Section 14. Timers

HIGHLIGHTS

This section of the manual contains the following major topics:

14.1   Introduction ............................................................................................................... 14-2
14.2   Timer Variants ............................................................................................................. 14-3
14.3   Control Registers ...................................................................................................... 14-6
14.4   Modes of Operation ................................................................................................. 14-9
14.5   Timer Prescalers ....................................................................................................... 14-14
14.6   Timer Interrupts ......................................................................................................... 14-14
14.7   Reading and Writing 16-Bit Timer Module Registers ................................................ 14-15
14.8   Secondary Oscillator 32 kHz Crystal Input............................................................. 14-15
14.9   32-Bit Timer Configuration ...................................................................................... 14-16
14.10  32-Bit Timer Modes of Operation ............................................................................. 14-18
14.11  Reading and Writing Into 32-Bit Timers ................................................................. 14-21
14.12  Timer Operation in Power-Saving States ............................................................... 14-21
14.13  Peripherals Using Timer Modules ........................................................................... 14-22
14.14  Register Maps ........................................................................................................... 14-23
14.15  Related Application Notes ..................................................................................... 14-24
14.16  Revision History ....................................................................................................... 14-25
14.1 INTRODUCTION

Depending on the specific variant, the PIC24F device family offers several 16-bit timers. These timers are designated as Timer1, Timer2, Timer3, ..., etc.

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

- TMRx: 16-Bit Timer Count register
- PRx: 16-Bit Timer Period register associated with the timer
- TxCON: 16-Bit Timer Control register associated with the timer

Each timer module also has the associated bits for interrupt control:

- Interrupt Enable Control bit (TxIE)
- Interrupt Flag Status bit (TxIF)
- Interrupt Priority Control bits (TxIP<2:0>)

With certain exceptions, all of the 16-bit timers have the same functional circuitry. The 16-bit timers are classified into three types to account for their functional differences:

- Type A time base
- Type B time base
- Type C time base

Some 16-bit timers can be combined to form a 32-bit timer.

This section does not describe the dedicated timers that are associated with peripheral devices. For example, this includes the time base associated with the input capture or output compare modules.
14.2 TIMER VARIANTS

All 16-bit timers available on the PIC24F devices are functionally identical with certain exceptions. The 16-bit timers are classified into three functional types; Type A timers, Type B timers and Type C timers.

**Note:** Please refer to the specific device data sheet for the available timers and their corresponding type.

### 14.2.1 Type A Timer

At least one Type A timer is available on most PIC24F devices. For most PIC24F devices, Timer1 is a Type A timer. A Type A timer has the following unique features over other types:

- Can be operated from the device low-power 32 kHz oscillator
- Can be operated in an Asynchronous mode from an external clock source

In particular, the unique features of a Type A timer allow it to be used for timekeeping functions or as a secondary system clock source.

**Note:** Most PIC24F devices have an HW RTCC module eliminating the need for hardware RTCC.

---

**Figure 14-1: Type A Timer Block Diagram**

---

**Note 1:** Refer to Section 6. “Oscillator” for information on enabling the secondary oscillator.
14.2.2 Type B Timer

Timer2 and Timer4, if present, are Type B timers on most PIC24F devices. A Type B timer has the following unique features over other types of timers:

- A Type B timer can be concatenated with a Type C timer to form a 32-bit timer. The TxCB register for a Type B timer has the T32 control bit to enable the 32-bit timer function.
- The clock synchronization for a Type B timer is performed after the prescale logic. The advantage of placing clock synchronization after the prescale logic is explained in Section 14.4.4 “Timer Operation with Fast External Clock Source”.

A block diagram of the Type B timer is shown in Figure 14-2.

Figure 14-2: Type B Timer Block Diagram
14.2.3 Type C Timer

Timer3 and Timer5 are Type C timers on most PIC24F devices. A Type C timer has the following unique features over other types of timers:

- A Type C timer can be concatenated with a Type B timer to form a 32-bit timer.
- On a given device, at least one Type C timer has the ability to trigger an A/D conversion.

A block diagram of the Type C timer is shown in Figure 14-3.

Figure 14-3: Type C Timer Block Diagram

* The ADC Event Trigger is available only on Timer4/5.

**Note 1:** In certain variants of the PIC24F family, the TxCK pin may not be available. Refer to the device data sheet for the I/O pin details. In such cases, the timer must use the system clock (Fosc/2) as its input clock unless it is configured for 32-bit operation.
14.3 CONTROL REGISTERS

Register 14-1: TxCON: Type A Time Base Control

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>TON: Timerx On bit. 1 = Starts the timer, 0 = Stops the timer.</td>
</tr>
<tr>
<td>14</td>
<td>Unimplemented: Read as ‘0’.</td>
</tr>
<tr>
<td>13</td>
<td>TSIDL: Stop in Idle Mode bit. 1 = Discontinue timer operation when device enters Idle mode, 0 = Continue timer operation in Idle mode.</td>
</tr>
<tr>
<td>12-7</td>
<td>Unimplemented: Read as ‘0’.</td>
</tr>
<tr>
<td>6</td>
<td>TGATE: Timerx Gated Time Accumulation Enable bit.</td>
</tr>
<tr>
<td>5-4</td>
<td>TCKPS&lt;1:0&gt;: Timerx Input Clock Prescale Select bits.</td>
</tr>
<tr>
<td>3</td>
<td>Unimplemented: Read as ‘0’.</td>
</tr>
<tr>
<td>2</td>
<td>TSYNC: Timerx External Clock Input Synchronization Select bit.</td>
</tr>
<tr>
<td>1</td>
<td>TCS: Timerx Clock Source Select bit.</td>
</tr>
<tr>
<td>0</td>
<td>Unimplemented: Read as ‘0’.</td>
</tr>
</tbody>
</table>

Legend:
- R = Readable bit
- W = Writable bit
- U = Unimplemented bit, read as ‘0’
- ‘1’ = Bit is set
- ‘0’ = Bit is cleared
- x = Bit is unknown

- bit 15: TON: Timerx 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: 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: Unimplemented: Read as ‘0’
- bit 2: TSYNC: Timerx 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: Timerx Clock Source Select bit
  - 1: External clock from TxCK pin
  - 0: Internal clock (FOSC/2)
- bit 0: Unimplemented: Read as ‘0’
## Register 14-2: TxCON: Type B Time Base Control

<table>
<thead>
<tr>
<th>bit 15</th>
<th>bit 14</th>
<th>bit 13</th>
<th>bit 12-7</th>
<th>bit 6</th>
<th>bit 5-4</th>
<th>bit 3</th>
<th>bit 2</th>
<th>bit 1</th>
<th>bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>TON</td>
<td>Unimplemented</td>
<td>TSIDL</td>
<td>Stop in Idle Mode bit</td>
<td>TGATE</td>
<td>Timerx Gated Time Accumulation Enable bit</td>
<td>TCKPS&lt;1:0&gt;</td>
<td>32-Bit Timer Mode Select bit</td>
<td>TCS</td>
<td>Unimplemented</td>
</tr>
</tbody>
</table>

### 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 Descriptions:

#### bit 15: TON
- Timerx On bit
  - When T32 = 1 (in 32-Bit Timer mode):
    - 1 = Starts 32-bit TMRx:TMRy timer pair
    - 0 = Stops 32-bit TMRx:TMRy 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 form a 32-bit timer
  - 0 = TMRx and TMRy form separate 16-bit timer

#### 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’
Register 14-3: TyCON: Type C Time Base Control

<table>
<thead>
<tr>
<th>R/W-0</th>
<th>U-0</th>
<th>R/W-0</th>
<th>U-0</th>
<th>U-0</th>
<th>U-0</th>
<th>U-0</th>
<th>U-0</th>
</tr>
</thead>
<tbody>
<tr>
<td>TON(1)</td>
<td>—</td>
<td>TSIDL</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bit 15</th>
<th>bit 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>U-0</td>
<td>R/W-0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>R/W-0</th>
<th>R/W-0</th>
<th>U-0</th>
<th>U-0</th>
<th>R/W-0</th>
<th>U-0</th>
</tr>
</thead>
<tbody>
<tr>
<td>—</td>
<td>TGATE(1)</td>
<td>TCKPS&lt;1:0&gt;</td>
<td>—</td>
<td>—</td>
<td>TCS</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bit 7</th>
<th>bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>U-0</td>
<td>R/W-0</td>
</tr>
</tbody>
</table>

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

Note 1: When 32-bit operation is enabled (T2CON<3> = 1), these bits have no effect on Timery operation; all timer functions are set through T2CON.
14.4 MODES OF OPERATION

Each timer module can operate in one of the following modes:

- Timer
- Synchronous counter
- As a gated timer
- Asynchronous counter (Type A and C time base only)

The Timer modes are determined by the following bits:

- TCS (TxCON<1>): Timer Clock Source Control bit
- TSYNC (TxCON<2>): Timer Synchronization Control bit (Type A time base only)
- TGATE (TxCON<6>): Timer Gate Control bit

Each timer module is enabled or disabled using the TON bit (TxCON<15>).

**Note:** Only Type A and C time bases support the External Asynchronous Counter mode.

14.4.1 Timer Mode

All types of timers have the ability to operate in Timer mode based on the system clock. In Timer mode, the input clock to the timer is provided from the internal system clock (Fosc/2). When enabled, the timer increments once per instruction cycle for a 1:1 prescaler setting. The Timer mode is selected by clearing the TCS control bit (TxCON<1>). The Synchronous mode control bit, TSYNC (TxCON<2>), has no effect, since the system clock source is used to generate the timer clock.

**Example 14-1: Initialization Code for 16-Bit Timer Using System Clock**

```c
/* The following code example will enable Timer1 interrupts, load the Timer1
Period register and start Timer1.

When a Timer1 period match interrupt occurs, the interrupt service
routine must clear the Timer1 interrupt status flag in software.
*/

T1CON = 0x00; //Stops the Timer1 and reset control reg.
TMR1 = 0x00; //Clear contents of the timer register
PR1 = 0xFFFF; //Load the Period register with the value 0xFFFF
IPC0bits.T1IP = 0x01; //Setup Timer1 interrupt for desired priority level
// (This example assigns level 1 priority)
IFS0bits.T1IF = 0; //Clear the Timer1 interrupt status flag
IEC0bits.T1IE = 1; //Enable Timer1 interrupts
T1CONbits.TON = 1; //Start Timer1 with prescaler settings at 1:1 and
//clock source set to the internal instruction cycle

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

IFS0bits.T1IF = 0; //Reset Timer1 interrupt flag and Return from ISR
```


14.4.2 Synchronous Counter Mode Using External Clock Input

When the TCS control bit (TxCON<1>) is set, the clock source for the timer is provided externally and the selected timer increments on every rising edge of clock input on the TxCK pin.

For a Type A time base, the external clock synchronization must be enabled to run it in Synchronized Counter mode. This is accomplished by setting the TSYNC control bit (TxCON<2>). For Type B and Type C time bases, the external clock input is always synchronized to the system instruction cycle clock, TCY.

When the timer is operated in the Synchronized Counter mode, there are minimum requirements for the external clock high time and low time. The synchronization of the external clock source with the device instruction clock is accomplished by sampling the external clock signal at two different times within an instruction cycle.

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

**Note:** The external input clock must meet certain minimum high time and low time requirements when used in the Synchronous Counter mode.

Example 14-2: Initialization Code for 16-Bit Synchronous Counter Mode Using an External Clock Input

```c
/* The following code example will enable Timer1 interrupts, load the
 Timer1 Period register and start Timer1 using an external clock
 and a 1:8 prescaler setting.

 When a Timer1 period match interrupt occurs, the interrupt service
 routine must clear the Timer1 interrupt status flag in software.
 */

T1CON = 0x00; //Stops the Timer1 and reset control reg.
TMR1 = 0x00;  //Clear contents of the timer register
PR1 = 0x8CFF; //Load the Period register with the value 0x8CFF
IPC0bits.T1IP = 0x01; //Setup Timer1 interrupt for desired priority level
                     // (this example assigns level 1 priority)
IFS0bits.T1IF = 0; //Clear the Timer1 interrupt status flag
IEC0bits.T1IE = 1; //Enable Timer1 interrupts
T1CON = 0x8016; //Start Timer1 with prescaler settings at 1:8 and
                 //clock source set to the external clock in the
                 //synchronous mode

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

    IFS0bits.T1IF = 0; //Reset Timer1 interrupt flag and Return from ISR
}
```
14.4.3 Type A Timer Asynchronous Counter Mode Using External Clock Input

A Type A time base has the ability to operate in an Asynchronous Counting mode, using an external clock source connected to the TxCK pin. When the TSYNC control bit (TxCON<2>) is cleared, the external clock input is not synchronized with the device system clock source. The time base continues to increment asynchronously to the internal device clock.

The asynchronous operation time base is beneficial for the following applications:

- The time base can operate during Sleep mode and can generate an interrupt on period register match that will wake-up the processor.
- The time base can be clocked from the low-power 32 kHz oscillator to provide a secondary system clock source.

### Note 1:
Only Type A time bases support the Asynchronous Counter mode.

### Note 2:
The external input clock must meet certain minimum high time and low time requirements when Timerx is used in the Asynchronous Counter mode.

Example 14-3: Initialization Code for 16-Bit Asynchronous Counter Mode Using an External Clock Input

```c
/* The following code example will enable Timer1 interrupts, load the Timer1 Period register and start Timer1 using an asynchronous external clock and a 1:8 prescaler setting. When a Timer1 period match interrupt occurs, the interrupt service routine must clear the Timer1 interrupt status flag in software. */

T1CON = 0x00;   //Stops the Timer1 and reset control reg.
TMR1 = 0x00;    //Clear contents of the timer register
PR1 = 0x8CFF;   //Load the Period register with the value 0x8CFF
IPCONbits.T1IP = 0x01; //Setup Timer1 interrupt for desired priority level // (this example assigns level 1 priority)
IFS0bits.T1IF = 0; //Clear the Timer1 interrupt status flag
IEC0bits.T1IE = 1; //Enable Timer1 interrupts
T1CON = 0x8012; //Start Timer1 with prescaler settings at 1:8 and //clock source set to the external clock in the //asynchronous mode

/* Example code for Timer1 ISR*/
void __attribute__((__interrupt__, __shadow__)) _T1Interrupt(void)
{
    /* Interrupt Service Routine code goes here */
    IFS0bits.T1IF = 0; //Reset Timer1 interrupt flag and Return from ISR
}
```
14.4.4 Timer Operation with Fast External Clock Source

In some applications, it may be desirable to use one of the timers to count clock edges from a relatively high-frequency external clock source. In these situations, Type A and Type B time bases are the most suitable choices for counting the external clock source because the clock synchronization logic for these timers is located after the timer prescaler (see Figure 14-1 and Figure 14-2). This allows a higher external clock frequency to be used that will not violate the minimum high and low times required by the prescaler. When a timer prescaler ratio other than 1:1 is selected for a Type A or Type B time base, the minimum high and low times for the external clock input are reduced by the chosen prescaler ratio.

A Type A time base is unique because it can be operated in an Asynchronous mode, eliminating any prescaler timing requirements.

Note that in all cases, there are minimum high and low times for the external clock signal that cannot be exceeded. These minimum times are required to satisfy the I/O pin timing requirements.

Please refer to the specific device data sheet for the external clock timing specifications associated with the time bases.

14.4.5 Gated Time Accumulation Mode

The Gated Time Accumulation mode allows the internal timer register to increment based upon the duration of the high time applied to the TxCK pin. In the Gated Time Accumulation mode, the timer clock source is derived from the internal system clock. When the TxCK pin state is high, the timer register will count up until a period match has occurred, or the TxCK pin state is changed to a low state. A pin state transition from high-to-low will set the TxIF interrupt flag. Depending on when the edge occurs, the interrupt flag is asserted 1 or 2 instruction cycles after the falling edge of the signal on the TxCK pin.

The TGATE control bit (TxCON<6>) must be set to enable the Gated Time Accumulation mode. The timer must be enabled, TON (TxCON<15>) = 1, and the timer clock source set to the internal clock, TCS (TxCON<1>) = 0.

The gate operation starts on a rising edge of the signal applied to the TxCK pin and terminates on the falling edge of the signal applied to the TxCK pin. The respective timer will increment while the external gate signal is high.

The falling edge of the gate signal sets the TxIF interrupt flag and generates an interrupt if enabled.

**Note:** The timer will not interrupt the CPU when a timer period match occurs in Gate Time Accumulation mode.

The resolution of the timer count is directly related to the timer clock period. For a timer prescaler of 1:1, the timer clock period is one instruction cycle. For a timer prescaler of 1:256, the timer clock period is 256 times the instruction cycle. The timer clock resolution can be associated with the pulse width of the gate signal. Refer to the “Electrical Characteristics” section in the specific device data sheet for further details on the gate width pulse requirements.
**Section 14. Timers**

**Figure 14-4: Gated Timer Mode Operation**

![Gated Timer Mode Operation Diagram]

**Example 14-4: Initialization Code for 16-Bit Gated Time Accumulation Mode**

```c
/* The following code example will enable Timer2 interrupts, load the Timer2 period register and start Timer2 using an internal clock and an external gate signal. On the falling edge of the gate signal a Timer2 interrupt occurs. The interrupt service routine must clear the Timer2 interrupt status flag in software.
 */

T2CON = 0x00;  //Stops the Timer2 and reset control reg.
TMR2 = 0x00;   //Clear contents of the timer register
PR2 = 0xFFFF;  //Load the Period register with the value 0xFFFF
IPC1bits.T2IP = 0x01;  //Setup Timer2 interrupt for desired priority level
                      // (this example assigns level 1 priority)
IFS0bits.T2IF = 0;  //Clear the Timer2 interrupt status flag
IEC0bits.T2IE = 1;  //Enable Timer2 interrupts
T2CONbits.TON = 1;  //Start Timer2

void __attribute__((__interrupt__, __shadow__)) _T2Interrupt(void)
{
    /* Interrupt Service Routine code goes here */

    IFS0bits.T2IF = 0;  //Reset Timer2 interrupt flag and Return from ISR
}
```
14.5  TIMER PRESCALERS

The input clock (Fosc/2 or external clock) 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 TCKPS<1:0> control bits (TxCON<5:4>). The prescaler counter is cleared when any of the following occurs:

- A write to the TMRx register
- Clearing TON (TxCON<15>) to ‘0’
- Any device Reset

Note: The TMRx register is not cleared when TxCON is written.

14.6  TIMER INTERRUPTS

A 16-bit timer has the ability to generate an interrupt on a period match or falling edge of the external gate signal, depending on the operating mode.

The TxIF bit is set when one of the following conditions is true:

- The timer count matches the respective period register and the timer module is not operating in Gated Time Accumulation mode.
- The falling edge of the “gate” signal is detected when the timer is operating in Gated Time Accumulation mode.

The TxIF bit must be cleared in software.

A timer is enabled as a source of interrupt via the respective Timer Interrupt Enable bit, TxIE. Furthermore, the interrupt priority level bits (TxIP<2:0>) must be written with a non-zero value in order for the timer to be a source of interrupt. Refer to Section 8. “Interrupts” for further details.

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

Figure 14-5: Interrupt Timing for Timer Period Match
14.7 READING AND WRITING 16-BIT TIMER MODULE REGISTERS

- All timer module SFRs can be written to as a byte (8 bits) or as a word (16 bits).
- All timer module SFRs can only be read as a word (16 bits).

14.7.1 Writing to the 16-Bit Timers

The timer and its respective period register can be written to while the module is operating. The user should be aware of the following when byte writes are performed:

- If the timer is incrementing and the low byte of the timer is written to, the upper byte of the timer is not affected. If 0xFF is written into the low byte of the timer, the next timer count clock after this write will cause the low byte to rollover to 0x00 and generate a carry into the high byte of the timer.
- If the timer is incrementing and the high byte of the timer is written to, the low byte of the timer is not affected. If the low byte of the timer contains 0xFF when the write occurs, the next timer count clock will generate a carry from the timer low byte and this carry will cause the upper byte of the timer to increment.

When the TMRx register is written to (word or byte) via an instruction, the TMRx register increment is masked and does not occur during that instruction cycle.

Writers to a timer with an Asynchronous mode should be avoided in a real timekeeping application. See Section 14.4.1 “Timer Mode” for information on Asynchronous Counter mode.

14.7.2 Reading From the 16-Bit Timers

All reads of the timers and their associated SFRs must be word reads (16 bits). A byte read will have no effect (‘0’ will be returned).

The timer and respective period register can be read while the module is operating. A read of the TMRx register does not prevent the timer from incrementing during the same instruction cycle.

14.8 SECONDARY OSCILLATOR 32 kHz CRYSTAL INPUT

In each device variant, a 32 kHz crystal oscillator is available to the Type A timer module for Real-Time Clock (RTC) type of applications.

- The secondary oscillator becomes the clock source for the timer when the secondary oscillator is enabled and the timer is configured to use the external clock source.
- The secondary oscillator is enabled by setting the SOSCEN control bit in the OSCCON register.
- The 32 kHz crystal is connected to the SOSCO/SOSCI device pins.

Refer to Section 6. “Oscillator” for further details.
14.9  32-BIT TIMER CONFIGURATION

A 32-bit timer module can be formed by combining a Type B and a Type C 16-bit timer module. The Type C time base becomes the most significant word (msw) of the combined timer and the Type B time base is the least significant word (lsw).

When configured for 32-bit operation, the control bits for the Type B time base control the operation of the 32-bit timer. The control bits in the TxCON register for the Type C time base have no effect.

For interrupt control, the combined 32-bit timer uses the interrupt enable, interrupt flag and interrupt priority control bits of the Type C time base. The interrupt control and status bits for the Type B time base are not used during 32-bit timer operation.

The following configuration settings assume Timer3 is a Type C time base and Timer2 is a Type B time base:

- TON (T2CON<15>) = 1.
- T32 (T2CON<3>) = 1.
- TCKPS<1:0> bits (T2CON<5:4>) are used to set the Prescaler mode for Timer2 (Type B time base).
- The TMR3:TMR2 register pair contains the 32-bit value of the timer module. The TMR3 (Type C time base) register is the most significant word, while the TMR2 (Type B time base) register is the least significant word of the 32-bit timer value.
- The PR3:PR2 register pair contains the 32-bit period value that is used for comparison with the TMR3:TMR2 timer value.
- T3IE (IEC0<8>) is used to enable the 32-bit timer interrupt for this configuration.
- T3IF (IFS0<8>) is used as a status flag for the timer interrupt.
- T3IP<2:0> bits (IPC2<2:0>) set the interrupt priority level for the 32-bit timer.
- T3CON<15:0> are “don’t care” bits.

A block diagram representation of the 32-bit timer module using Timer2 and Timer3 as an example is shown in Figure 14-6.

Note: Input capture and output compare are not available in 32-Bit Timer mode.
Figure 14-6: Type B/Type C Timer Pair Block Diagram (32-Bit Timer)

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>T2CK</td>
<td>(T4CK)</td>
</tr>
<tr>
<td>TGATE</td>
<td></td>
</tr>
<tr>
<td>Set T3IF (T5IF)</td>
<td></td>
</tr>
<tr>
<td>ADC Event Trigger*</td>
<td></td>
</tr>
<tr>
<td>Q</td>
<td></td>
</tr>
<tr>
<td>CK</td>
<td></td>
</tr>
<tr>
<td>PR3</td>
<td>PR2</td>
</tr>
<tr>
<td>Equal</td>
<td></td>
</tr>
<tr>
<td>MSB</td>
<td>LSB</td>
</tr>
<tr>
<td>TMR3HLD</td>
<td>TMR2 (TMR4)</td>
</tr>
<tr>
<td>Read TMR2 (TMR4)</td>
<td>Write TMR2 (TMR4)</td>
</tr>
<tr>
<td>TMR3HLD</td>
<td>TMR5HLD</td>
</tr>
<tr>
<td>Data Bus&lt;15:0&gt;</td>
<td></td>
</tr>
</tbody>
</table>

* The ADC Event Trigger is available only on Timer4/5.

**Note:** The 32-bit Timer Configuration bit, T32, must be set for 32-bit timer/counter operation. All control bits are respective to the T2CON and T4CON registers.
14.10 32-BIT TIMER MODES OF OPERATION

14.10.1 Timer Mode

Example 14-5 shows how to configure a 32-bit timer in Timer mode. This example assumes Timer2 is a Type B time base and Timer3 is a Type C time base. For 32-bit timer operation, the T32 control bit must be set in the T2CON register (Type B time base). When Timer2 and Timer3 are configured for a 32-bit timer, the T3CON control bits are ignored. Only the T2CON control bits are required for setup and control. The Timer2 clock and gate input is utilized for the 32-bit timer module, but an interrupt is generated with the T3IF flag. Timer2 is the lsw and Timer3 is the msw of the 32-bit timer. TMR3 is incremented by an overflow (carry out) from TMR2. The 32-bit timer increments up to a match value preloaded into the combined 32-bit period register, formed by PR2 and PR3, then rolls over and continues. For a maximum 32-bit timer count, load PR3:PR2 with a value of 0xFFFFFFFF. An interrupt is generated on a period match if enabled.

Example 14-5: Initialization Code for 32-Bit Timer Using Instruction Cycle as Input Clock

```c
/* The following code example will enable Timer3 interrupts, load the
Timer3:Timer2 Period Register and start the 32-bit timer module
consisting of Timer3 and Timer2.

When a 32-bit period match interrupt occurs, the user must clear the
Timer3 interrupt status flag in software.
*/

T2CON = 0x00;  //Stops any 16/32-bit Timer2 operation
T3CON = 0x00;  //Stops any 16-bit Timer3 operation
TMR3 = 0x00;   //Clear contents of the timer3 register
TMR2 = 0x00;   //Clear contents of the timer2 register
PR3 = 0xFFFF;  //Load the Period register3 with the value 0xFFFF
PR2 = 0xFFFF;  //Load the Period register2 with the value 0xFFFF
IPC2bits.T3IP = 0x01; //Setup Timer3 interrupt for desired priority level
                      //(this example assigns level 1 priority)
IFS0bits.T3IF = 0; //Clear the Timer3 interrupt status flag
IEC0bits.T3IE = 1; //Enable Timer3 interrupts
T2CONbits.T32 = 1; //Enable 32-bit Timer operation
T2CONbits.TON = 1; //Start 32-bit timer with prescaler
                   //settings at 1:1 and clock source set to
                   //the internal instruction cycle

void __attribute__((__interrupt__, __shadow__)) _T3Interrupt(void)
{
  /* Interrupt Service Routine code goes here */
  IFS0bits.T3IF = 0; //Reset Timer1 interrupt flag and Return from ISR
}
```
14.10.2 Synchronous Counter Mode

The 32-bit timer operates similarly to a 16-bit timer in Synchronous Counter mode. Example 14-6 shows how to configure a 32-bit timer in Synchronous Counter mode. This example assumes Timer2 is a Type B time base and Timer3 is a Type C time base.

Example 14-6: Initialization Code for 32-Bit Synchronous Counter Mode Using an External Clock Input

```c
/* The following code example will enable Timer2 interrupts, load the Timer3:Timer2 Period register and start the 32-bit timer module consisting of Timer3 and Timer2.

When a 32-bit period match interrupt occurs, the user must clear the Timer3 interrupt status flag in the software.
*/
T2CON = 0x00; //Stops any 16/32-bit Timer2 operation
T3CON = 0x00; //Stops any 16-bit Timer3 operation
TMR3 = 0x00; //Clear contents of the timer3 register
TMR2 = 0x00; //Clear contents of the timer2 register
PR3 = 0xFFFF; //Load the Period register3 with the value 0xFFFF
PR2 = 0xFFFF; //Load the Period register2 with the value 0xFFFF
IPC2bits.T3IP = 0x01; //Setup Timer3 interrupt for desired priority level //this example assigns level 1 priority
IFS0bits.T3IF = 0; //Clear the Timer3 interrupt status flag
IEC0bits.T3IE = 1; //Enable Timer3 interrupts
T2CON = 0x801A; //Enable 32-bit Timer operation and start 32-bit timer with prescaler settings at 1:8 //and clock source set to external clock

void __attribute__((__interrupt__, __shadow__)) _T3Interrupt(void)
{
    /* Interrupt Service Routine code goes here */
    IFS0bits.T3IF = 0; //Reset Timer1 interrupt flag and Return from ISR
}
```

14.10.3 Asynchronous Counter Mode

Type B and Type C time bases do not support the Asynchronous External Clock mode, therefore, no 32-bit Asynchronous Counter mode is supported.

14.10.4 Gated Time Accumulation Mode

The 32-bit timer operates similarly to a 16-bit timer in Gated Time Accumulation mode. Example 14-7 shows how to configure a 32-bit timer in Gated Time Accumulation mode. This example assumes Timer2 is a Type B time base and Timer3 is a Type C time base.

Example 14-7: Initialization Code for 32-Bit Gated Time Accumulation Mode

```c
/* The following code example will enable Timer2 interrupts, load the Timer3:Timer2 Period register and start the 32-bit timer module consisting of Timer3 and Timer2.
When a 32-bit period match occurs the timer will simply roll over and continue counting.

However, when at the falling edge of the Gate signal on T2CK an interrupt is generated, if enabled. The user must clear the Timer3 interrupt status flag in the software.
*/

T2CON = 0x00; //Stops any 16/32-bit Timer2 operation
T3CON = 0x00; //Stops any 16-bit Timer3 operation
TMR3 = 0x00; //Clear contents of the timer3 register
TMR2 = 0x00; //Clear contents of the timer2 register
PR3 = 0xFFFF; //Load the Period register3 with the value 0xFFFF
PR2 = 0xFFFF; //Load the Period register2 with the value 0xFFFF
IPC2bits.T3IP = 0x01; //Setup Timer3 interrupt for desired priority level (this example assigns level 1 priority)
IFS0bits.T3IF = 0; //Clear the Timer3 interrupt status flag
IEC0bits.T3IE = 1; //Enable Timer3 interrupts
T2CON = 0x8048; //Enable 32-bit Timer operation and start 32-bit timer in gated time accumulation mode.

void __attribute__((__interrupt__, __shadow__)) _T3Interrupt(void)
{
    /* Interrupt Service Routine code goes here */
    IFS0bits.T3IF = 0; //Reset Timer1 interrupt flag and Return from ISR
}
```
14.11 READING AND WRITING INTO 32-BIT TIMERS

In order 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 utilized (see Figure 14-6). Each Type C time base has a register called TMRxHLD that is used when reading or writing the timer register pair. The TMRxHLD registers are only used when their respective timers are configured for 32-bit operation.

Assuming TMR3:TMR2 form a 32-bit timer pair, the user should first read the lsw of the timer value from the TMR2 register. The read of the lsw will automatically transfer the contents of TMR3 into the TMR3HLD register. The user can then read TMR3HLD to get the msw of the timer value. This is shown in Example 14-8:

Example 14-8: Reading From a 32-Bit Timer

```c
/* The following code segment reads the 32-bit timer formed by the
Timer3-Timer2 pair into the registers W1(MS Word) and W0(LS Word).
*/
unsigned int temp_lsb;
unsigned int temp_msb;
temp_lsb = TMR2; //Transfer the LSW into temp_lsb
temp_msb = TMR3HLD; //Transfer the MSW from the holding register to into
                       //temp_msb
```

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

14.12 TIMER OPERATION IN POWER-SAVING STATES

14.12.1 Timer Operation in Sleep Mode

When the device enters Sleep mode, the system clock is disabled. If the timer module is running from the internal clock source (Fosc/2), it will also be disabled.

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

- The Timer1 module is enabled, TON (TxCON<15>) = 1;
- The Timer1 clock source is selected as external, TCS (TxCON<1>) = 1 and
- The TSYNC bit (TxCON<2>) is set to logic ‘0’ (Asynchronous Counter mode enabled).

**Note:** Asynchronous counter operation is only supported for the Timer1 module.

When all of the above conditions are met, Timer1 will continue to count and detect period matches when the device is in Sleep mode. When a match between the timer and the period register occurs, the TxIF bit will be set and an interrupt can be generated to optionally wake the device from Sleep. Refer to Section 10. “Power-Saving Features” for further details.

14.12.2 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 modules can optionally continue to operate in Idle mode.

The TSIDL bit (TxCON<13>) selects if the timer module will stop in Idle mode or continue to operate normally. If TSIDL = 0, the module will continue operation in Idle mode. If TSIDL = 1, the module will stop in Idle mode.

14.12.3 Timer Operation in Doze Mode

Timer operation in Doze mode is the same as in normal mode. When the device enters Doze mode, the system clock sources remain functional and the CPU may run at a slower clock rate. Refer to Section 10. “Power-Saving Features” for further details.
14.13 PERIPHERALS USING TIMER MODULES

14.13.1 Time Base for Input Capture/Output Compare

The input capture and output compare peripherals can select one of two timer modules as their time base. Refer to Section 15. “Input Capture”, Section 16. “Output Compare” and the specific device data sheet for further details.

14.13.2 A/D Special Event Trigger

On each device variant, one Type C time base has the capability to generate a special A/D conversion trigger signal, on a period match, in both 16 and 32-bit modes. The timer module provides a conversion start signal to the A/D 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 A/D 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 A/D 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 A/D converter control registers. Refer to Section 17. “10-Bit A/D Converter” and the specific device data sheet for additional information.

14.13.3 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, PRx, is written with a non-zero value and the TMRx register is initialized to a value of 1 less than the value written to the period register. The timer must be configured for a 1:1 clock prescaler. An interrupt will be generated when the next rising edge of the external clock signal is detected.

14.13.4 I/O Pin Control

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

14.13.5 Enabling Timer

To enable the timer, the Timer Module Disable bit (TxMD in PMD1 register) must be cleared along with setting the module enable bit. Setting the TxMD bit will disable all clock sources to that module, reducing its power consumption to an absolute minimum. In this state, the control and status registers associated with the peripheral will also be disabled, so writes to those registers will have no effect and read values will be invalid.
## 14.14 REGISTER MAPS

A summary of the Special Function Registers associated with the PIC24F timer module is provided in Table 14-1.

| Table 14-1: Special Function Registers Associated with Timer Modules |
|-------------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| **SFR Name** | **Bit 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** |
|-------------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| PMD1        | T5MD  | T4MD  | T3MD  | T2MD  | T1MD  | —     | —     | —     | 2C1MD | U2MD  | U1MD  | SPI2MD | SPI1MD | —     | —     | —     | —     | —     | —     | ADOMD | 0000   |
| TMR1        | TMR1  | —     | —     | —     | —     | —     | —     | —     | —     | —     | —     | —     | —     | —     | —     | —     | —     | —     | —     | xxx   |        |
| PR1         | PR1   | —     | —     | —     | —     | —     | —     | —     | —     | —     | —     | —     | —     | —     | —     | —     | —     | —     | —     | yyyy  |        |

**Note:** Please refer to the specific device data sheet for memory map details.
### 14.15 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 PIC24F device family, but the concepts are pertinent and could be used with modification and possible limitations. The current application notes related to the Timer modules are:

<table>
<thead>
<tr>
<th>Title</th>
<th>Application Note #</th>
</tr>
</thead>
<tbody>
<tr>
<td>Using Timer1 in Asynchronous Clock Mode</td>
<td>AN580</td>
</tr>
</tbody>
</table>

**Note:** Please visit the Microchip web site ([www.microchip.com](http://www.microchip.com)) for additional application notes and code examples for the PIC24F family of devices.
14.16 REVISION HISTORY

Revision A (April 2006)

This is the initial released revision of this document.