**Welcome to E-XFL.COM**

## What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

## Applications of "Embedded - Microcontrollers"

| Details | |
|---|---|
| Product Status | Active |
| Core Processor | PIC |
| Core Size | 8-Bit |
| Speed | 32MHz |
| Connectivity | I²C, LINbus, SPI, UART/USART |
| Peripherals | Brown-out Detect/Reset, POR, PWM, WDT |
| Number of I/O | 26 |
| Program Memory Size | 14KB (8K x 14) |
| Program Memory Type | FLASH |
| EEPROM Size | 256 x 8 |
| RAM Size | 1K x 8 |
| Voltage - Supply (Vcc/Vdd) | 1.8V ~ 3.6V |
| Data Converters | A/D 24x12b; D/A 1x5b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 85°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 28-SSOP (0.209", 5.30mm Width) |
| Supplier Device Package | 28-SSOP |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/pic16lf18455-i-ss |

### 4.7.2 CONFIG2

**Name:** CONFIG2
**Address:** 0x8008

Configuration Word 2

Supervisor

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | $\overline{\text{DEBUG}}$ | STVREN | PPS1WAY | ZCD | BORV | |
| Access | | | R/P | R/P | R/P | R/P | R/P | U |
| Reset | | | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BOREN[1:0] | | $\overline{\text{LPBOREN}}$ | | | PWRTS[1:0] | | MCLRE |
| Access | R/P | R/P | R/P | U | U | R/P | R/P | R/P |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bit 13 – $\overline{\text{DEBUG}}$** Debugger Enable bit(1)

| Value | Description |
|---|---|
| 1 | Background debugger disabled |
| 0 | Background debugger enabled |

**Bit 12 – STVREN** Stack Overflow/Underflow Reset Enable bit

| Value | Description |
|---|---|
| 1 | Stack Overflow or Underflow will cause a Reset |
| 0 | Stack Overflow or Underflow will not cause a Reset |

**Bit 11 – PPS1WAY** PPSLOCKED bit One-Way Set Enable bit

| Value | Description |
|---|---|
| 1 | The PPSLOCKED bit can be cleared and set only once; PPS registers remain locked after one clear/set cycle |
| 0 | The PPSLOCKED bit can be set and cleared repeatedly (subject to the unlock sequence) |

**Bit 10 – ZCD** ZCD Control bit

| Value | Description |
|---|---|
| 1 | ZCD disabled. ZCD can be enabled by setting the ZCDSEN bit of the ZCDCON register. |
| 0 | ZCD always enabled, ZCDSEN bit is ignored |

**Bit 9 – BORV** Brown-out Reset Voltage Selection bit(2)

| Value | Description |
|---|---|
| 1 | Brown-out Reset voltage ($V_{BOR}$) set to lower trip point level |
| 0 | Brown-out Reset voltage ($V_{BOR}$) set to higher trip point level |

**Bits 7:6 – BOREN[1:0]** Brown-out Reset Enable bits
When enabled, Brown-out Reset Voltage ($V_{BOR}$) is set by BORV bit

**Figure 7-5. Accessing the Stack Example 2**

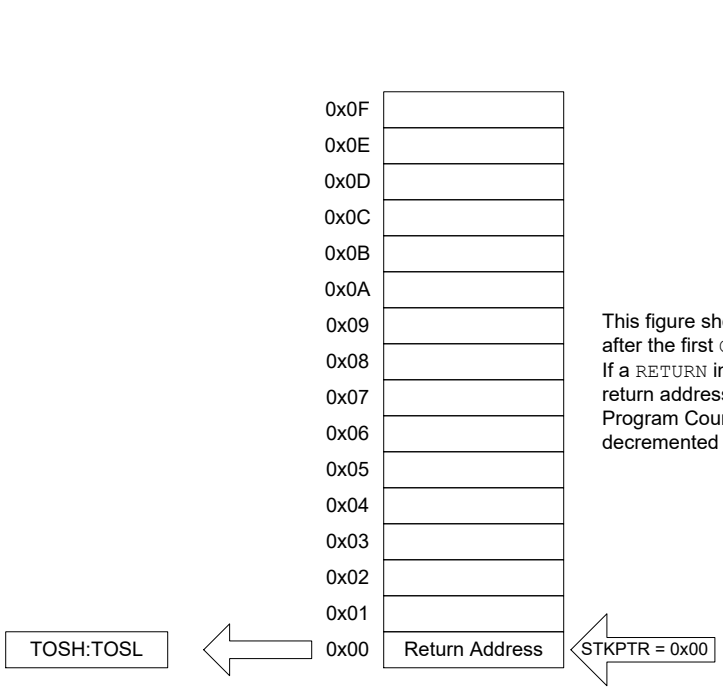



**Figure 7-6. Accessing the Stack Example 3**

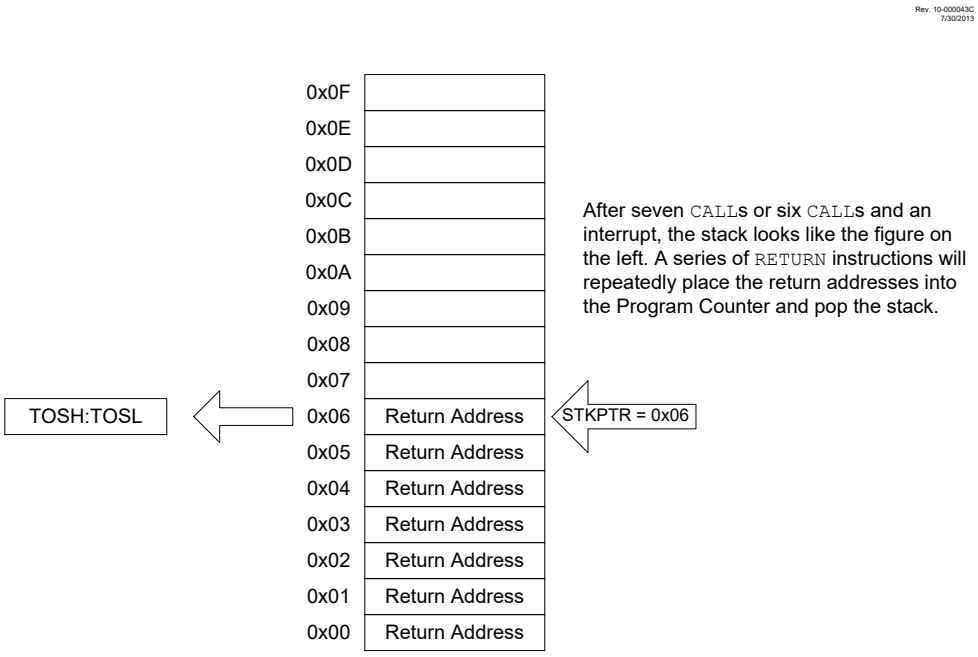
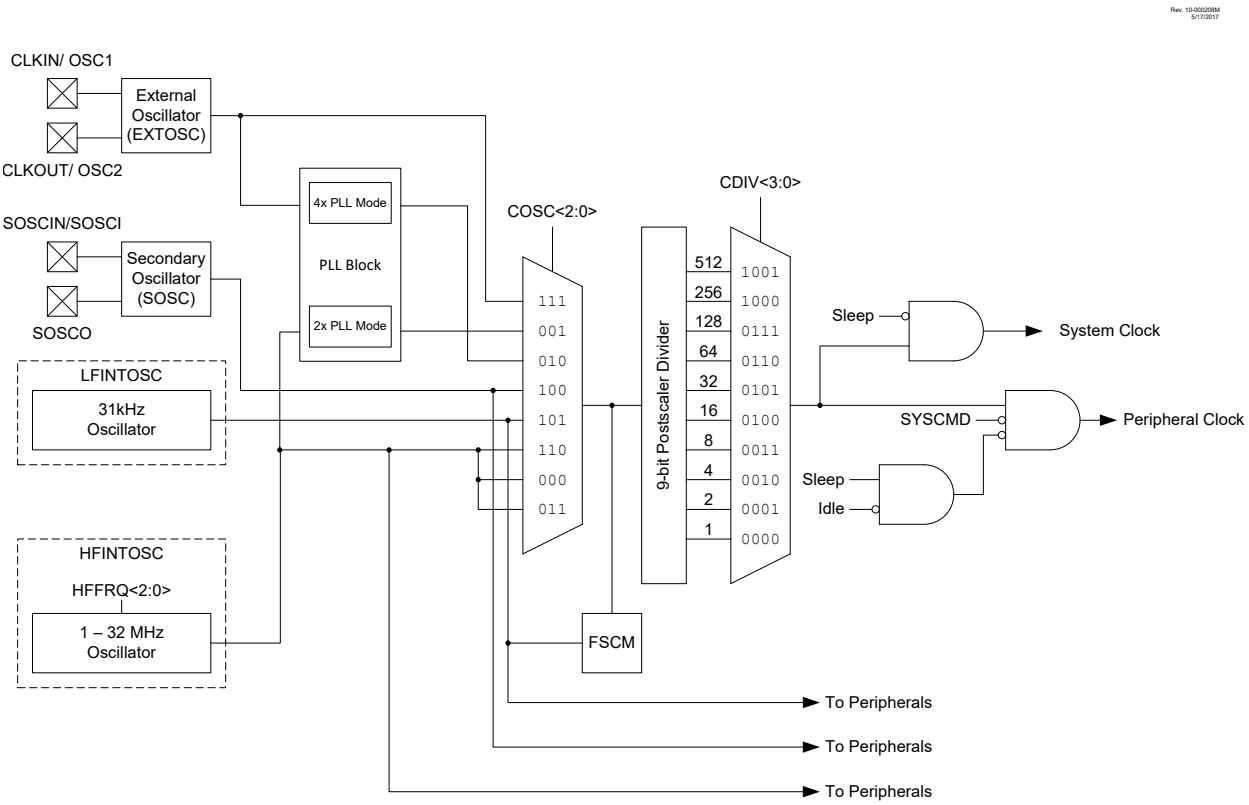


**Figure 9-1. Simplified PIC® MCU Clock Source Block Diagram**




**Related Links**

4.7.1 CONFIG1

## 9.2 Clock Source Types

Clock sources can be classified as external or internal.

External clock sources rely on external circuitry for the clock source to function. Examples are: oscillator modules (ECH, ECM, ECL mode), quartz crystal resonators or ceramic resonators (LP, XT and HS modes).

There is also a secondary oscillator block which is optimized for a 32.768 kHz external clock source, which can be used as an alternate clock source.

There are two internal oscillator blocks:

- HFINTOSC
- LFINTOSC

The HFINTOSC can produce clock frequencies from 1-32 MHz, and is responsible for generating the two MFINTOSC frequencies (500 kHz and 32 kHz) that can be used by some peripherals. The LFINTOSC generates a 31 kHz clock frequency.

There is a 4x PLL that can be used by the external oscillator. Additionally, there is a PLL that can be used by the HFINTOSC at certain frequencies.

**Related Links**

9.2.1.4 4x PLL

**Related Links**

9.6.4 OSCSTAT

9.6.5 OSCEN

#### 9.2.2.8 HFOR and MFOR Bits

The HFOR and MFOR bits indicate that the HFINTOSC and MFINTOSC is ready. These clocks are always valid for use at all times, but only accurate after they are ready.

When a new value is loaded into the OSCFRQ register, the HFOR and MFOR bits will clear, and set again when the oscillator is ready. During pending OSCFRQ changes the MFINTOSC clock will stall at a high or a low state, until the HFINTOSC resumes operation.

## 9.3 Clock Switching

The system clock source can be switched between external and internal clock sources via software using the New Oscillator Source (NOSC) and New Divider selection request (NDIV) bits. The following clock sources can be selected:

- External Oscillator (EXTOSC)
- High-Frequency Internal Oscillator (HFINTOSC)
- Low-Frequency Internal Oscillator (LFINTOSC)
- Secondary Oscillator (SOSC)
- EXTOSC with 4x PLL
- HFINTOSC with 2x PLL

### 9.3.1 New Oscillator Source (NOSC) and New Divider Selection Request (NDIV) Bits

The New Oscillator Source (NOSC) and New Divider Selection Request (NDIV) bits select the system clock source and frequency that are used for the CPU and peripherals.

When new values of NOSC and NDIV are written to OSCCON1, the current oscillator selection will continue to operate while waiting for the new clock source to indicate that it is stable and ready. In some cases, the newly requested source may already be in use, and is ready immediately. In the case of a divider-only change, the new and old sources are the same, so the old source will be ready immediately. The device may enter Sleep while waiting for the switch.

When the new oscillator is ready, the New Oscillator Ready (NOSCR) bit is set and also the Clock Switch Interrupt Flag (CSWIF) bit of PIR1 sets. If Clock Switch Interrupts are enabled (CSWIE = 1), an interrupt will be generated at that time. The Oscillator Ready (ORDY) bit of OSCCON3 can also be polled to determine when the oscillator is ready in lieu of an interrupt.

If the Clock Switch Hold (CSWHOLD) bit is clear, the oscillator switch will occur when the New Oscillator is READY bit (NOSCR) is set, and the interrupt (if enabled) will be serviced at the new oscillator setting.

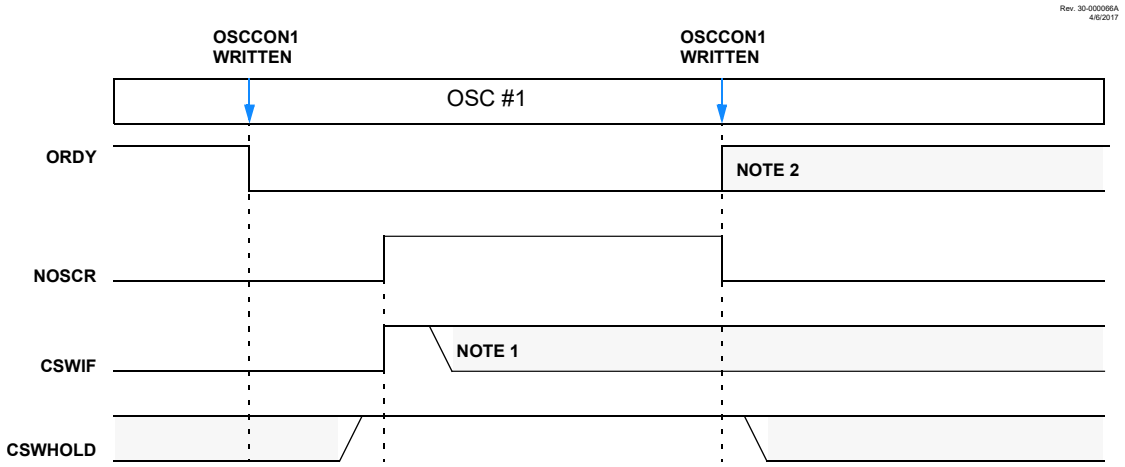
If CSWHOLD is set, the oscillator switch is suspended, while execution continues using the current (old) clock source. When the NOSCR bit is set, software should:

- Set CSWHOLD = 0 so the switch can complete, or
- Copy COSC into NOSC to abandon the switch.

If DOZE is in effect, the switch occurs on the next clock cycle, whether or not the CPU is operating during that cycle.

Changing the clock post-divider without changing the clock source (i.e., changing $F_{OSC}$ from 1 MHz to 2 MHz) is handled in the same manner as a clock source change, as described previously. The clock

**Figure 9-8. Clock Switch Abandoned**



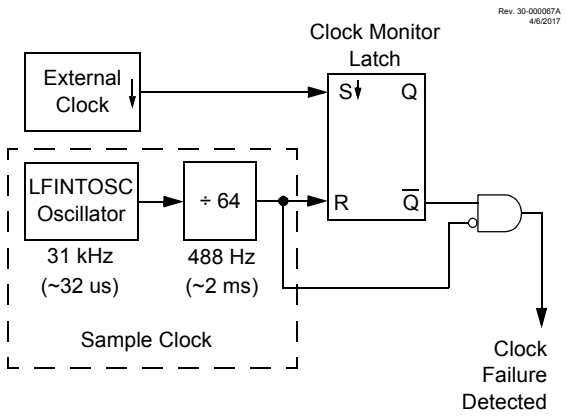

**Note:**

1. CSWIF may be cleared before or after rewriting OSCCON1; CSWIF is not automatically cleared.
2. ORDY = 0 if OSCCON1 does not match OSCCON2; a new switch will begin.

## 9.4 Fail-Safe Clock Monitor

The Fail-Safe Clock Monitor (FSCM) allows the device to continue operating should the external oscillator fail. The FSCM is enabled by setting the FCMEN bit in the Configuration Words. The FSCM is applicable to all external Oscillator modes (LP, XT, HS, ECL/M/H and Secondary Oscillator).

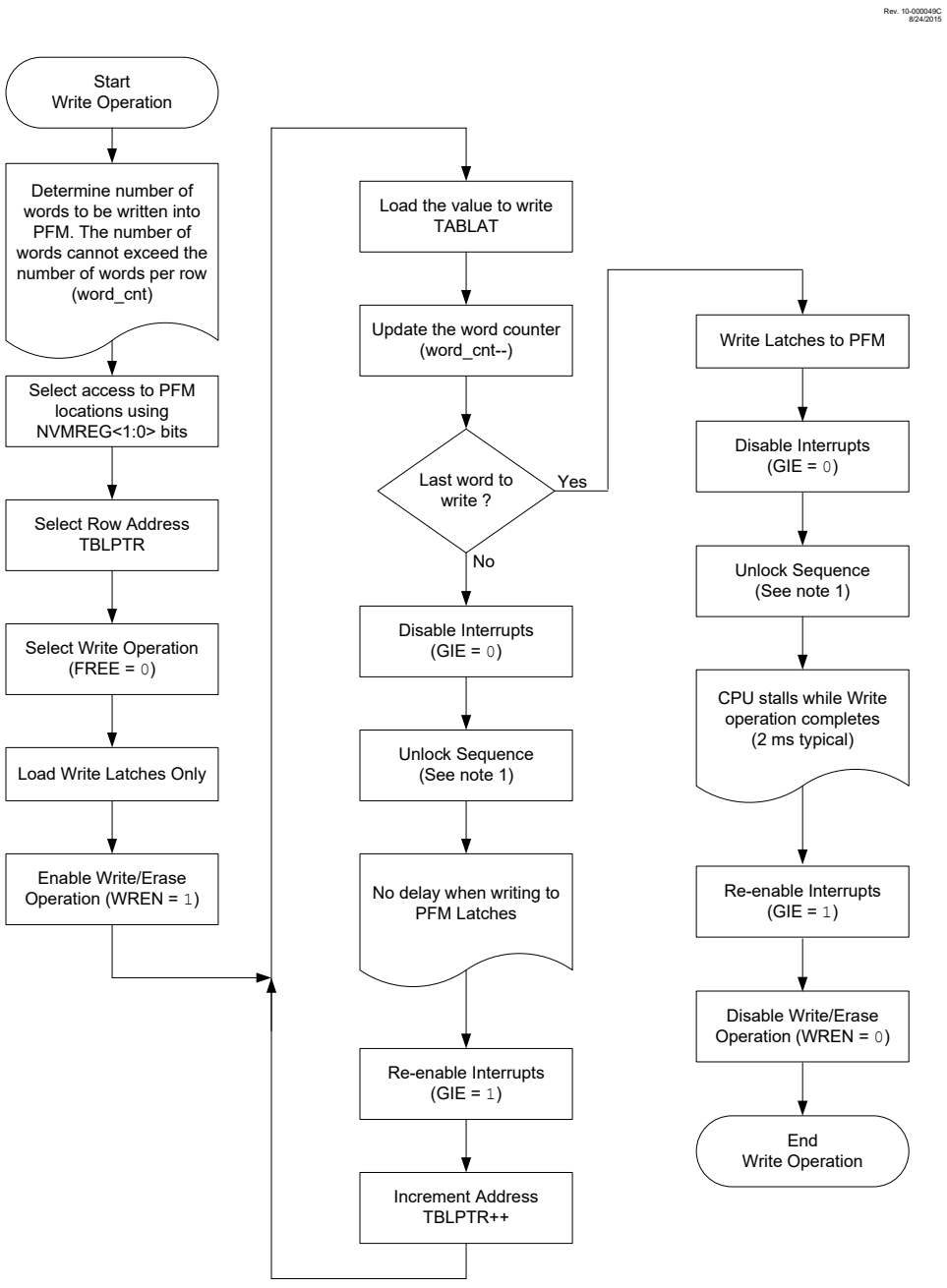
**Figure 9-9. FSCM Block Diagram**




### 9.4.1 Fail-Safe Detection

The FSCM module detects a failed oscillator by comparing the external oscillator to the FSCM sample clock. The sample clock is generated by dividing the LFINTOSC by 64. See Figure 9-9. Inside the fail detector block is a latch. The external clock sets the latch on each falling edge of the external clock. The sample clock clears the latch on each rising edge of the sample clock. A failure is detected when an entire half-cycle of the sample clock elapses before the external clock goes low.

**Figure 13-5. Program Flash Memory Flowchart**




**Note:**

1. See NVM Unlock Sequence Flowchart

**Example 13-4. Writing to Program Flash Memory**

```
   ; This write routine assumes the following:
   ; 1. 64 bytes of data are loaded, starting at the address in DATA_ADDR
   ; 2. Each word of data to be written is made up of two adjacent bytes in
DATA_ADDR,
   ; stored in little endian format
   ; 3. A valid starting address (the least significant bits = 00000) is
```

### 14.7.9 LATB

**Name:** LATB
**Address:** 0x019

Output Latch Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LATB7 | LATB6 | LATB5 | LATB4 | LATB3 | LATB2 | LATB1 | LATB0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | x | x | x | x | x | x | x | x |

**Bits 0, 1, 2, 3, 4, 5, 6, 7 – LATBn** Output Latch B Value bits
Reset States: POR/BOR = xxxxxxxx
All Other Resets = uuuuuuuuu

**Note:** Writes to LATB are equivalent with writes to the corresponding PORTB register. Reads from LATB register return register values, not I/O pin values.

| Input Signal Name | Input Register Name | Default Location at POR | Reset Value (xxxPPS<4:0>) | PORT From Which Input Is Available | | |
|---|---|---|---|---|---|---|
| SCK2 | SCL2PPS | RB1 | 0 1001 | — | B | C |
| SCL2 | SCL2PPS | RB1 | 0 1001 | — | B | C |
| SDI2 | SDA2PPS | RB2 | 0 1000 | — | B | C |
| SDA2 | SDA2PPS | RB2 | 0 1000 | — | B | C |
| SS2 | SS2PPS | RB2 | 0 1000 | — | B | C |
| RX1 | RX1PPS | RC7 | 1 0111 | — | B | C |
| DT1 | RX1PPS | RC7 | 1 0111 | — | B | C |
| TX1 | CK1PPS | RC6 | 1 0110 | — | B | C |
| CK1 | CK1PPS | RC6 | 1 0110 | — | B | C |
| RX2 | RX2PPS | RB7 | 0 1111 | — | B | C |
| DT2 | RX2PPS | RB7 | 0 1111 | — | B | C |
| TX2 | CK2PPS | RB6 | 0 1110 | — | B | C |
| CK2 | CK2PPS | RB6 | 0 1110 | — | B | C |
| SMT1SIG | SMT1SIGPPS | RC1 | 1 0001 | — | B | C |
| SMT1WIN | SMT1WINPPS | RC0 | 1 0000 | — | B | C |
| SMT2SIG | SMT2SIGPPS | RB5 | 0 1101 | — | B | C |
| SMT2WIN | SMT2WINPPS | RB4 | 0 1100 | — | B | C |

## 15.2 PPS Outputs

Each I/O pin has an RxyPPS register with which the pin output source is selected. With few exceptions, the port TRIS control associated with that pin retains control over the pin output driver. Peripherals that control the pin output driver as part of the peripheral operation will override the TRIS control as needed. These peripherals include:

- EUSART (synchronous operation)
- MSSP ($I^2C$)

Although every pin has its own RxyPPS peripheral selection register, the selections are identical for every pin as shown in the following table.

**Important:** The notation “Rxy” is a place holder for the pin identifier. The 'x' holds the place of the PORT letter and the 'y' holds the place of the bit number. For example, Rxy = RA0 for the RA0PPS register.

| Value | Description |
|---|---|
| 1 | CCP1 module disabled |
| | CCP1 module enabled |

## 17.5 Register Summary - Interrupt-on-Change

| Address | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x1F3D | IOCAP | 7:0 | IOCAP7 | IOCAP6 | IOCAP5 | IOCAP4 | IOCAP3 | IOCAP2 | IOCAP1 | IOCAP0 |
| 0x1F3E | IOCAN | 7:0 | IOCAN7 | IOCAN6 | IOCAN5 | IOCAN4 | IOCAN3 | IOCAN2 | IOCAN1 | IOCAN0 |
| 0x1F3F | IOCAF | 7:0 | IOCAF7 | IOCAF6 | IOCAF5 | IOCAF4 | IOCAF3 | IOCAF2 | IOCAF1 | IOCAF0 |
| 0x1F40 ... 0x1F47 | Reserved | | | | | | | | | |
| 0x1F48 | IOCBP | 7:0 | IOCBP7 | IOCBP6 | IOCBP5 | IOCBP4 | IOCBP3 | IOCBP2 | IOCBP1 | IOCBP0 |
| 0x1F49 | IOCBN | 7:0 | IOCBN7 | IOCBN6 | IOCBN5 | IOCBN4 | IOCBN3 | IOCBN2 | IOCBN1 | IOCBN0 |
| 0x1F4A | IOCBF | 7:0 | IOCBF7 | IOCBF6 | IOCBF5 | IOCBF4 | IOCBF3 | IOCBF2 | IOCBF1 | IOCBF0 |
| 0x1F4B ... 0x1F52 | Reserved | | | | | | | | | |
| 0x1F53 | IOCCP | 7:0 | IOCCP7 | IOCCP6 | IOCCP5 | IOCCP4 | IOCCP3 | IOCCP2 | IOCCP1 | IOCCP0 |
| 0x1F54 | IOCCN | 7:0 | IOCCN7 | IOCCN6 | IOCCN5 | IOCCN4 | IOCCN3 | IOCCN2 | IOCCN1 | IOCCN0 |
| 0x1F55 | IOCCF | 7:0 | IOCCF7 | IOCCF6 | IOCCF5 | IOCCF4 | IOCCF3 | IOCCF2 | IOCCF1 | IOCCF0 |
| 0x1F56 ... 0x1F68 | Reserved | | | | | | | | | |
| 0x1F69 | IOCEP | 7:0 | | | | | IOCEP3 | | | |
| 0x1F6A | IOCEN | 7:0 | | | | | IOCEN3 | | | |
| 0x1F6B | IOCEF | 7:0 | | | | | IOCEF3 | | | |

## 17.6 Register Definitions: Interrupt-on-Change Control

### 17.6.8 IOCEN

**Name:** IOCEN
**Address:** 0x1F6A

Interrupt-on-Change Negative Edge Register Example

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | IOCEN3 | | | |
| Access | | | | | R/W | | | |
| Reset | | | | | 0 | | | |

**Bit 3 – IOCEN3** Interrupt-on-Change Negative Edge Enable bits(1)

| Value | Description |
|---|---|
| 1 | Interrupt-on-Change enabled on the IOCA pin for a negative-going edge. Associated Status bit and interrupt flag will be set upon detecting an edge. |
| 0 | Interrupt-on-Change disabled for the associated pin |

**Note:**

1. If MCLRE = 1 or LVP = 1, RE3 port functionality is disabled and IOC on RE3 is not available.

### 26.14.2 TxGCON

**Name:** TxGCON
**Address:** 0x20F,0x215,0x21B

Timer Gate Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | GE | GPOL | GTM | GSPM | GGO/DONE | GVAL | | |
| Access | R/W | R/W | R/W | R/W | R/W | RO | | |
| Reset | 0 | 0 | 0 | 0 | 0 | x | | |

**Bit 7 – GE** Timer Gate Enable bit
Reset States: POR/BOR = 0
All Other Resets = u

| Value | Condition | Description |
|---|---|---|
| 1 | ON = 1 | Timer counting is controlled by the Timer gate function |
| 0 | ON = 1 | Timer is always counting |
| X | ON = 0 | This bit is ignored |

**Bit 6 – GPOL** Timer Gate Polarity bit
Reset States: POR/BOR = 0
All Other Resets = u

| Value | Description |
|---|---|
| 1 | Timer gate is active-high (Timer counts when gate is high) |
| 0 | Timer gate is active-low (Timer counts when gate is low) |

**Bit 5 – GTM** Timer Gate Toggle Mode bit
Timer Gate Flip-Flop Toggles on every rising edge

Reset States: POR/BOR = 0
All Other Resets = u

| Value | Description |
|---|---|
| 1 | Timer Gate Toggle mode is enabled |
| 0 | Timer Gate Toggle mode is disabled and Toggle flip-flop is cleared |

**Bit 4 – GSPM** Timer Gate Single Pulse Mode bit
Reset States: POR/BOR = 0
All Other Resets = u

| Value | Description |
|---|---|
| 1 | Timer Gate Single Pulse mode is enabled and is controlling Timer gate) |
| 0 | Timer Gate Single Pulse mode is disabled |

**Bit 3 – GGO/$\overline{\text{DONE}}$** Timer Gate Single Pulse Acquisition Status bit
This bit is automatically cleared when TxGSPM is cleared.

Reset States: POR/BOR = 0
All Other Resets = u

register then a one clock period wide pulse occurs on the TMR2_postscaled output, and the postscaler count is cleared.

### 27.1.2 One-Shot Mode

The One-Shot mode is identical to the Free Running Period mode except that the ON bit is cleared and the timer is stopped when T2TMR matches T2PR and will not restart until the ON bit is cycled off and on. Postscaler (OUTPS) values other than zero are ignored in this mode because the timer is stopped at the first period event and the postscaler is reset when the timer is restarted.

### 27.1.3 Monostable Mode

Monostable modes are similar to One-Shot modes except that the ON bit is not cleared and the timer can be restarted by an external Reset event.

## 27.2 Timer2 Output

The Timer2 module's primary output is TMR2_postscaled, which pulses for a single TMR2_clk period upon each match of the postscaler counter and the OUTPS bits of the T2CON register. The postscaler is incremented each time the T2TMR value matches the T2PR value. This signal can be selected as an input to several other input modules:

- The ADC module, as an auto-conversion trigger
- CWG, as an auto-shutdown source
- The CRC memory scanner, as a trigger for triggered mode
- Gate source for odd numbered timers (Timer1, Timer3, etc.)
- Alternate SPI clock
- Reset signals for other instances of even numbered timers (Timer2, Timer4, etc.)

In addition, the Timer2 is also used by the CCP module for pulse generation in PWM mode. See *"PWM Overview"* and *"Pulse-width Modulation"* sections for more details on setting up Timer2 for use with the CCP and PWM modules.

**Related Links**

29.4 PWM Overview

30. (PWM) Pulse-Width Modulation

## 27.3 External Reset Sources

In addition to the clock source, the Timer2 also takes in an external Reset source. This external Reset source is selected for each timer with the corresponding TxRST register. This source can control starting and stopping of the timer, as well as resetting the timer, depending on which mode the timer is in. Reset source selections are shown in the following table.

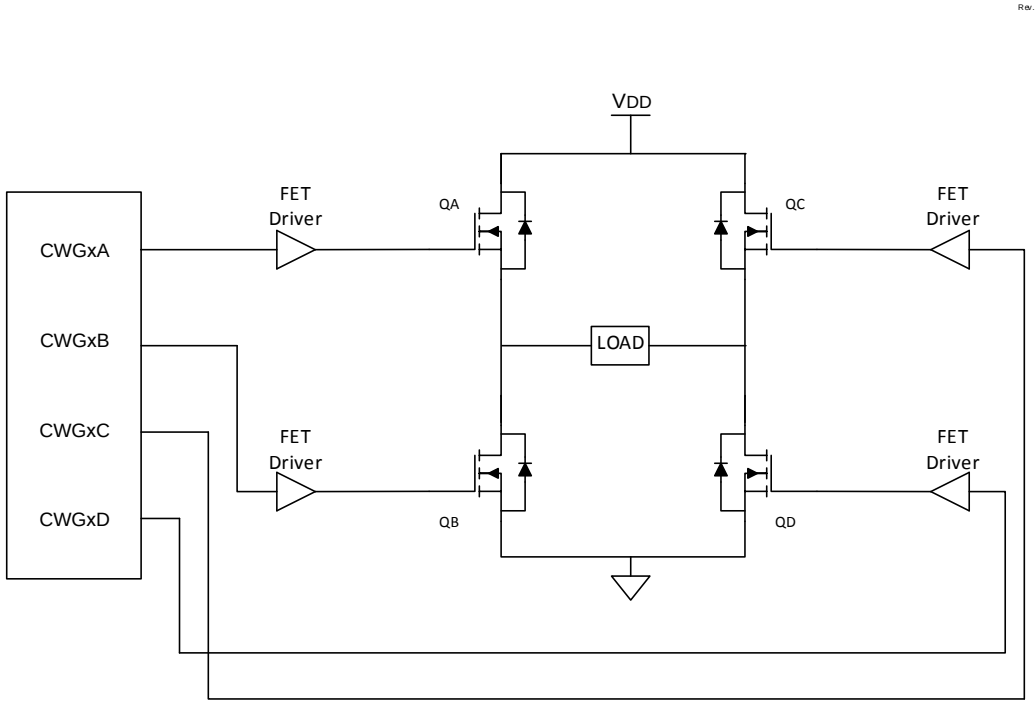
**Table 27-2. External Reset Sources**

| RSEL<3:0> | Reset Source | | |
|---|---|---|---|
| | TMR2 | TMR4 | TMR6 |
| 1111 | CCP5_out | CCP5_out | CCP5_out |
| 1101 | CLC4_out | CLC4_out | CLC4_out |

### 31.2.3 Full-Bridge Modes

In Forward and Reverse Full-Bridge modes, three outputs drive static values while the fourth is modulated by the input data signal. The mode selection may be toggled between forward and reverse by toggling the MODE<0> bit of the CWGxCON0 while keeping MODE<2:1> static, without disabling the CWG module. When connected, as shown in Figure 31-5, the outputs are appropriate for a full-bridge motor driver. Each CWG output signal has independent polarity control, so the circuit can be adapted to high-active and low-active drivers. A simplified block diagram for the Full-Bridge modes is shown in Figure 31-6.

**Figure 31-5. Example of Full-Bridge Application**



### 31.15.6 CWGxAS0

**Name:** CWGxAS0
**Address:** 0x612,0x61C,0x692

CWG Auto-Shutdown Control Register 0

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHUTDOWN | REN | LSBD[1:0] | | LSAC[1:0] | | | |
| Access | R/W/HS/HC | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 1 | 0 | 1 | | |

**Bit 7 – SHUTDOWN** Auto-Shutdown Event Status bit(1,2)

| Value | Description |
|---|---|
| 1 | An auto-shutdown state is in effect |
| 0 | No auto-shutdown event has occurred |

**Bit 6 – REN** Auto-Restart Enable bit

| Value | Description |
|---|---|
| 1 | Auto-restart is enabled |
| 0 | Auto-restart is disabled |

**Bits 5:4 – LSBD[1:0]** CWGxB and CWGxD Auto-Shutdown State Control bits

| Value | Description |
|---|---|
| 11 | A logic '1' is placed on CWGxB/D when an auto-shutdown event occurs. |
| 10 | A logic '0' is placed on CWGxB/D when an auto-shutdown event occurs. |
| 01 | Pin is tri-stated on CWGxB/D when an auto-shutdown event occurs. |
| 00 | The inactive state of the pin, including polarity, is placed on CWGxB/D after the required dead-band interval when an auto-shutdown event occurs. |

**Bits 3:2 – LSAC[1:0]** CWGxA and CWGxC Auto-Shutdown State Control bits

| Value | Description |
|---|---|
| 11 | A logic '1' is placed on CWGxA/C when an auto-shutdown event occurs. |
| 10 | A logic '0' is placed on CWGxA/C when an auto-shutdown event occurs. |
| 01 | Pin is tri-stated on CWGxA/C when an auto-shutdown event occurs. |
| 00 | The inactive state of the pin, including polarity, is placed on CWGxA/C after the required dead-band interval when an auto-shutdown event occurs. |

**Note:**

1. This bit may be written while EN = 0 (31.15.1 CWGxCON0), to place the outputs into the shutdown configuration.
2. The outputs will remain in auto-shutdown state until the next rising edge of the CWG data input after this bit is cleared.

This list describes the steps that need to be taken by slave software to use these options for $I^2C$ communication. Figure 35-16 displays a module using both address and data holding. Figure 35-17 includes the operation with the SEN bit of the SSPxCON2 register set.

1. S bit is set; SSPxIF is set if interrupt on Start detect is enabled.
2. Matching address with R/W bit clear is clocked in. SSPxIF is set and CKP cleared after the eighth falling edge of SCL.
3. Slave clears the SSPxIF.
4. Slave can look at the ACKTIM bit to determine if the SSPxIF was after or before the $\overline{ACK}$.
5. Slave reads the address value from SSPxBUF, clearing the BF flag.
6. Slave sets $\overline{ACK}$ value clocked out to the master by setting ACKDT.
7. Slave releases the clock by setting CKP.
8. SSPxIF is set after an $\overline{ACK}$, not after a NACK.
9. If SEN = 1, the slave hardware will stretch the clock after the $\overline{ACK}$.
10. Slave clears SSPxIF.

**Important:** SSPxIF is still set after the ninth falling edge of SCL even if there is no clock stretching and BF has been cleared. Only if NACK is sent to master is SSPxIF not set

11. SSPxIF set and CKP cleared after eighth falling edge of SCL for a received data byte.
12. Slave looks at ACKTIM bit to determine the source of the interrupt.
13. Slave reads the received data from SSPxBUF clearing BF.
14. Steps 7-14 are the same for each received data byte.
15. Communication is ended by either the slave sending an $\overline{ACK}$ = 1, or the master sending a Stop condition. If a Stop is sent and Interrupt on Stop Detect is disabled, the slave will only know by polling the P bit.

To terminate the auto-baud process before the RCxIF flag is set, clear the ABDEN bit then clear the ABDOVF bit of the BAUDxCON register. The ABDOVF bit will remain set if the ABDEN bit is not cleared first.

### 36.2.3 Auto-Wake-up on Break

During Sleep mode, all clocks to the EUSART are suspended. Because of this, the Baud Rate Generator is inactive and a proper character reception cannot be performed. The Auto-Wake-up feature allows the controller to wake-up due to activity on the RX/DT line. This feature is available only in Asynchronous mode.

The Auto-Wake-up feature is enabled by setting the WUE bit of the BAUDxCON register. Once set, the normal receive sequence on RX/DT is disabled, and the EUSART remains in an Idle state, monitoring for a wake-up event independent of the CPU mode. A wake-up event consists of a high-to-low transition on the RX/DT line. (This coincides with the start of a Sync Break or a wake-up signal character for the LIN protocol.)

The EUSART module generates an RCxIF interrupt coincident with the wake-up event. The interrupt is generated synchronously to the Q clocks in normal CPU operating modes as shown in Figure 36-7, and asynchronously if the device is in Sleep mode as shown in Figure 36-8. The interrupt condition is cleared by reading the RCxREG register.

The WUE bit is automatically cleared by the low-to-high transition on the RX line at the end of the Break. This signals to the user that the Break event is over. At this point, the EUSART module is in Idle mode waiting to receive the next character.

#### 36.2.3.1 Special Considerations

Break Character

To avoid character errors or character fragments during a wake-up event, the wake-up character must be all zeros.

When the wake-up is enabled the function works independent of the low time on the data stream. If the WUE bit is set and a valid non-zero character is received, the low time from the Start bit to the first rising edge will be interpreted as the wake-up event. The remaining bits in the character will be received as a fragmented character and subsequent characters can result in framing or overrun errors.

Therefore, the initial character in the transmission must be all ‘0’s. This must be ten or more bit times, 13-bit times recommended for LIN bus, or any number of bit times for standard RS-232 devices.

Oscillator Start-up Time

Oscillator start-up time must be considered, especially in applications using oscillators with longer start-up intervals (i.e., LP, XT or HS/PLL mode). The Sync Break (or wake-up signal) character must be of sufficient length, and be followed by a sufficient interval, to allow enough time for the selected oscillator to start and provide proper initialization of the EUSART.

WUE Bit

The wake-up event causes a receive interrupt by setting the RCxIF bit. The WUE bit is cleared in hardware by a rising edge on RX/DT. The interrupt condition is then cleared in software by reading the RCxREG register and discarding its contents.

To ensure that no actual data is lost, check the RCIDL bit to verify that a receive operation is not in process before setting the WUE bit. If a receive operation is not occurring, the WUE bit may then be set just prior to entering the Sleep mode.

| Address | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x059E | T0CON0 | 7:0 | T0EN | | T0OUT | T016BIT | T0OUTPS[3:0] | | | |
| 0x059F | T0CON1 | 7:0 | T0CS[2:0] | | | T0ASYNC | T0CKPS[3:0] | | | |
| 0x05A0 ... 0x05FF | Reserved | | | | | | | | | |
| 0x0600 | INDF0 | 7:0 | INDF0[7:0] | | | | | | | |
| 0x0601 | INDF1 | 7:0 | INDF1[7:0] | | | | | | | |
| 0x0602 | PCL | 7:0 | PCL[7:0] | | | | | | | |
| 0x0603 | STATUS | 7:0 | | | | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C |
| 0x0604 | FSR0 | 7:0 | FSRL[7:0] | | | | | | | |
| | | 15:8 | FSRH[7:0] | | | | | | | |
| 0x0606 | FSR1 | 7:0 | FSRL[7:0] | | | | | | | |
| | | 15:8 | FSRH[7:0] | | | | | | | |
| 0x0608 | BSR | 7:0 | | | BSR[5:0] | | | | | |
| 0x0609 | WREG | 7:0 | WREG[7:0] | | | | | | | |
| 0x060A | PCLATH | 7:0 | | PCLATH[6:0] | | | | | | |
| 0x060B | INTCON | 7:0 | GIE | PEIE | | | | | | INTEDG |
| 0x060C | CWG1CLK | 7:0 | | | | | | | | CS |
| 0x060D | CWG1ISM | 7:0 | | | | | ISM[3:0] | | | |
| 0x060E | CWG1DBR | 7:0 | | | DBR[5:0] | | | | | |
| 0x060F | CWG1DBF | 7:0 | | | DBF[5:0] | | | | | |
| 0x0610 | CWG1CON0 | 7:0 | EN | LD | | | | MODE[2:0] | | |
| 0x0611 | CWG1CON1 | 7:0 | | | IN | | POLD | POLC | POLB | POLA |
| 0x0612 | CWG1AS0 | 7:0 | SHUTDOWN | REN | LSBD[1:0] | | LSAC[1:0] | | | |
| 0x0613 | CWG1AS1 | 7:0 | | | AS5E | AS4E | AS3E | AS2E | AS1E | AS0E |
| 0x0614 | CWG1STR | 7:0 | OVRD | OVRC | OVRB | OVRA | STRD | STRC | STRB | STRA |
| 0x0615 | Reserved | | | | | | | | | |
| 0x0616 | CWG2CLK | 7:0 | | | | | | | | CS |
| 0x0617 | CWG2ISM | 7:0 | | | | | ISM[3:0] | | | |
| 0x0618 | CWG2DBR | 7:0 | | | DBR[5:0] | | | | | |
| 0x0619 | CWG2DBF | 7:0 | | | DBF[5:0] | | | | | |
| 0x061A | CWG2CON0 | 7:0 | EN | LD | | | | MODE[2:0] | | |
| 0x061B | CWG2CON1 | 7:0 | | | IN | | POLD | POLC | POLB | POLA |
| 0x061C | CWG2AS0 | 7:0 | SHUTDOWN | REN | LSBD[1:0] | | LSAC[1:0] | | | |
| 0x061D | CWG2AS1 | 7:0 | | | AS5E | AS4E | AS3E | AS2E | AS1E | AS0E |
| 0x061E | CWG2STR | 7:0 | OVRD | OVRC | OVRB | OVRA | STRD | STRC | STRB | STRA |
| 0x061F ... 0x067F | Reserved | | | | | | | | | |
| 0x0680 | INDF0 | 7:0 | INDF0[7:0] | | | | | | | |
| 0x0681 | INDF1 | 7:0 | INDF1[7:0] | | | | | | | |
| 0x0682 | PCL | 7:0 | PCL[7:0] | | | | | | | |
| 0x0683 | STATUS | 7:0 | | | | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C |
| 0x0684 | FSR0 | 7:0 | FSRL[7:0] | | | | | | | |
| | | 15:8 | FSRH[7:0] | | | | | | | |
| 0x0686 | FSR1 | 7:0 | FSRL[7:0] | | | | | | | |

| IORWF | Inclusive OR W with f |
|---|---|
| Status Affected: | Z |
| Description: | Inclusive OR the W register with register ‘f’.<br>If ‘d’ is ‘0’, the result is placed in the W register.<br>If ‘d’ is ‘1’, the result is placed back in register ‘f’. |

| LSLF | Logical Left Shift |
|---|---|
| Syntax: | [ *label* ] LSLF f {,d} |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | (f<7>) → C<br>(f<6:0>) → dest<7:1><br>0 → dest<0> |
| Status Affected: | C, Z |
| Description: | The contents of register ‘f’ are shifted one bit to the left through the Carry flag.<br>A ‘0’ is shifted into the LSb.<br>If ‘d’ is ‘0’, the result is placed in W.<br>If ‘d’ is ‘1’, the result is stored back in register ‘f’.<br>**C ← Register f ← 0** |

| LSRF | Logical Right Shift |
|---|---|
| Syntax: | [ *label* ] LSRF f {,d} |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | 0 → dest<7><br>(f<7:1>) → dest<6:0>,<br>(f<0>) → C |
| Status Affected: | C, Z |
| Description: | The contents of register ‘f’ are shifted one bit to the right through the Carry flag.<br>A ‘0’ is shifted into the MSb.<br>If ‘d’ is ‘0’, the result is placed in W.<br>If ‘d’ is ‘1’, the result is stored back in register ‘f’.<br>**0 → register f → C** |

### 42.4.6 Temperature Indicator Requirements

**Table 42-12.**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Param. No.** | **Sym.** | **Characteristic** | | **Min.** | | **Max.** | **Units** | **Conditions** |
| TS01 | $T_{ACQMIN}$ | Minimum ADC Acquisition Time Delay | | — | 25 | — | µs | |
| TS02 | Mv | Voltage Sensitivity | High Range | — | -3.684 | — | mV/°C | TSRNG = 1 |
| | | | Low Range | — | -3.456 | — | mV/°C | TSRNG = 0 |

* - These parameters are characterized but not tested.

† Data in “Typ” column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

### 42.4.7 Analog-To-Digital Converter (ADC) Accuracy Specifications(1,2)

**Table 42-13.**

| Standard Operating Conditions (unless otherwise stated)<br>$V_{DD}$ = 3.0V, $T_A$ = 25°C, $T_{AD}$ = 1µs | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ. †** | **Max.** | **Units** | **Conditions** |
| AD01 | $N_R$ | Resolution | — | — | 12 | bit | |
| AD02 | $E_{IL}$ | Integral Error | — | ±0.2 | ±1.0 | Lsb | $ADC_{REF}$+=3.0V, $ADC_{REF}$- = 0V |
| AD03 | $E_{DL}$ | Differential Error | — | ±1.0 | ±1.0 | Lsb | $ADC_{REF}$+=3.0V, $ADC_{REF}$- = 0V |
| AD04 | $E_{OFF}$ | Offset Error | — | 0.5 | 6.5 | Lsb | $ADC_{REF}$+=3.0V, $ADC_{REF}$- = 0V |
| AD05 | $E_{GN}$ | Gain Error | — | ±0.2 | ±6.0 | Lsb | $ADC_{REF}$+=3.0V, $ADC_{REF}$- = 0V |
| AD06 | $V_{ADREF}$ | ADC Reference Voltage ($AD_{REF}$+ - $AD_{REF}$-) | 1.8 | — | $V_{DD}$ | V | |
| AD07 | $V_{AIN}$ | Full-Scale Range | $AD_{REF}$- | — | $AD_{REF}$+ | V | |
| AD08 | $Z_{AIN}$ | Recommended Impedance of Analog Voltage Source | — | 10 | — | kΩ | |
| AD09 | $R_{VREF}$ | ADC Voltage Reference Ladder Impedance | — | 50 | — | kΩ | |

* - These parameters are characterized but not tested.