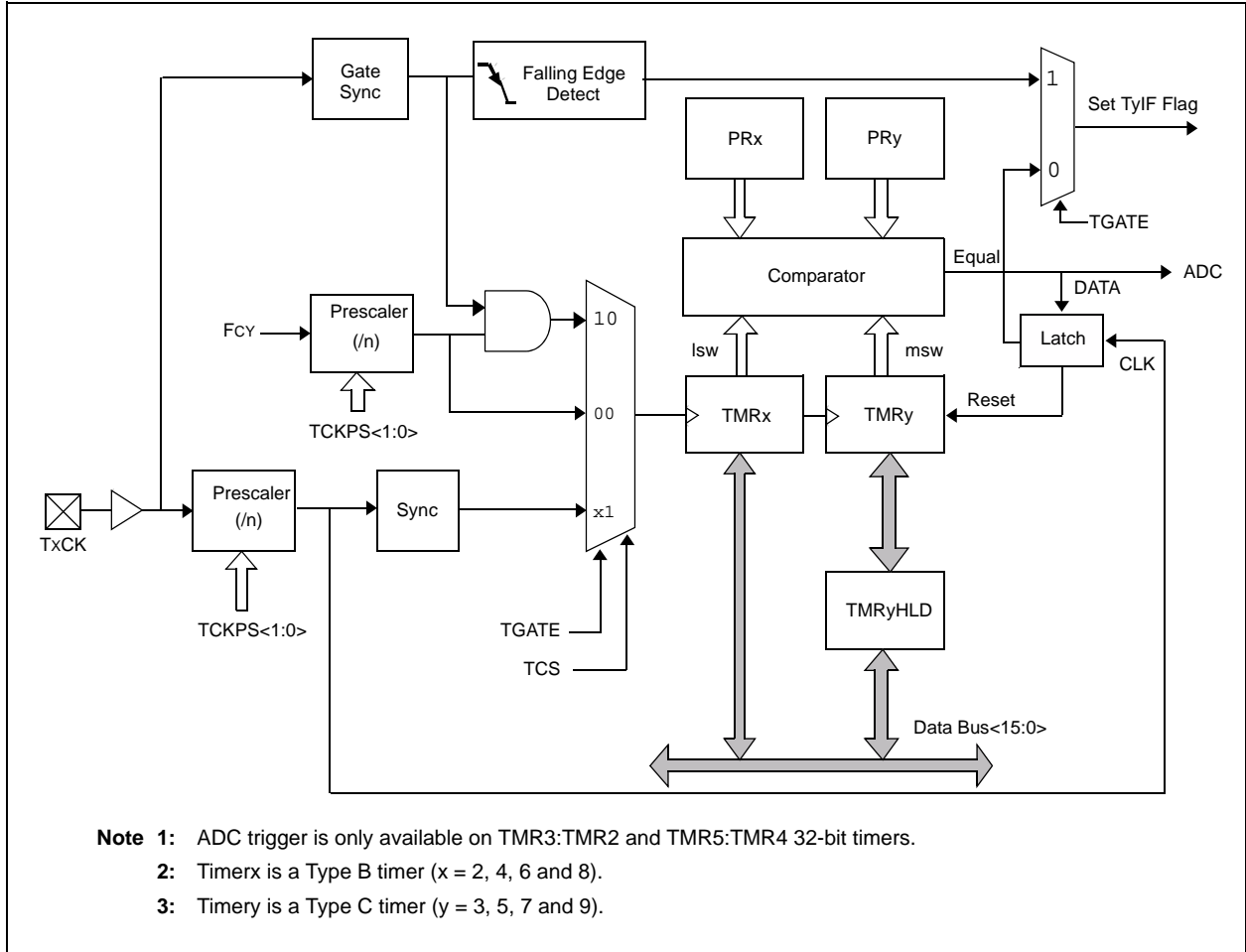
**Example 11-5: 32-bit Timer Access**

```
// Reading from 32-bit timer
lsw = TMR2;      // Read lsw from the Type B timer register
msw = TMR3HLD;  // Read msw from the Type C timer holding register

// Writing to 32-bit timer
TMR3HLD = msw;  // Write msw to the Type C timer holding register
TMR2 = lsw;     // Write lsw to the Type B timer register
```



Figure 11-6: Type B/Type C Timer Pair Block Diagram (32-bit Timer)



## 11.7 32-BIT TIMER MODES OF OPERATION

### 11.7.1 Timer Mode

The 32-bit timer operates similarly to a 16-bit timer in Timer mode. [Example 11-6](#) illustrates the code sequence to set up Timer2 and Timer3 in 32-bit Timer mode.

#### Example 11-6: Initialization Code for 32-bit Timer Mode

```
T3CONbits.TON = 0;           // Stop any 16-bit Timer3 operation
T2CONbits.TON = 0;           // Stop any 16/32-bit Timer2 operation
T2CONbits.T32 = 1;           // Enable 32-bit Timer mode
T2CONbits.TCS = 0;           // Select internal instruction cycle clock
T2CONbits.TGATE = 0;         // Disable Gated Timer mode
T2CONbits.TCKPS = 0b00;      // Select 1:1 Prescaler
TMR3 = 0x00;                 // Clear 32-bit Timer (msw)
TMR2 = 0x00;                 // Clear 32-bit Timer (lsw)
PR3 = 0x0002;                // Load 32-bit period value (msw)
PR2 = 0x0000;                // Load 32-bit period value (lsw)

IPC2bits.T3IP = 0x01;        // Set Timer3 Interrupt Priority Level
IFS0bits.T3IF = 0;           // Clear Timer3 Interrupt Flag
IEC0bits.T3IE = 1;           // Enable Timer3 interrupt

T2CONbits.TON = 1;           // Start 32-bit Timer

/* Example code for Timer3 ISR */
void __attribute__((__interrupt__, no_auto_psv)) _T3Interrupt(void)
{
    /* Interrupt Service Routine code goes here */

    IFS0bits.T3IF = 0;        // Clear Timer3 Interrupt Flag
}
```

### 11.7.2 Gated Timer Mode

The 32-bit timer operates similarly to a 16-bit timer in Gated Timer mode. [Example 11-7](#) illustrates the code sequence to set up Timer2 and Timer3 in 32-bit Gated Timer mode.

#### Example 11-7: Initialization Code for 32-bit Gated Timer Mode

```
T3CONbits.TON = 0;           // Stop any 16-bit Timer3 operation
T2CONbits.TON = 0;           // Stop any 16/32-bit Timer2 operation
T2CONbits.T32 = 1;           // Enable 32-bit Timer mode
T2CONbits.TCS = 0;           // Select internal instruction cycle clock
T2CONbits.TGATE = 1;         // Enable Gated Timer mode
T2CONbits.TCKPS = 0b00;      // Select 1:1 Prescaler
TMR3 = 0x00;                 // Clear 32-bit Timer (msw)
TMR2 = 0x00;                 // Clear 32-bit Timer (lsw)
PR3 = 0x0002;                // Load 32-bit period value (msw)
PR2 = 0x0000;                // Load 32-bit period value (lsw)

IPC2bits.T3IP = 0x01;        // Set Timer3 Interrupt Priority Level
IFS0bits.T3IF = 0;           // Clear Timer3 Interrupt Flag
IEC0bits.T3IE = 1;           // Enable Timer3 interrupt

T2CONbits.TON = 1;           // Start 32-bit Timer

/* Example code for Timer3 ISR */
void __attribute__((__interrupt__, no_auto_psv)) _T3Interrupt(void)
{
    /* Interrupt Service Routine code goes here */

    IFS0bits.T3IF = 0;        // Clear Timer3 Interrupt Flag
}
```

### 11.7.3 Synchronous Counter Mode

The 32-bit timer operates similarly to a 16-bit timer in Synchronous Counter mode. [Example 11-8](#) illustrates the code sequence to set up Timer2 and Timer3 in 32-bit Synchronous Counter mode.

#### Example 11-8: Initialization Code for 32-bit Synchronous Counter Mode

```

T3CONbits.TON = 0;           // Stop any 16-bit Timer3 operation
T2CONbits.TON = 0;           // Stop any 16/32-bit Timer2 operation
T2CONbits.T32 = 1;           // Enable 32-bit Timer mode
T2CONbits.TCS = 1;           // Select External clock
T2CONbits.TCKPS = 0b00;      // Select 1:1 Prescaler
TMR3 = 0x00;                 // Clear 32-bit Timer (msw)
TMR2 = 0x00;                 // Clear 32-bit Timer (lsw)
PR3 = 0x0002;                // Load 32-bit period value (msw)
PR2 = 0x0000;                // Load 32-bit period value (lsw)

IPC2bits.T3IP = 0x01;        // Set Timer3 Interrupt Priority Level
IFS0bits.T3IF = 0;           // Clear Timer3 Interrupt Flag
IEC0bits.T3IE = 1;           // Enable Timer3 interrupt

T2CONbits.TON = 1;           // Start 32-bit Timer

/* Example code for Timer3 ISR */
void __attribute__((__interrupt__, no_auto_psv)) _T3Interrupt(void)
{
    /* Interrupt Service Routine code goes here */

    IFS0bits.T3IF = 0;        // Clear Timer3 Interrupt Flag
}

```

## 11.8 TIMER OPERATION IN POWER-SAVING MODES

### 11.8.1 Timer Operation in Sleep Mode

When the device enters Sleep mode, the system clock is disabled. If the Timer module is running on the internal clock source (FCY), it is disabled as well.

A Type A timer is different from the other timers because it can operate asynchronously from the system clock source. Because of this distinction, the Type A timer can continue to operate during Sleep mode. To operate in Sleep mode, the Type A timer must be configured as follows:

- Clear the TSYNC control bit (T1CON<2>) to disable clock synchronization
- Set the TCS control bit (T1CON<1>) to select external clock source
- Enable the secondary oscillator, if the external clock input (T1CK) is not active

**Note:** The secondary oscillator is enabled by setting the Secondary Oscillator Enable bit (LPOSCEN) in the Oscillator Control register (OSCCON<1>). For more information, refer to **Section 7. “Oscillator”** (DS70186). The 32 kHz watch crystal must be connected to the SOSCO/SOSCI device pins.

When all of these conditions are met, the timer continues to count and detect period matches while the device is in Sleep mode. When a match between the timer and the period register occurs, the TnIF bit is set. The timer interrupt is generated, if the timer interrupt is enabled (TnIE = 1).

The timer interrupt wakes up the device from Sleep, and the following events occur:

- If the assigned priority level for the interrupt is less than or equal to the current CPU priority, the device wakes up and continues code execution from the instruction following the PWRSAV instruction that initiated Sleep mode
- If the assigned priority level for the interrupt source is greater than the current CPU priority, the device wakes up and the CPU exception process begins. Code execution continues from the first instruction of the timer Interrupt Service Routine (ISR)

For more information, refer to **Section 9. “Watchdog Timer and Power-Saving Modes”** (DS70196).

### 11.8.2 Timer Operation in Idle Mode

When the device enters Idle mode, the system clock sources remain functional and the CPU stops executing code. The Timer Stop in Idle bit (TSIDL) in the Timer Control register (TnCON<13>) determines whether the module stops in Idle mode or continues to operate in Idle mode.

- If TSIDL = 0, the timer continues to operate in Idle mode providing full functionality. For 32-bit timer operation, the TSIDL bit (TxCON<13> and TyCON<13>) must be cleared in Type B and Type C Timer Control registers for a timer to operate in Idle mode.
- If TSIDL = 1, the timer performs the same functions when stopped in Idle mode as in Sleep mode (refer to **11.8.1 “Timer Operation in Sleep Mode”**)

## 11.9 PERIPHERALS USING TIMER MODULES

### 11.9.1 Time Base for Input Capture and Output Compare

The input capture and output compare peripherals can select Timer2 or Timer3 as the time base. For more information, refer to **Section 12. “Input Capture”** (DS70198), **Section 13. “Output Compare”** (DS70209), and the specific device data sheet.

### 11.9.2 Analog-to-Digital Special Event Trigger

On each device variant, one Type C timer can generate a special analog-to-digital conversion trigger signal on a period match, in both 16-bit and 32-bit modes. The Timer module provides a conversion start signal to the analog-to-digital sampling logic.

- If  $T32 = 0$ , when a match occurs between the 16-bit timer register (TMRx) and the respective 16-bit period register (PRx), the Analog-to-Digital Special Event Trigger signal is generated
- If  $T32 = 1$ , when a match occurs between the 32-bit timer (TMRx:TMRy) and the 32-bit respective combined period register (PRx:PRy), the Analog-to-Digital Special Event Trigger signal is generated

The Special Event Trigger signal is always generated by the timer. The trigger source must be selected in the Analog-to-Digital Converter (ADC) control registers. For additional information, refer to **Section 16. “Analog-to-Digital Converter (ADC)”** (DS70183) and the specific device data sheet.

### 11.9.3 Timer as an External Interrupt Pin

The external clock input pin for each timer can be used as an additional interrupt pin. To provide the interrupt, the timer period register, PRn, is written with a non-zero value and the TMRn register is initialized to a value of one less than the value written to the period register. The timer must be configured for a 1:1 clock prescaler. An interrupt is generated when the next rising edge of the external clock signal is detected.

### 11.9.4 I/O Pin Control

When a Timer module is enabled and configured for external clock or gate operation, the user application must ensure the I/O pin direction is configured for an input. Enabling the Timer module does not configure the pin direction.

## 11.10 REGISTER MAP

A summary of the Special Function Registers (SFRs) associated with the dsPIC33F/PIC24H Timer module is provided in [Table 11-4](#).

**Table 11-4: Timer Register Map**

SFR Name	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
TMR1	Timer1 Register																xxxx
PR1	Period Register 1																FFFF
T1CON	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS<1:0>	—	TSYNC	TCS	—	0000	
TMRx	Timerx Register																xxxx
TMRyHLD	Timery Holding Register (for 32-bit timer operations only)																xxxx
TMRy	Timery Register																xxxx
PRx	Period Register x																FFFF
PRy	Period Register y																FFFF
TxCON	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS<1:0>	T32	—	TCS	—	0000	
TyCON	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS<1:0>	—	—	TCS	—	0000	

**Legend:** x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

- Note 1:** An 'n' used in the names of pins, control/status bits and registers denotes all timers (n = 1 through 9).  
**Note 2:** An 'x' used in the names of pins, control/status bits and registers denotes the particular Type B timer number (x = 2, 4, 6 and 8).  
**Note 3:** A 'y' used in the names of pins, control/status bits and registers denotes the particular Type C timer number (y = 3, 5, 7 and 9).

## 11.11 RELATED APPLICATION NOTES

This section lists application notes that are related to this section of the manual. These application notes may not be written specifically for the dsPIC33F/PIC24H device family, but the concepts are pertinent and could be used with modification and possible limitations. The current application notes related to the Timers module are:

Title	Application Note #
Using Timer1 in Asynchronous Clock Mode	AN580

**Note:** For additional application notes and code examples for the dsPIC33F/PIC24H device family, visit the Microchip web site ([www.microchip.com](http://www.microchip.com)).

## 11.12 REVISION HISTORY

### Revision A (April 2007)

This is the initial released revision of this document.

### Revision B (April 2008)

This revision incorporates the following updates:

- Notes:
  - Added a note in [11.4.3 “Synchronous Counter Mode”](#), which provides information on timer operation when configured for External Counter mode (TCS = 1)
  - Corrected Note 2 in [Figure 11-6](#). TMR5:TMR2 was changed to TMR5:TMR4
- Additional minor corrections such as language and formatting updates were incorporated throughout the document

### Revision C (January 2010)

This revision incorporates the following updates:

- All references to dsPIC33F in the document were updated to dsPIC33F/PIC24H
- Changed all instances of `__shadow__` to `no_auto_psv` in the code examples
- Added latch block to [Figure 11-1](#), [Figure 11-2](#), [Figure 11-3](#) and [Figure 11-6](#)
- Notes:
  - Added a shaded note at the beginning of the section, which provides information on complimentary documentation
  - Added the following note in [11.4.1 “Timer Mode”](#):
    - The PRx register resets one timer clock period only after the TxIF bit is set
  - Added the following note in [11.4.3 “Synchronous Counter Mode”](#) and [11.4.4 “Asynchronous Counter Mode \(Type A Timer only\)”](#):
    - The PRx register resets on the subsequent rising edge of the timer clock input
  - Added the following notes in [11.4.1 “Timer Mode”](#), [11.4.3 “Synchronous Counter Mode”](#), and [11.4.4 “Asynchronous Counter Mode \(Type A Timer only\)”](#):
    - The TxIF bit is set one instruction cycle after a period match
    - The timer counts PRx times for the first TxIF event and (PRx + 1) times for all subsequent TxIF events. For applications in which the asymmetry in interrupt timing is not acceptable, it is recommended to ignore the first TxIF event after enabling the timer.
- Removed Table 11-5: Interrupt Control Register Map
- Additional minor corrections such as language and formatting updates were incorporated throughout the document

### Revision D (September 2011)

This revision includes the following updates:

- Global updates were made throughout the document to distinguish between the three timer types: A (Timer1), B (Timerx), and C (Timery). In addition, the use of the letter ‘n’ was introduced to indicate pins, control/status bits and registers that are common to all three types of timers.
- Updated the Timer block diagrams (see [Figure 11-1](#), [Figure 11-2](#), and [Figure 11-3](#))
- Updated the third and fourth paragraphs of [11.4.3 “Synchronous Counter Mode”](#)
- Updated the Initialization Code for 32-bit Timer, 32-bit Gated Timer, and 32-bit Synchronous Counter Mode (see [Example 11-6](#), [Example 11-7](#), and [Example 11-8](#))
- Updated the Timer Register Map (see [Table 11-4](#))
- The Preliminary document status was removed
- Minor corrections to text and formatting were incorporated throughout the document



---

---

**Note the following details of the code protection feature on Microchip devices:**

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families 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 Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is 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 products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

---

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights.

**Trademarks**

The Microchip name and logo, the Microchip logo, dsPIC, KEELOQ, KEELOQ logo, MPLAB, PIC, PICmicro, PICSTART, PIC<sup>32</sup> logo, rPIC and UNI/O are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.


FilterLab, Hampshire, HI-TECH C, Linear Active Thermistor, MXDEV, MXLAB, SEEVAL and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Analog-for-the-Digital Age, Application Maestro, chipKIT, chipKIT logo, CodeGuard, dsPICDEM, dsPICDEM.net, dsPICworks, dsSPEAK, ECAN, ECONOMONITOR, FanSense, HI-TIDE, In-Circuit Serial Programming, ICSP, Mindi, MiWi, MPASM, MPLAB Certified logo, MPLIB, MPLINK, mTouch, Omniscent Code Generation, PICC, PICC-18, PICDEM, PICDEM.net, PICkit, PICtail, REAL ICE, rfLAB, Select Mode, Total Endurance, TSHARC, UniWinDriver, WiperLock and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

All other trademarks mentioned herein are property of their respective companies.

© 2007-2011, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

 Printed on recycled paper.

ISBN: 978-1-61341-624-2

*Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC<sup>®</sup> MCUs and dsPIC<sup>®</sup> DSCs, KEELOQ<sup>®</sup> code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.*

**QUALITY MANAGEMENT SYSTEM**  
**CERTIFIED BY DNV**  
**== ISO/TS 16949:2009 ==**



# MICROCHIP

## 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:  
<http://www.microchip.com/support>  
Web Address:  
[www.microchip.com](http://www.microchip.com)

**Atlanta**  
Duluth, GA  
Tel: 678-957-9614  
Fax: 678-957-1455

**Boston**  
Westborough, MA  
Tel: 774-760-0087  
Fax: 774-760-0088

**Chicago**  
Itasca, IL  
Tel: 630-285-0071  
Fax: 630-285-0075

**Cleveland**  
Independence, OH  
Tel: 216-447-0464  
Fax: 216-447-0643

**Dallas**  
Addison, TX  
Tel: 972-818-7423  
Fax: 972-818-2924

**Detroit**  
Farmington Hills, MI  
Tel: 248-538-2250  
Fax: 248-538-2260

**Indianapolis**  
Noblesville, IN  
Tel: 317-773-8323  
Fax: 317-773-5453

**Los Angeles**  
Mission Viejo, CA  
Tel: 949-462-9523  
Fax: 949-462-9608

**Santa Clara**  
Santa Clara, CA  
Tel: 408-961-6444  
Fax: 408-961-6445

**Toronto**  
Mississauga, Ontario,  
Canada  
Tel: 905-673-0699  
Fax: 905-673-6509

### ASIA/PACIFIC

**Asia Pacific Office**  
Suites 3707-14, 37th Floor  
Tower 6, The Gateway  
Harbour City, Kowloon  
Hong Kong  
Tel: 852-2401-1200  
Fax: 852-2401-3431

**Australia - Sydney**  
Tel: 61-2-9868-6733  
Fax: 61-2-9868-6755

**China - Beijing**  
Tel: 86-10-8569-7000  
Fax: 86-10-8528-2104

**China - Chengdu**  
Tel: 86-28-8665-5511  
Fax: 86-28-8665-7889

**China - Chongqing**  
Tel: 86-23-8980-9588  
Fax: 86-23-8980-9500

**China - Hangzhou**  
Tel: 86-571-2819-3187  
Fax: 86-571-2819-3189

**China - Hong Kong SAR**  
Tel: 852-2401-1200  
Fax: 852-2401-3431

**China - Nanjing**  
Tel: 86-25-8473-2460  
Fax: 86-25-8473-2470

**China - Qingdao**  
Tel: 86-532-8502-7355  
Fax: 86-532-8502-7205

**China - Shanghai**  
Tel: 86-21-5407-5533  
Fax: 86-21-5407-5066

**China - Shenyang**  
Tel: 86-24-2334-2829  
Fax: 86-24-2334-2393

**China - Shenzhen**  
Tel: 86-755-8203-2660  
Fax: 86-755-8203-1760

**China - Wuhan**  
Tel: 86-27-5980-5300  
Fax: 86-27-5980-5118

**China - Xian**  
Tel: 86-29-8833-7252  
Fax: 86-29-8833-7256

**China - Xiamen**  
Tel: 86-592-2388138  
Fax: 86-592-2388130

**China - Zhuhai**  
Tel: 86-756-3210040  
Fax: 86-756-3210049

### ASIA/PACIFIC

**India - Bangalore**  
Tel: 91-80-3090-4444  
Fax: 91-80-3090-4123

**India - New Delhi**  
Tel: 91-11-4160-8631  
Fax: 91-11-4160-8632

**India - Pune**  
Tel: 91-20-2566-1512  
Fax: 91-20-2566-1513

**Japan - Yokohama**  
Tel: 81-45-471- 6166  
Fax: 81-45-471-6122

**Korea - Daegu**  
Tel: 82-53-744-4301  
Fax: 82-53-744-4302

**Korea - Seoul**  
Tel: 82-2-554-7200  
Fax: 82-2-558-5932 or  
82-2-558-5934

**Malaysia - Kuala Lumpur**  
Tel: 60-3-6201-9857  
Fax: 60-3-6201-9859

**Malaysia - Penang**  
Tel: 60-4-227-8870  
Fax: 60-4-227-4068

**Philippines - Manila**  
Tel: 63-2-634-9065  
Fax: 63-2-634-9069

**Singapore**  
Tel: 65-6334-8870  
Fax: 65-6334-8850

**Taiwan - Hsin Chu**  
Tel: 886-3-5778-366  
Fax: 886-3-5770-955

**Taiwan - Kaohsiung**  
Tel: 886-7-536-4818  
Fax: 886-7-330-9305

**Taiwan - Taipei**  
Tel: 886-2-2500-6610  
Fax: 886-2-2508-0102

**Thailand - Bangkok**  
Tel: 66-2-694-1351  
Fax: 66-2-694-1350

### EUROPE

**Austria - Wels**  
Tel: 43-7242-2244-39  
Fax: 43-7242-2244-393

**Denmark - Copenhagen**  
Tel: 45-4450-2828  
Fax: 45-4485-2829

**France - Paris**  
Tel: 33-1-69-53-63-20  
Fax: 33-1-69-30-90-79

**Germany - Munich**  
Tel: 49-89-627-144-0  
Fax: 49-89-627-144-44

**Italy - Milan**  
Tel: 39-0331-742611  
Fax: 39-0331-466781

**Netherlands - Druenen**  
Tel: 31-416-690399  
Fax: 31-416-690340

**Spain - Madrid**  
Tel: 34-91-708-08-90  
Fax: 34-91-708-08-91

**UK - Wokingham**  
Tel: 44-118-921-5869  
Fax: 44-118-921-5820

08/02/11