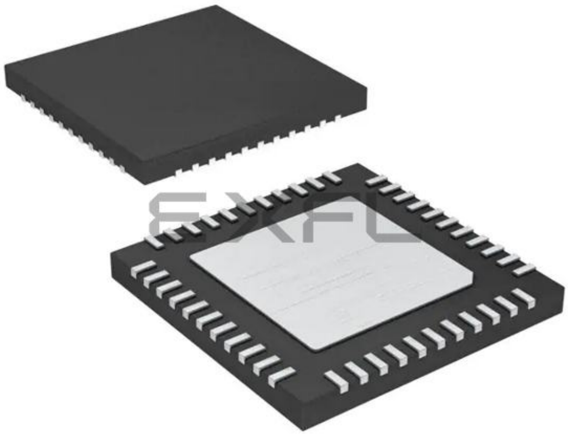


Welcome to E-XFL.COM

## 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 | 40MHz |
| Connectivity | I²C, SPI, UART/USART |
| Peripherals | Brown-out Detect/Reset, HLVD, POR, PWM, WDT |
| Number of I/O | 36 |
| Program Memory Size | 16KB (8K x 16) |
| Program Memory Type | FLASH |
| EEPROM Size | 256 x 8 |
| RAM Size | 768 x 8 |
| Voltage - Supply (Vcc/Vdd) | 4.2V ~ 5.5V |
| Data Converters | A/D 13x10b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 85°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 44-VQFN Exposed Pad |
| Supplier Device Package | 44-QFN (8x8) |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/pic18f4420t-i-ml |

**TABLE 1-2: PIC18F2420/2520 PINOUT I/O DESCRIPTIONS (CONTINUED)**

| Pin Name | Pin Number SPDIP, SOIC | Pin Number QFN | Pin Type | Buffer Type | Description |
|---|---|---|---|---|---|
| | | | | | PORTB is a bidirectional I/O port. PORTB can be software programmed for internal weak pull-ups on all inputs. |
| RB0/INT0/FLT0/AN12 | 21 | 18 | | | |
| RB0 | | | I/O | TTL | Digital I/O. |
| INT0 | | | I | ST | External interrupt 0. |
| FLT0 | | | I | ST | PWM Fault input for CCP1. |
| AN12 | | | I | Analog | Analog input 12. |
| RB1/INT1/AN10 | 22 | 19 | | | |
| RB1 | | | I/O | TTL | Digital I/O. |
| INT1 | | | I | ST | External interrupt 1. |
| AN10 | | | I | Analog | Analog input 10. |
| RB2/INT2/AN8 | 23 | 20 | | | |
| RB2 | | | I/O | TTL | Digital I/O. |
| INT2 | | | I | ST | External interrupt 2. |
| AN8 | | | I | Analog | Analog input 8. |
| RB3/AN9/CCP2 | 24 | 21 | | | |
| RB3 | | | I/O | TTL | Digital I/O. |
| AN9 | | | I | Analog | Analog input 9. |
| CCP2[(1)] | | | I/O | ST | Capture 2 input/Compare 2 output/PWM2 output. |
| RB4/KBI0/AN11 | 25 | 22 | | | |
| RB4 | | | I/O | TTL | Digital I/O. |
| KBI0 | | | I | TTL | Interrupt-on-change pin. |
| AN11 | | | I | Analog | Analog input 11. |
| RB5/KBI1/PGM | 26 | 23 | | | |
| RB5 | | | I/O | TTL | Digital I/O. |
| KBI1 | | | I | TTL | Interrupt-on-change pin. |
| PGM | | | I/O | ST | Low-Voltage ICSP™ Programming enable pin. |
| RB6/KBI2/PGC | 27 | 24 | | | |
| RB6 | | | I/O | TTL | Digital I/O. |
| KBI2 | | | I | TTL | Interrupt-on-change pin. |
| PGC | | | I/O | ST | In-Circuit Debugger and ICSP programming clock pin. |
| RB7/KBI3/PGD | 28 | 25 | | | |
| RB7 | | | I/O | TTL | Digital I/O. |
| KBI3 | | | I | TTL | Interrupt-on-change pin. |
| PGD | | | I/O | ST | In-Circuit Debugger and ICSP programming data pin. |

**Legend:** TTL = TTL compatible input; CMOS = CMOS compatible input or output
ST = Schmitt Trigger input with CMOS levels; I = Input
O = Output; P = Power

**Note 1:** Default assignment for CCP2 when Configuration bit, CCP2MX, is set.
**2:** Alternate assignment for CCP2 when Configuration bit, CCP2MX, is cleared.

**TABLE 1-2: PIC18F2420/2520 PINOUT I/O DESCRIPTIONS (CONTINUED)**

| Pin Name | Pin Number | | Pin Type | Buffer Type | Description |
|---|---|---|---|---|---|
| | SPDIP, SOIC | QFN | | | |
| | | | | | PORTC is a bidirectional I/O port. |
| RC0/T1OSO/T13CKI | 11 | 8 | | | |
| RC0 | | | I/O | ST | Digital I/O. |
| T1OSO | | | O | — | Timer1 oscillator output. |
| T13CKI | | | I | ST | Timer1/Timer3 external clock input. |
| RC1/T1OSI/CCP2 | 12 | 9 | | | |
| RC1 | | | I/O | ST | Digital I/O. |
| T1OSI | | | I | Analog | Timer1 oscillator input. |
| CCP2(2) | | | I/O | ST | Capture 2 input/Compare 2 output/PWM2 output. |
| RC2/CCP1 | 13 | 10 | | | |
| RC2 | | | I/O | ST | Digital I/O. |
| CCP1 | | | I/O | ST | Capture 1 input/Compare 1 output/PWM1 output. |
| RC3/SCK/SCL | 14 | 11 | | | |
| RC3 | | | I/O | ST | Digital I/O. |
| SCK | | | I/O | ST | Synchronous serial clock input/output for SPI mode. |
| SCL | | | I/O | ST | Synchronous serial clock input/output for $I^2C$™ mode. |
| RC4/SDI/SDA | 15 | 12 | | | |
| RC4 | | | I/O | ST | Digital I/O. |
| SDI | | | I | ST | SPI data in. |
| SDA | | | I/O | ST | $I^2C$ data I/O. |
| RC5/SDO | 16 | 13 | | | |
| RC5 | | | I/O | ST | Digital I/O. |
| SDO | | | O | — | SPI data out. |
| RC6/TX/CK | 17 | 14 | | | |
| RC6 | | | I/O | ST | Digital I/O. |
| TX | | | O | — | EUSART asynchronous transmit. |
| CK | | | I/O | ST | EUSART synchronous clock (see related RX/DT). |
| RC7/RX/DT | 18 | 15 | | | |
| RC7 | | | I/O | ST | Digital I/O. |
| RX | | | I | ST | EUSART asynchronous receive. |
| DT | | | I/O | ST | EUSART synchronous data (see related TX/CK). |
| RE3 | — | — | — | — | See $\overline{MCLR}$/VPP/RE3 pin. |
| VSS | 8, 19 | 5, 16 | P | — | Ground reference for logic and I/O pins. |
| VDD | 20 | 17 | P | — | Positive supply for logic and I/O pins. |

**Legend:** TTL = TTL compatible input; CMOS = CMOS compatible input or output
ST = Schmitt Trigger input with CMOS levels; I = Input
O = Output; P = Power

**Note 1:** Default assignment for CCP2 when Configuration bit, CCP2MX, is set.
**2:** Alternate assignment for CCP2 when Configuration bit, CCP2MX, is cleared.

**FIGURE 4-6: SLOW RISE TIME (MCLR TIED TO VDD, VDD RISE > TPWRT)**

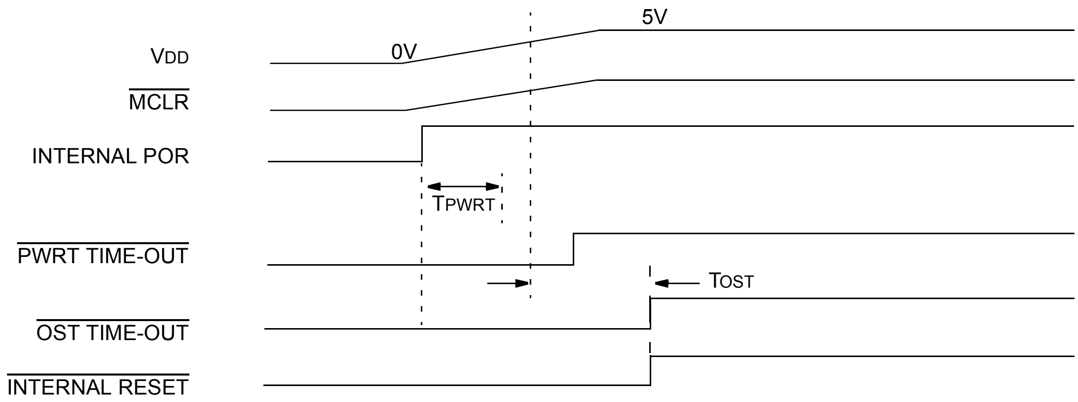
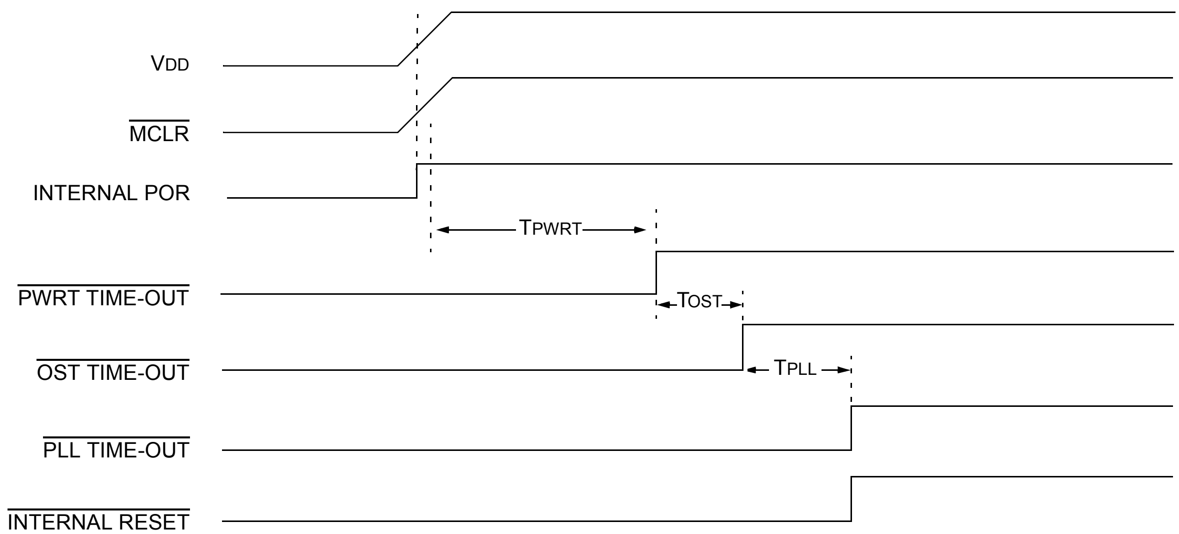
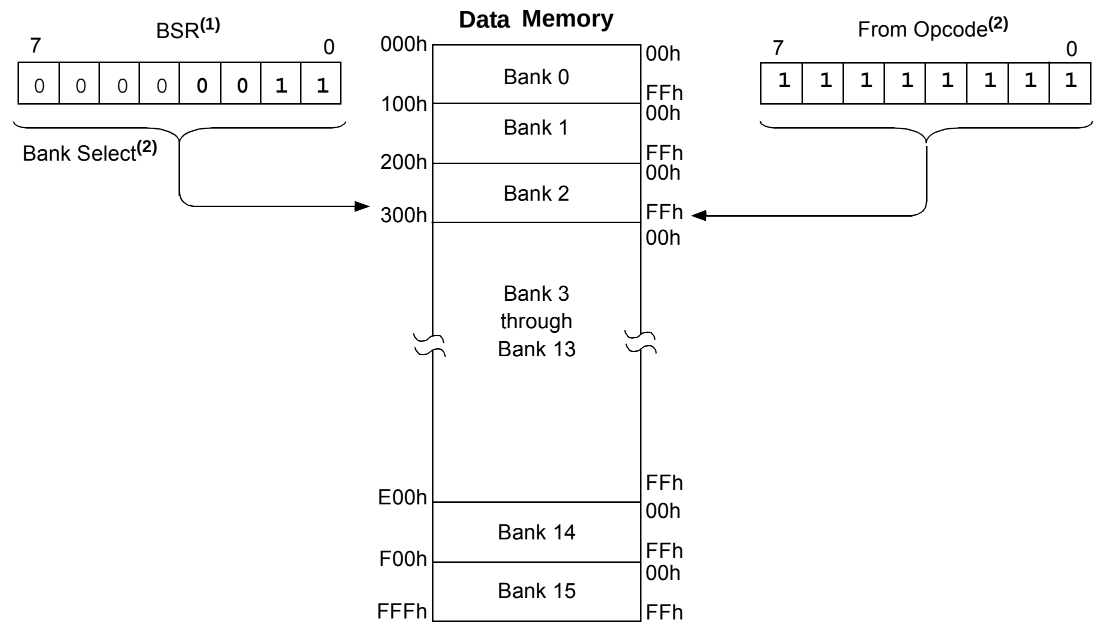



**FIGURE 4-7: TIME-OUT SEQUENCE ON POR W/PLL ENABLED (MCLR TIED TO VDD)**




**Note:** TOST = 1024 clock cycles.
TPLL ≈ 2 ms max. First three stages of the PWRT timer.

**FIGURE 5-7: USE OF THE BANK SELECT REGISTER (DIRECT ADDRESSING)**




**Note 1:** The Access RAM bit of the instruction can be used to force an override of the selected bank (BSR<3:0>) to the registers of the Access Bank.

**2:** The `MOVFF` instruction embeds the entire 12-bit address in the instruction.

### 5.3.2 ACCESS BANK

While the use of the BSR with an embedded 8-bit address allows users to address the entire range of data memory, it also means that the user must always ensure that the correct bank is selected. Otherwise, data may be read from or written to the wrong location. This can be disastrous if a GPR is the intended target of an operation, but an SFR is written to instead. Verifying and/or changing the BSR for each read or write to data memory can become very inefficient.

To streamline access for the most commonly used data memory locations, the data memory is configured with an Access Bank, which allows users to access a mapped block of memory without specifying a BSR. The Access Bank consists of the first 128 bytes of memory (00h-7Fh) in Bank 0 and the last 128 bytes of memory (80h-FFh) in Block 15. The lower half is known as the “Access RAM” and is composed of GPRs. This upper half is also where the device’s SFRs are mapped. These two areas are mapped contiguously in the Access Bank and can be addressed in a linear fashion by an 8-bit address (Figure 5-5).

The Access Bank is used by core PIC18 instructions that include the Access RAM bit (the ‘a’ parameter in the instruction). When ‘a’ is equal to ‘`1`’, the instruction uses the BSR and the 8-bit address included in the opcode for the data memory address. When ‘a’ is ‘`0`’, however, the instruction is forced to use the Access Bank address map; the current value of the BSR is ignored entirely.

Using this “forced” addressing allows the instruction to operate on a data address in a single cycle, without updating the BSR first. For 8-bit addresses of 80h and above, this means that users can evaluate and operate on SFRs more efficiently. The Access RAM below 80h is a good place for data values that the user might need to access rapidly, such as immediate computational results or common program variables. Access RAM also allows for faster and more code efficient context saving and switching of variables.

The mapping of the Access Bank is slightly different when the extended instruction set is enabled (XINST Configuration bit = `1`). This is discussed in more detail in **Section 5.5.3 “Mapping the Access Bank in Indexed Literal Offset Mode”**.

### 5.3.3 GENERAL PURPOSE REGISTER FILE

PIC18 devices may have banked memory in the GPR area. This is data RAM, which is available for use by all instructions. GPRs start at the bottom of Bank 0 (address 000h) and grow upwards towards the bottom of the SFR area. GPRs are not initialized by a Power-on Reset and are unchanged on all other Resets.

### TABLE 5-2: PIC18F2420/2520/4420/4520 REGISTER FILE SUMMARY (CONTINUED)

| File Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Details on page: |
|---|---|---|---|---|---|---|---|---|---|---|
| SPBRGH | EUSART Baud Rate Generator Register High Byte | | | | | | | | 0000 0000 | 51, 206 |
| SPBRG | EUSART Baud Rate Generator Register Low Byte | | | | | | | | 0000 0000 | 51, 206 |
| RCREG | EUSART Receive Register | | | | | | | | 0000 0000 | 51, 213 |
| TXREG | EUSART Transmit Register | | | | | | | | 0000 0000 | 51, 211 |
| TXSTA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 0000 0010 | 51, 202 |
| RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 0000 000x | 51, 203 |
| EEADR | EEPROM Address Register | | | | | | | | 0000 0000 | 51, 74, 83 |
| EEDATA | EEPROM Data Register | | | | | | | | 0000 0000 | 51, 74, 83 |
| EECON2 | EEPROM Control Register 2 (not a physical register) | | | | | | | | 0000 0000 | 51, 74, 83 |
| EECON1 | EEPGD | CFGS | — | FREE | WRERR | WREN | WR | RD | xx-0 x000 | 51, 75, 84 |
| IPR2 | OSCFIP | CMIP | — | EEIP | BCLIP | HLVDIP | TMR3IP | CCP2IP | 11-1 1111 | 52, 101 |
| PIR2 | OSCFIF | CMIF | — | EEIF | BCLIF | HLVDIF | TMR3IF | CCP2IF | 00-0 0000 | 52, 97 |
| PIE2 | OSCFIE | CMIE | — | EEIE | BCLIE | HLVDIE | TMR3IE | CCP2IE | 00-0 0000 | 52, 99 |
| IPR1 | PSPIP(2) | ADIP | RCIP | TXIP | SSPIP | CCP1IP | TMR2IP | TMR1IP | 1111 1111 | 52, 100 |
| PIR1 | PSPIF(2) | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 0000 0000 | 52, 96 |
| PIE1 | PSPIE(2) | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 0000 0000 | 52, 98 |
| OSCTUNE | INTSRC | PLLEN(3) | — | TUN4 | TUN3 | TUN2 | TUN1 | TUN0 | 0q-0 0000 | 27, 52 |
| TRISE(2) | IBF | OBF | IBOV | PSPMODE | — | TRISE2 | TRISE1 | TRISE0 | 0000 -111 | 52, 118 |
| TRISD(2) | PORTD Data Direction Register | | | | | | | | 1111 1111 | 52, 114 |
| TRISC | PORTC Data Direction Register | | | | | | | | 1111 1111 | 52, 111 |
| TRISB | PORTB Data Direction Register | | | | | | | | 1111 1111 | 52, 108 |
| TRISA | TRISA7(5) | TRISA6(5) | PORTA Data Direction Register | | | | | | 1111 1111 | 52, 105 |
| LATE(2) | — | — | — | — | — | PORTE Data Latch Register (Read and Write to Data Latch) | | | ---- -xxx | 52, 117 |
| LATD(2) | PORTD Data Latch Register (Read and Write to Data Latch) | | | | | | | | xxxx xxxx | 52, 114 |
| LATC | PORTC Data Latch Register (Read and Write to Data Latch) | | | | | | | | xxxx xxxx | 52, 111 |
| LATB | PORTB Data Latch Register (Read and Write to Data Latch) | | | | | | | | xxxx xxxx | 52, 108 |
| LATA | LATA7(5) | LATA6(5) | PORTA Data Latch Register (Read and Write to Data Latch) | | | | | | xxxx xxxx | 52, 105 |
| PORTE | — | — | — | — | RE3(4) | RE2(2) | RE1(2) | RE0(2) | ---- xxxx | 52, 117 |
| PORTD(2) | RD7 | RD6 | RD5 | RD4 | RD3 | RD2 | RD1 | RD0 | xxxx xxxx | 52, 114 |
| PORTC | RC7 | RC6 | RC5 | RC4 | RC3 | RC2 | RC1 | RC0 | xxxx xxxx | 52, 111 |
| PORTB | RB7 | RB6 | RB5 | RB4 | RB3 | RB2 | RB1 | RB0 | xxxx xxxx | 52, 108 |
| PORTA | RA7(5) | RA6(5) | RA5 | RA4 | RA3 | RA2 | RA1 | RA0 | xx0x 0000 | 52, 105 |

**Legend:** x = unknown, u = unchanged, – = unimplemented, q = value depends on condition. Shaded cells are unimplemented, read as '0'.

**Note 1:** The SBOREN bit is only available when the BOREN<1:0> Configuration bits = 01; otherwise, it is disabled and reads as '0'. See **Section 4.4 "Brown-out Reset (BOR)"**.

**2:** These registers and/or bits are not implemented on 28-pin devices and are read as '0'. Reset values are shown for 40/44-pin devices; individual unimplemented bits should be interpreted as '-'.

**3:** The PLLEN bit is only available in specific oscillator configurations; otherwise, it is disabled and reads as '0'. See **Section 2.6.4 "PLL in INTOSC Modes"**.

**4:** The RE3 bit is only available when Master Clear Reset is disabled (MCLRE Configuration bit = 0); otherwise, RE3 reads as '0'. This bit is read-only.

**5:** RA6/RA7 and their associated latch and direction bits are individually configured as port pins based on various primary oscillator modes. When disabled, these bits read as '0'.

#### 5.4.3.1 FSR Registers and the INDF Operand

At the core of Indirect Addressing are three sets of registers: FSR0, FSR1 and FSR2. Each represents a pair of 8-bit registers, FSRnH and FSRnL. The four upper bits of the FSRnH register are not used so each FSR pair holds a 12-bit value. This represents a value that can address the entire range of the data memory in a linear fashion. The FSR register pairs, then, serve as pointers to data memory locations.

Indirect Addressing is accomplished with a set of Indirect File Operands, INDF0 through INDF2. These can be thought of as “virtual” registers: they are mapped in the SFR space but are not physically implemented. Reading or writing to a particular INDF register actually accesses its corresponding FSR register pair. A read from INDF1, for example, reads the data at the address indicated by FSR1H:FSR1L. Instructions that use the INDF registers as operands actually use the contents of their corresponding FSR as a pointer to the instruction’s target. The INDF operand is just a convenient way of using the pointer.

Because Indirect Addressing uses a full 12-bit address, data RAM banking is not necessary. Thus, the current contents of the BSR and the Access RAM bit have no effect on determining the target address.

#### 5.4.3.2 FSR Registers and POSTINC, POSTDEC, PREINC and PLUSW

In addition to the INDF operand, each FSR register pair also has four additional indirect operands. Like INDF, these are “virtual” registers that cannot be indirectly read or written to. Accessing these registers actually accesses the associated FSR register pair, but also performs a specific action on it stored value. They are:

- POSTDEC: accesses the FSR value, then automatically decrements it by 1 afterwards
- POSTINC: accesses the FSR value, then automatically increments it by 1 afterwards
- PREINC: increments the FSR value by 1, then uses it in the operation
- PLUSW: adds the signed value of the W register (range of -127 to 128) to that of the FSR and uses the new value in the operation.

In this context, accessing an INDF register uses the value in the FSR registers without changing them. Similarly, accessing a PLUSW register gives the FSR value offset by that in the W register; neither value is actually changed in the operation. Accessing the other virtual registers changes the value of the FSR registers.

Operations on the FSRs with POSTDEC, POSTINC and PREINC affect the entire register pair; that is, rollovers of the FSRnL register from FFh to 00h carry over to the FSRnH register. On the other hand, results of these operations do not change the value of any flags in the STATUS register (e.g., Z, N, OV, etc.).

**FIGURE 5-8: INDIRECT ADDRESSING**

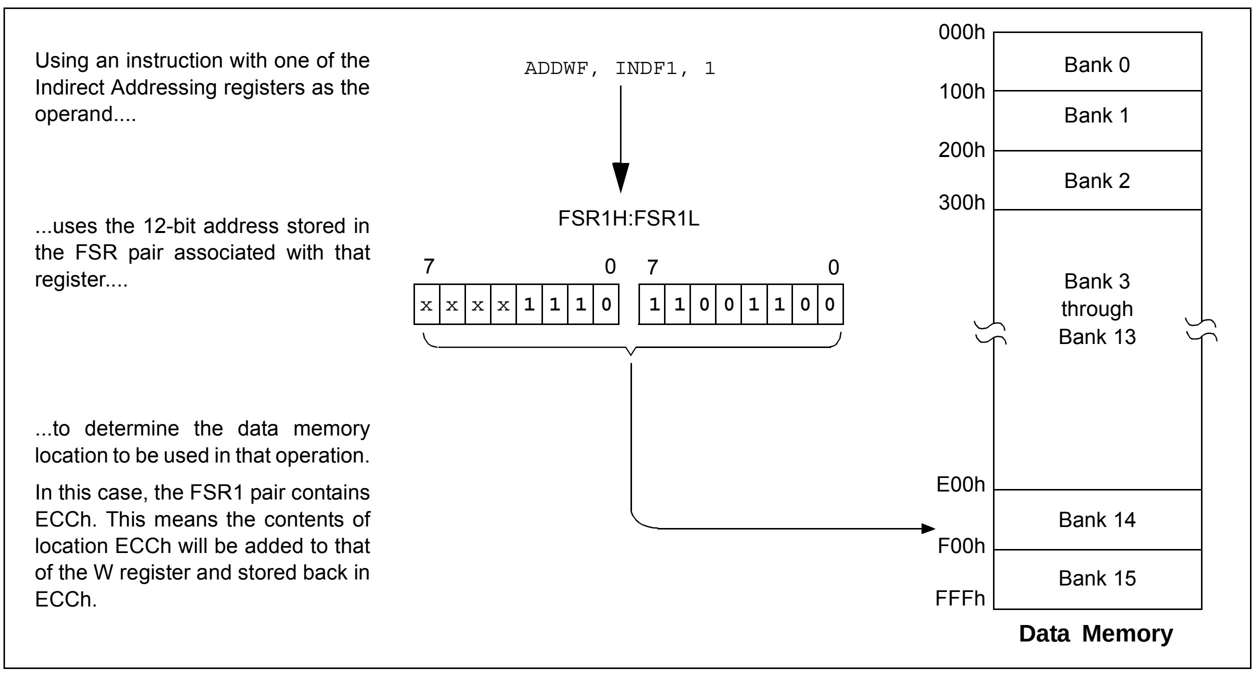


**TABLE 10-7: PORTD I/O SUMMARY**

| Pin | Function | TRIS Setting | I/O | I/O Type | Description |
|---|---|---|---|---|---|
| RD0/PSP0 | RD0 | 0 | O | DIG | LATD<0> data output. |
| | | 1 | I | ST | PORTD<0> data input. |
| | PSP0 | x | O | DIG | PSP read data output (LATD<0>); takes priority over port data. |
| | | x | I | TTL | PSP write data input. |
| RD1/PSP1 | RD1 | 0 | O | DIG | LATD<1> data output. |
| | | 1 | I | ST | PORTD<1> data input. |
| | PSP1 | x | O | DIG | PSP read data output (LATD<1>); takes priority over port data. |
| | | x | I | TTL | PSP write data input. |
| RD2/PSP2 | RD2 | 0 | O | DIG | LATD<2> data output. |
| | | 1 | I | ST | PORTD<2> data input. |
| | PSP2 | x | O | DIG | PSP read data output (LATD<2>); takes priority over port data. |
| | | x | I | TTL | PSP write data input. |
| RD3/PSP3 | RD3 | 0 | O | DIG | LATD<3> data output. |
| | | 1 | I | ST | PORTD<3> data input. |
| | PSP3 | x | O | DIG | PSP read data output (LATD<3>); takes priority over port data. |
| | | x | I | TTL | PSP write data input. |
| RD4/PSP4 | RD4 | 0 | O | DIG | LATD<4> data output. |
| | | 1 | I | ST | PORTD<4> data input. |
| | PSP4 | x | O | DIG | PSP read data output (LATD<4>); takes priority over port data. |
| | | x | I | TTL | PSP write data input. |
| RD5/PSP5/P1B | RD5 | 0 | O | DIG | LATD<5> data output. |
| | | 1 | I | ST | PORTD<5> data input. |
| | PSP5 | x | O | DIG | PSP read data output (LATD<5>); takes priority over port data. |
| | | x | I | TTL | PSP write data input. |
| | P1B | 0 | O | DIG | ECCP1 Enhanced PWM output, channel B; takes priority over port and PSP data. May be configured for tri-state during Enhanced PWM shutdown events. |
| RD6/PSP6/P1C | RD6 | 0 | O | DIG | LATD<6> data output. |
| | | 1 | I | ST | PORTD<6> data input. |
| | PSP6 | x | O | DIG | PSP read data output (LATD<6>); takes priority over port data. |
| | | x | I | TTL | PSP write data input. |
| | P1C | 0 | O | DIG | ECCP1 Enhanced PWM output, channel C; takes priority over port and PSP data. May be configured for tri-state during Enhanced PWM shutdown events. |
| RD7/PSP7/P1D | RD7 | 0 | O | DIG | LATD<7> data output. |
| | | 1 | I | ST | PORTD<7> data input. |
| | PSP7 | x | O | DIG | PSP read data output (LATD<7>); takes priority over port data. |
| | | x | I | TTL | PSP write data input. |
| | P1D | 0 | O | DIG | ECCP1 Enhanced PWM output, channel D; takes priority over port and PSP data. May be configured for tri-state during Enhanced PWM shutdown events. |

**Legend:** DIG = Digital level output; TTL = TTL input buffer; ST = Schmitt Trigger input buffer; x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

## 15.3 Compare Mode

In Compare mode, the 16-bit CCPRx register value is constantly compared against either the TMR1 or TMR3 register pair value. When a match occurs, the CCPx pin can be:

- driven high
- driven low
- toggled (high-to-low or low-to-high)
- remain unchanged (that is, reflects the state of the I/O latch)

The action on the pin is based on the value of the mode select bits (CCPxM<3:0>). At the same time, the interrupt flag bit, CCPxIF, is set.

### 15.3.1 CCP PIN CONFIGURATION

The user must configure the CCPx pin as an output by clearing the appropriate TRIS bit.

> **Note:** Clearing the CCP2CON register will force the RB3 or RC1 compare output latch (depending on device configuration) to the default low level. This is not the PORTB or PORTC I/O data latch.

### 15.3.2 TIMER1/TIMER3 MODE SELECTION

Timer1 and/or Timer3 must be running in Timer mode or Synchronized Counter mode if the CCP module is using the compare feature. In Asynchronous Counter mode, the compare operation may not work.

### 15.3.3 SOFTWARE INTERRUPT MODE

When the Generate Software Interrupt mode is chosen (CCPxM<3:0> = `1010`), the corresponding CCPx pin is not affected. A CCP interrupt is generated when the CCPxIF interrupt flag is set while the CCPxIE bit is set.

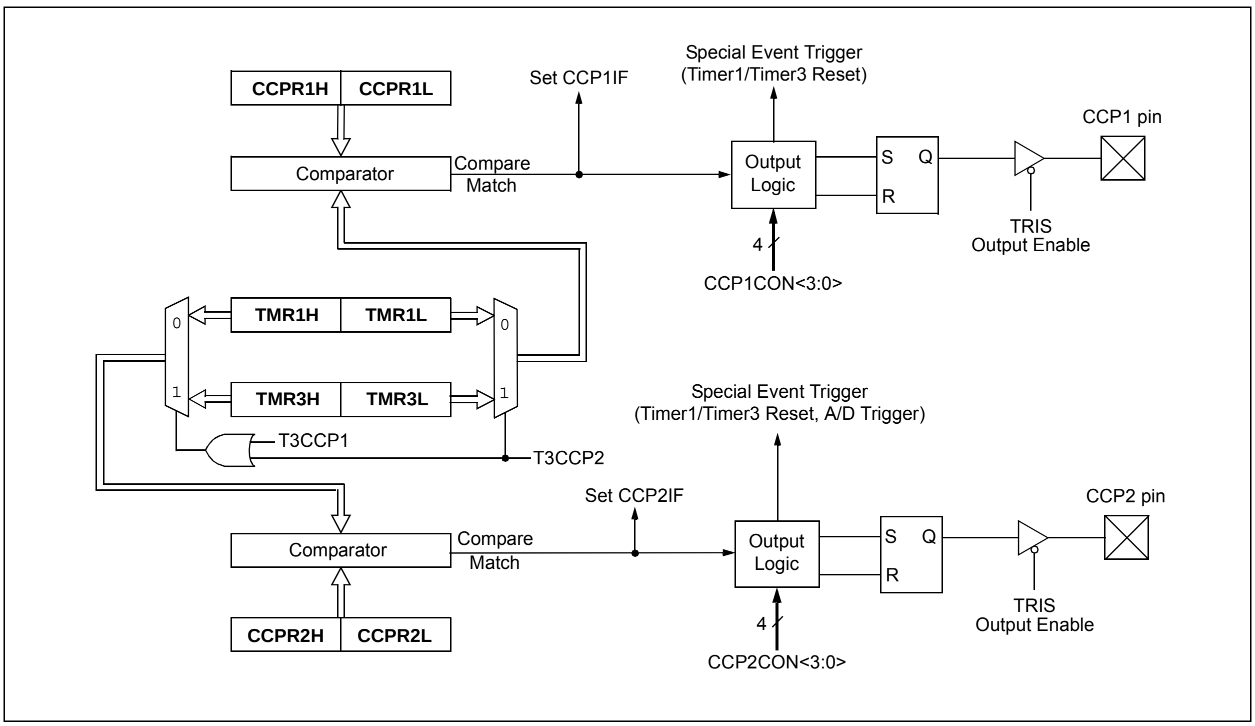
### 15.3.4 SPECIAL EVENT TRIGGER

Both CCP modules are equipped with a Special Event Trigger. This is an internal hardware signal generated in Compare mode to trigger actions by other modules. The Special Event Trigger is enabled by selecting the Compare Special Event Trigger mode (CCPxM<3:0> = `1011`).

For either CCP module, the Special Event Trigger resets the Timer register pair for whichever timer resource is currently assigned as the module's time base. This allows the CCPRx registers to serve as a programmable Period register for either timer.

The Special Event Trigger for CCP2 can also start an A/D conversion. In order to do this, the A/D Converter must already be enabled.

**FIGURE 15-2: COMPARE MODE OPERATION BLOCK DIAGRAM**



## 15.4 PWM Mode

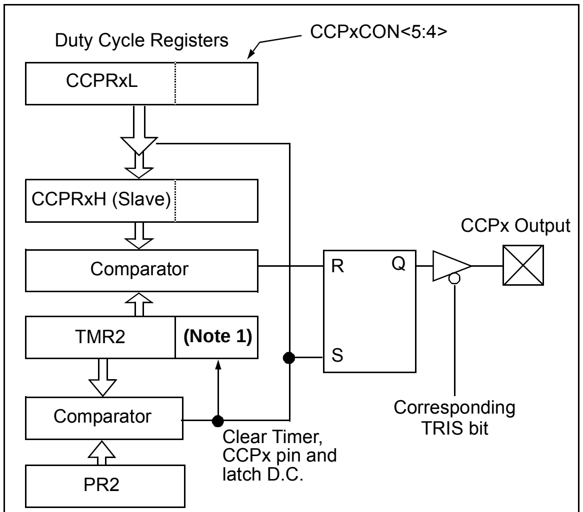
In Pulse-Width Modulation (PWM) mode, the CCPx pin produces up to a 10-bit resolution PWM output. Since the CCP2 pin is multiplexed with a PORTB or PORTC data latch, the appropriate TRIS bit must be cleared to make the CCP2 pin an output.

| **Note:** | Clearing the CCP2CON register will force the RB3 or RC1 output latch (depending on device configuration) to the default low level. This is not the PORTB or PORTC I/O data latch. |
|---|---|

Figure 15-3 shows a simplified block diagram of the CCP module in PWM mode.

For a step-by-step procedure on how to set up the CCP module for PWM operation, see **Section 15.4.4 "Setup for PWM Operation"**.

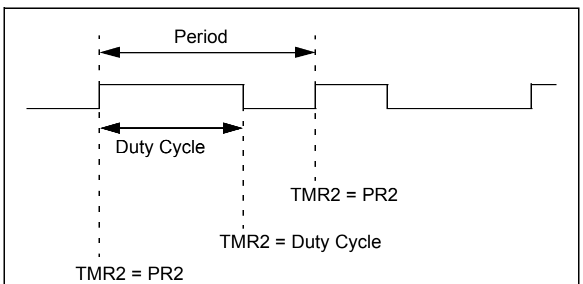
**FIGURE 15-3: SIMPLIFIED PWM BLOCK DIAGRAM**




**Note 1:** The 8-bit TMR2 value is concatenated with the 2-bit internal Q clock, or 2 bits of the prescaler, to create the 10-bit time base.

A PWM output (Figure 15-4) has a time base (period) and a time that the output stays high (duty cycle). The frequency of the PWM is the inverse of the period (1/period).

**FIGURE 15-4: PWM OUTPUT**




### 15.4.1 PWM PERIOD

The PWM period is specified by writing to the PR2 register. The PWM period can be calculated using the following formula:

**EQUATION 15-1:**

$$\text{PWM Period} = [(PR2) + 1] \cdot 4 \cdot T_{OSC} \cdot (\text{TMR2 Prescale Value})$$

PWM frequency is defined as 1/[PWM period].

When TMR2 is equal to PR2, the following three events occur on the next increment cycle:

- TMR2 is cleared
- The CCPx pin is set (exception: if PWM duty cycle = 0%, the CCPx pin will not be set)
- The PWM duty cycle is latched from CCPRxL into CCPRxH

| **Note:** | The Timer2 postscalers (see **Section 13.0 "Timer2 Module"**) are not used in the determination of the PWM frequency. The postscaler could be used to have a servo update rate at a different frequency than the PWM output. |
|---|---|

### 15.4.2 PWM DUTY CYCLE

The PWM duty cycle is specified by writing to the CCPRxL register and to the CCPxCON<5:4> bits. Up to 10-bit resolution is available. The CCPRxL contains the eight MSbs and the CCPxCON<5:4> bits contain the two LSbs. This 10-bit value is represented by CCPRxL:CCPxCON<5:4>. The following equation is used to calculate the PWM duty cycle in time:

**EQUATION 15-2:**

$$\text{PWM Duty Cycle} = (CCPRxL:CCPxCON<5:4>) \cdot T_{OSC} \cdot (\text{TMR2 Prescale Value})$$

CCPRxL and CCPxCON<5:4> can be written to at any time, but the duty cycle value is not latched into CCPRxH until after a match between PR2 and TMR2 occurs (i.e., the period is complete). In PWM mode, CCPRxH is a read-only register.

# 17.0 MASTER SYNCHRONOUS SERIAL PORT (MSSP) MODULE

## 17.1 Master SSP (MSSP) Module Overview

The Master Synchronous Serial Port (MSSP) module is a serial interface, useful for communicating with other peripheral or microcontroller devices. These peripheral devices may be serial EEPROMs, shift registers, display drivers, A/D Converters, etc. The MSSP module can operate in one of two modes:

- Serial Peripheral Interface (SPI)
- Inter-Integrated Circuit ($I^2C$)
  - Full Master mode
  - Slave mode (with general address call)

The $I^2C$ interface supports the following modes in hardware:

- Master mode
- Multi-Master mode
- Slave mode

## 17.2 Control Registers

The MSSP module has three associated registers. These include a status register (SSPSTAT) and two control registers (SSPCON1 and SSPCON2). The use of these registers and their individual configuration bits differ significantly depending on whether the MSSP module is operated in SPI or $I^2C$ mode.

Additional details are provided under the individual sections.

## 17.3 SPI Mode

The SPI mode allows 8 bits of data to be synchronously transmitted and received simultaneously. All four modes of SPI are supported. To accomplish communication, typically three pins are used:

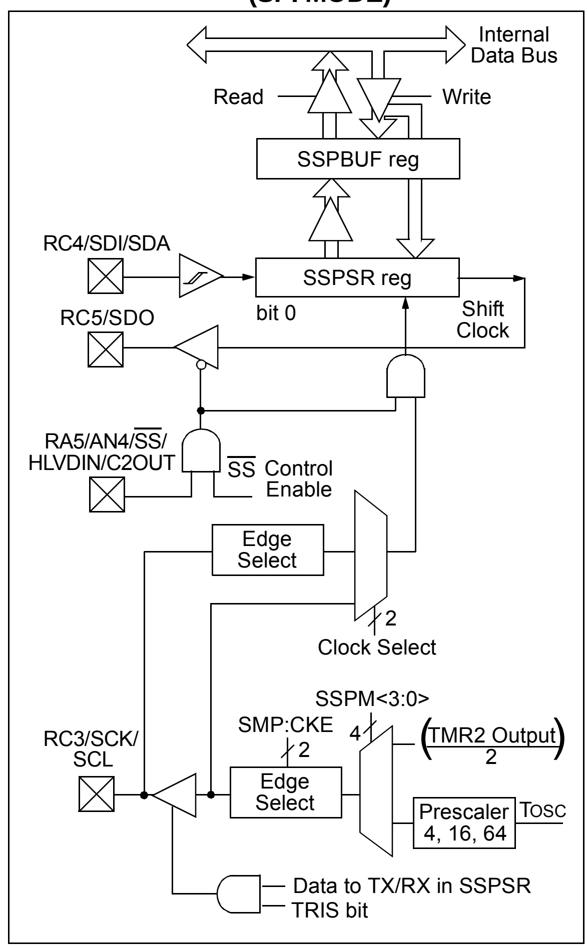
- Serial Data Out (SDO) – RC5/SDO
- Serial Data In (SDI) – RC4/SDI/SDA
- Serial Clock (SCK) – RC3/SCK/SCL

Additionally, a fourth pin may be used when in a Slave mode of operation:

- Slave Select ($\overline{SS}$) – RA5/$\overline{SS}$

Figure 17-1 shows the block diagram of the MSSP module when operating in SPI mode.

**FIGURE 17-1: MSSP BLOCK DIAGRAM (SPI MODE)**



### 17.3.2 OPERATION

When initializing the SPI, several options need to be specified. This is done by programming the appropriate control bits (SSPCON1<5:0> and SSPSTAT<7:6>). These control bits allow the following to be specified:

- Master mode (SCK is the clock output)
- Slave mode (SCK is the clock input)
- Clock Polarity (Idle state of SCK)
- Data Input Sample Phase (middle or end of data output time)
- Clock Edge (output data on rising/falling edge of SCK)
- Clock Rate (Master mode only)
- Slave Select mode (Slave mode only)

The MSSP consists of a transmit/receive shift register (SSPSR) and a buffer register (SSPBUF). The SSPSR shifts the data in and out of the device, MSb first. The SSPBUF holds the data that was written to the SSPSR until the received data is ready. Once the 8 bits of data have been received, that byte is moved to the SSPBUF register. Then, the Buffer Full detect bit, BF (SSPSTAT<0>) and the interrupt flag bit, SSPIF, are set. This double-buffering of the received data (SSPBUF) allows the next byte to start reception before reading the data that was just received. Any write to the SSPBUF register during transmission/reception of data will be ignored and the write collision detect bit, WCOL (SSPCON1<7>), will be set. User software must clear the WCOL bit so that it can be determined if the following write(s) to the SSPBUF register completed successfully.

When the application software is expecting to receive valid data, the SSPBUF should be read before the next byte of data to transfer is written to the SSPBUF. The Buffer Full bit, BF (SSPSTAT<0>), indicates when SSPBUF has been loaded with the received data (transmission is complete). When the SSPBUF is read, the BF bit is cleared. This data may be irrelevant if the SPI is only a transmitter. Generally, the MSSP interrupt is used to determine when the transmission/reception has completed. The SSPBUF must be read and/or written. If the interrupt method is not going to be used, then software polling can be done to ensure that a write collision does not occur. Example 17-1 shows the loading of the SSPBUF (SSPSR) for data transmission.

The SSPSR is not directly readable or writable and can only be accessed by addressing the SSPBUF register. Additionally, the MSSP Status register (SSPSTAT) indicates the various status conditions.

| **Note:** | The SSPBUF register cannot be used with read-modify-write instructions such as BCF, BTFSC and COMF, etc. |
|---|---|

**EXAMPLE 17-1: LOADING THE SSPBUF (SSPSR) REGISTER**

```
LOOP    BTFSS   SSPSTAT, BF     ;Has data been received (transmit complete)?
        BRA     LOOP            ;No
        MOVF    SSPBUF, W       ;WREG reg = contents of SSPBUF

        MOVWF   RXDATA          ;Save in user RAM, if data is meaningful

        MOVF    TXDATA, W       ;W reg = contents of TXDATA
        MOVWF   SSPBUF          ;New data to xmit
```

| **Note:** | To avoid lost data in Master mode, a read of the SSPBUF must be performed to clear the Buffer Full (BF) detect bit (SSPSTAT<0>) between each transmission. |
|---|---|

### 17.3.6 SLAVE MODE

In Slave mode, the data is transmitted and received as the external clock pulses appear on SCK. When the last bit is latched, the SSPIF interrupt flag bit is set.

Before enabling the module in SPI Slave mode, the clock line must match the proper Idle state. The clock line can be observed by reading the SCK pin. The Idle state is determined by the CKP bit (SSPCON1<4>).

While in Slave mode, the external clock is supplied by the external clock source on the SCK pin. This external clock must meet the minimum high and low times as specified in the electrical specifications.

While in Sleep mode, the slave can transmit/receive data. When a byte is received, the device will wake-up from Sleep.

### 17.3.7 SLAVE SELECT SYNCHRONIZATION

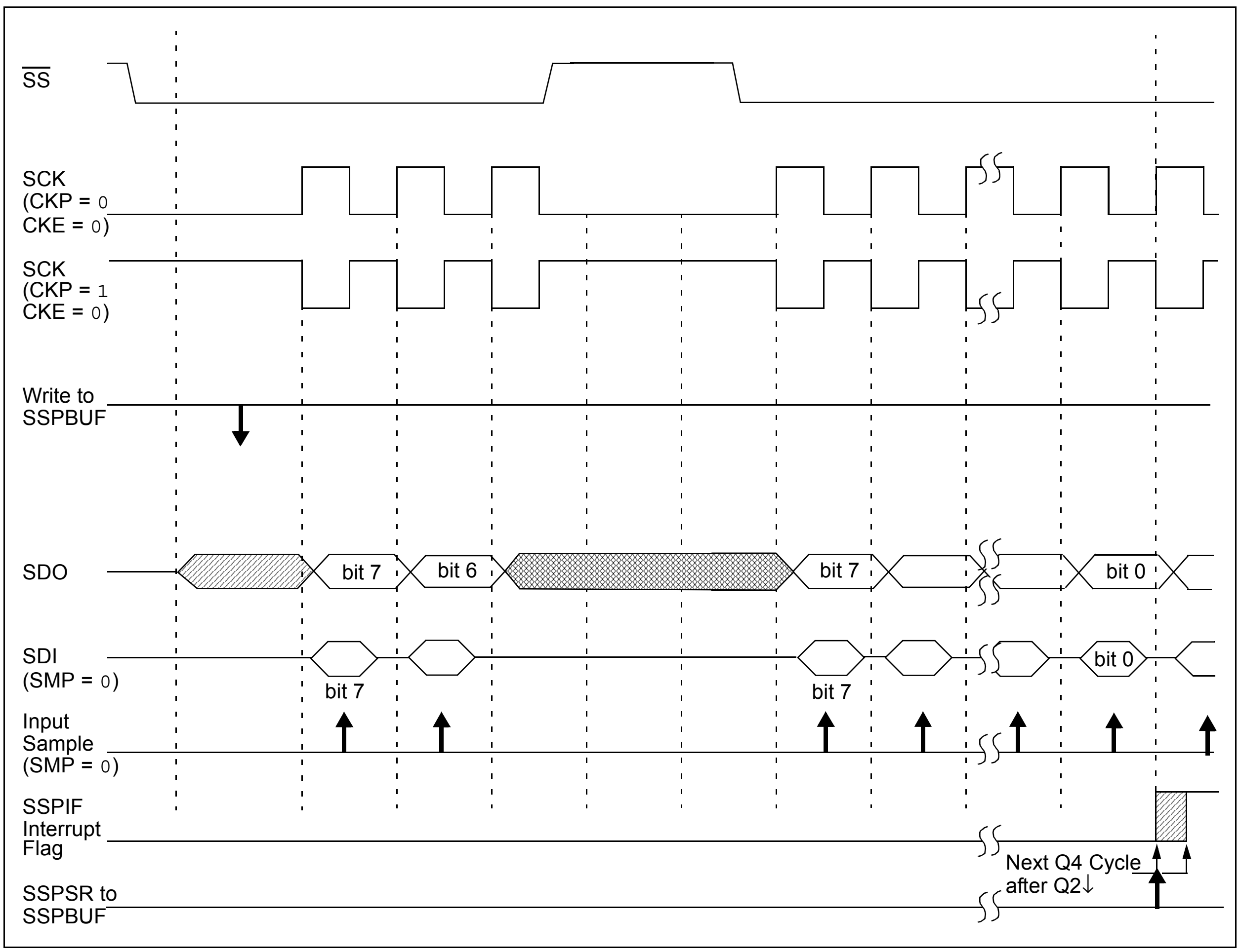
The $\overline{SS}$ pin allows a Synchronous Slave mode. The SPI must be in Slave mode with $\overline{SS}$ pin control enabled (SSPCON1<3:0> = 04h). The pin must not be driven low for the $\overline{SS}$ pin to function as an input. The data latch must be high. When the $\overline{SS}$ pin is low, transmission and reception are enabled and the SDO pin is driven. When the $\overline{SS}$ pin goes high, the SDO pin is no longer driven, even if in the middle of a transmitted byte and becomes a floating output. External pull-up/pull-down resistors may be desirable depending on the application.

> **Note 1:** When the SPI is in Slave mode with $\overline{SS}$ pin control enabled (SSPCON<3:0> = `0100`), the SPI module will reset if the $\overline{SS}$ pin is set to VDD.
>
> **2:** If the SPI is used in Slave mode with CKE set, then the $\overline{SS}$ pin control must be enabled.

When the SPI module resets, the bit counter is forced to ‘`0`’. This can be done by either forcing the $\overline{SS}$ pin to a high level or clearing the SSPEN bit.

To emulate two-wire communication, the SDO pin can be connected to the SDI pin. When the SPI needs to operate as a receiver, the SDO pin can be configured as an input. This disables transmissions from the SDO. The SDI can always be left as an input (SDI function) since it cannot create a bus conflict.

**FIGURE 17-4: SLAVE SYNCHRONIZATION WAVEFORM**



### 17.4.9 I$^2$C MASTER MODE REPEATED START CONDITION TIMING

A Repeated Start condition occurs when the RSEN bit (SSPCON2<1>) is programmed high and the I$^2$C logic module is in the Idle state. When the RSEN bit is set, the SCL pin is asserted low. When the SCL pin is sampled low, the Baud Rate Generator is loaded with the contents of SSPADD<5:0> and begins counting. The SDA pin is released (brought high) for one Baud Rate Generator count (TBRG). When the Baud Rate Generator times out, if SDA is sampled high, the SCL pin will be deasserted (brought high). When SCL is sampled high, the Baud Rate Generator is reloaded with the contents of SSPADD<6:0> and begins counting. SDA and SCL must be sampled high for one TBRG. This action is then followed by assertion of the SDA pin (SDA = `0`) for one TBRG while SCL is high. Following this, the RSEN bit (SSPCON2<1>) will be automatically cleared and the Baud Rate Generator will not be reloaded, leaving the SDA pin held low. As soon as a Start condition is detected on the SDA and SCL pins, the S bit (SSPSTAT<3>) will be set. The SSPIF bit will not be set until the Baud Rate Generator has timed out.

> **Note 1:** If RSEN is programmed while any other event is in progress, it will not take effect.
>
> **2:** A bus collision during the Repeated Start condition occurs if:
> - SDA is sampled low when SCL goes from low-to-high.
> - SCL goes low before SDA is asserted low. This may indicate that another master is attempting to transmit a data ‘`1`’.

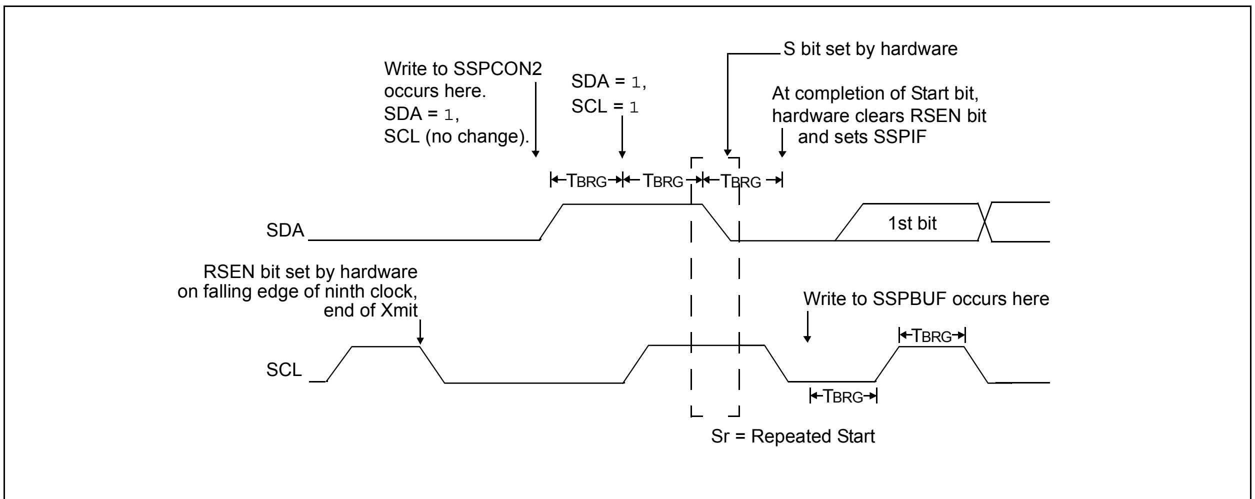
Immediately following the SSPIF bit getting set, the user may write the SSPBUF with the 7-bit address in 7-bit mode or the default first address in 10-bit mode. After the first eight bits are transmitted and an ACK is received, the user may then transmit an additional eight bits of address (10-bit mode) or eight bits of data (7-bit mode).

#### 17.4.9.1 WCOL Status Flag

If the user writes the SSPBUF when a Repeated Start sequence is in progress, the WCOL is set and the contents of the buffer are unchanged (the write doesn’t occur).

> **Note:** Because queueing of events is not allowed, writing of the lower 5 bits of SSPCON2 is disabled until the Repeated Start condition is complete.

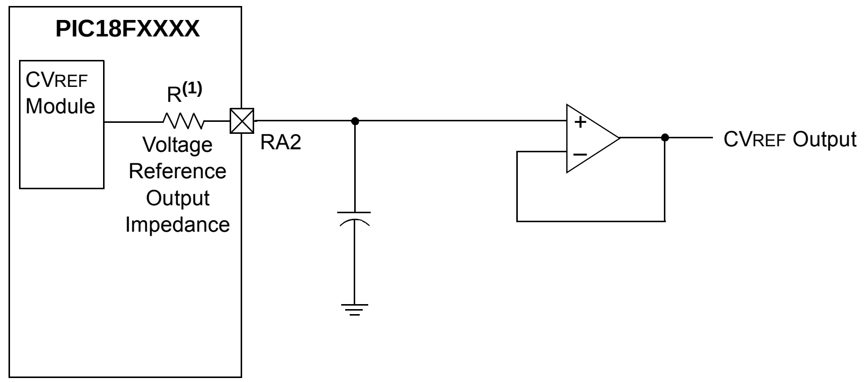
**FIGURE 17-20: REPEATED START CONDITION WAVEFORM**



**NOTES:**

**FIGURE 21-2: COMPARATOR VOLTAGE REFERENCE OUTPUT BUFFER EXAMPLE**




**Note 1:** R is dependent upon the comparator voltage reference configuration bits, CVRCON<3:0> and CVRCON<5>.

**TABLE 21-1: REGISTERS ASSOCIATED WITH COMPARATOR VOLTAGE REFERENCE**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|---|---|---|---|---|---|---|---|---|---|
| CVRCON | CVREN | CVROE | CVRR | CVRSS | CVR3 | CVR2 | CVR1 | CVR0 | 51 |
| CMCON | C2OUT | C1OUT | C2INV | C1INV | CIS | CM2 | CM1 | CM0 | 51 |
| TRISA | TRISA7[(1)] | TRISA6[(1)] | PORTA Data Direction Register | | | | | | 52 |

**Legend:** Shaded cells are not used with the comparator voltage reference.
**Note 1:** PORTA pins are enabled based on oscillator configuration.

| BCF | Bit Clear f |
|---|---|
| Syntax: | BCF f, b {,a} |
| Operands: | $0 \le f \le 255$<br>$0 \le b \le 7$<br>$a \in [0,1]$ |
| Operation: | 0 → f<b> |
| Status Affected: | None |
| Encoding: | 1001 \| bbba \| ffff \| ffff |
| Description: | Bit 'b' in register 'f' is cleared.<br>If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).<br>If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 24.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details. |
| Words: | 1 |
| Cycles: | 1 |

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register 'f' | Process Data | Write register 'f' |

Example: `BCF FLAG_REG, 7, 0`

Before Instruction
FLAG_REG = C7h
After Instruction
FLAG_REG = 47h

| BN | Branch if Negative |
|---|---|
| Syntax: | BN n |
| Operands: | $-128 \le n \le 127$ |
| Operation: | if Negative bit is '1',<br>(PC) + 2 + 2n → PC |
| Status Affected: | None |
| Encoding: | 1110 \| 0110 \| nnnn \| nnnn |
| Description: | If the Negative bit is '1', then the program will branch.<br>The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction. |
| Words: | 1 |
| Cycles: | 1(2) |

Q Cycle Activity:
If Jump:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read literal 'n' | Process Data | Write to PC |
| No operation | No operation | No operation | No operation |

If No Jump:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read literal 'n' | Process Data | No operation |

Example: `HERE BN Jump`

Before Instruction
PC = address (HERE)
After Instruction
If Negative = 1;
PC = address (Jump)
If Negative = 0;
PC = address (HERE + 2)

## TSTFSZ — Test f, Skip if 0

| | |
|---|---|
| Syntax: | TSTFSZ f {,a} |
| Operands: | $0 \leq f \leq 255$<br>$a \in [0,1]$ |
| Operation: | skip if f = 0 |
| Status Affected: | None |
| Encoding: | 0110 \| 011a \| ffff \| ffff |
| Description: | If ‘f’ = 0, the next instruction fetched during the current instruction execution is discarded and a NOP is executed, making this a two-cycle instruction.<br>If ‘a’ is ‘0’, the Access Bank is selected. If ‘a’ is ‘1’, the BSR is used to select the GPR bank (default).<br>If ‘a’ is ‘0’ and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 24.2.3 “Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode”** for details. |
| Words: | 1 |
| Cycles: | 1(2)<br>**Note:** 3 cycles if skip and followed by a 2-word instruction. |

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register ‘f’ | Process Data | No operation |

If skip:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| No operation | No operation | No operation | No operation |

If skip and followed by 2-word instruction:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| No operation | No operation | No operation | No operation |
| No operation | No operation | No operation | No operation |

Example:

```
HERE     TSTFSZ  CNT, 1
NZERO    :
ZERO     :
```

Before Instruction
- PC = Address (HERE)

After Instruction
- If CNT = 00h,
- PC = Address (ZERO)
- If CNT ≠ 00h,
- PC = Address (NZERO)

## XORLW — Exclusive OR Literal with W

| | |
|---|---|
| Syntax: | XORLW k |
| Operands: | $0 \leq k \leq 255$ |
| Operation: | (W) .XOR. k → W |
| Status Affected: | N, Z |
| Encoding: | 0000 \| 1010 \| kkkk \| kkkk |
| Description: | The contents of W are XORed with the 8-bit literal ‘k’. The result is placed in W. |
| Words: | 1 |
| Cycles: | 1 |

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read literal ‘k’ | Process Data | Write to W |

Example: `XORLW 0AFh`

Before Instruction
- W = B5h

After Instruction
- W = 1Ah

## 26.2 DC Characteristics: Power-Down and Supply Current PIC18F2420/2520/4420/4520 (Industrial) PIC18LF2420/2520/4420/4520 (Industrial) (Continued)

| PIC18LF2420/2520/4420/4520 (Industrial) | Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial |
|---|---|
| **PIC18F2420/2520/4420/4520** (Industrial, Extended) | **Standard Operating Conditions (unless otherwise stated)**<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended |

| Param No. | Device | Typ | Max | Units | Conditions | | |
|---|---|---|---|---|---|---|---|
| | **Supply Current (IDD)(2)** | | | | | | |
| | PIC18LF2X2X/4X20 | 65 | 100 | μA | -40°C | VDD = 2.0V | FOSC = 1 MHz (**PRI_IDLE** mode, EC oscillator) |
| | | 65 | 100 | μA | +25°C | | |
| | | 70 | 110 | μA | +85°C | | |
| | PIC18LF2X2X/4X20 | 120 | 140 | μA | -40°C | VDD = 3.0V | |
| | | 120 | 140 | μA | +25°C | | |
| | | 130 | 160 | μA | +85°C | | |
| | All devices | 230 | 300 | μA | -40°C | VDD = 5.0V | |
| | | 235 | 300 | μA | +25°C | | |
| | | 240 | 300 | μA | +85°C | | |
| | Extended devices only | 260 | 500 | μA | +125°C | | |
| | PIC18LF2X2X/4X20 | 260 | 360 | μA | -40°C | VDD = 2.0V | FOSC = 4 MHz (**PRI_IDLE** mode, EC oscillator) |
| | | 255 | 360 | μA | +25°C | | |
| | | 270 | 360 | μA | +85°C | | |
| | PIC18LF2X2X/4X20 | 420 | 620 | μA | -40°C | VDD = 3.0V | |
| | | 430 | 620 | μA | +25°C | | |
| | | 450 | 650 | μA | +85°C | | |
| | All devices | 0.9 | 1.2 | mA | -40°C | VDD = 5.0V | |
| | | 0.9 | 1.2 | mA | +25°C | | |
| | | 0.9 | 1.2 | mA | +85°C | | |
| | Extended devices only | 1 | 1.3 | mA | +125°C | | |
| | Extended devices only | 2.8 | 6.0 | mA | +125°C | VDD = 4.2V | FOSC = 25 MHz (**PRI_IDLE** mode, EC oscillator) |
| | | 4.3 | 8.0 | mA | +125°C | VDD = 5.0V | |
| | All devices | 6.0 | 10 | mA | -40°C | VDD = 4.2V | FOSC = 40 MHz (**PRI_IDLE** mode, EC oscillator) |
| | | 6.2 | 10 | mA | +25°C | | |
| | | 6.6 | 10 | mA | +85°C | | |
| | All devices | 8.1 | 13 | mA | -40°C | VDD = 5.0V | |
| | | 9.1 | 12 | mA | +25°C | | |
| | | 8.3 | 12 | mA | +85°C | | |

**Legend:** Shading of rows is to assist in readability of the table.

**Note 1:** The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

**2:** The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD or VSS;
MCLR = VDD; WDT enabled/disabled as specified.

**3:** When operation below -10°C is expected, use T1OSC High-Power mode, where LPT1OSC (CONFIG3H<2>) = 0. When operation will always be above -10°C, then the low-power Timer1 oscillator may be selected.

**4:** BOR and HLVD enable internal band gap reference. With both modules enabled, current consumption will be less than the sum of both specifications.

## 26.4 AC (Timing) Characteristics

### 26.4.1 TIMING PARAMETER SYMBOLOGY

The timing parameter symbols have been created using one of the following formats:

| | | | |
|---|---|---|---|
| 1. TppS2ppS | | 3. TCC:ST | ($I^2C$ specifications only) |
| 2. TppS | | 4. Ts | ($I^2C$ specifications only) |
| **T** | | | |
| F | Frequency | T | Time |
| **Lowercase letters (pp) and their meanings:** | | | |
| **pp** | | | |
| cc | CCP1 | osc | OSC1 |
| ck | CLKO | rd | $\overline{RD}$ |
| cs | $\overline{CS}$ | rw | $\overline{RD}$ or $\overline{WR}$ |
| di | SDI | sc | SCK |
| do | SDO | ss | $\overline{SS}$ |
| dt | Data in | t0 | T0CKI |
| io | I/O port | t1 | T13CKI |
| mc | $\overline{MCLR}$ | wr | $\overline{WR}$ |
| **Uppercase letters and their meanings:** | | | |
| **S** | | | |
| F | Fall | P | Period |
| H | High | R | Rise |
| I | Invalid (High-impedance) | V | Valid |
| L | Low | Z | High-impedance |
| **$I^2C$ only** | | | |
| AA | output access | High | High |
| BUF | Bus free | Low | Low |
| **TCC:ST ($I^2C$ specifications only)** | | | |
| **CC** | | | |
| HD | Hold | SU | Setup |
| **ST** | | | |
| DAT | DATA input hold | STO | Stop condition |
| STA | Start condition | | |

**TABLE 26-7: PLL CLOCK TIMING SPECIFICATIONS (VDD = 4.2V TO 5.5V)**

| Param No. | Sym | Characteristic | Min | Typ† | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| F10 | FOSC | Oscillator Frequency Range | 4 | — | 10 | MHz | HS mode only |
| F11 | FSYS | On-Chip VCO System Frequency | 16 | — | 40 | MHz | HS mode only |
| F12 | $t_{rc}$ | PLL Start-up Time (Lock Time) | — | — | 2 | ms | |
| F13 | ΔCLK | CLKO Stability (Jitter) | -2 | — | +2 | % | |

† Data in “Typ” column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**TABLE 26-8: AC CHARACTERISTICS: INTERNAL RC ACCURACY PIC18F2420/2520/4420/4520 (INDUSTRIAL) PIC18LF2420/2520/4420/4520 (INDUSTRIAL)**

| **PIC18LF2420/2520/4420/4520** (Industrial) | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial | | | | | |
|---|---|---|---|---|---|---|---|
| **PIC18F2420/2520/4420/4520** (Industrial) | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial | | | | | |
| **Param No.** | **Device** | **Min** | **Typ** | **Max** | **Units** | **Conditions** | |
| | **INTOSC Accuracy @ Freq = 8 MHz, 4 MHz, 2 MHz, 1 MHz, 500 kHz, 250 kHz, 125 kHz, 31 kHz(1)** | | | | | | |
| | PIC18LF2420/2520/4420/4520 | -2 | +/-1 | 2 | % | +25°C | VDD = 2.7-3.3V |
| | | -5 | +/-1 | 5 | % | -40°C to +85°C | VDD = 2.7-3.3V |
| | PIC18F2420/2520/4420/4520 | -2 | +/-1 | 2 | % | +25°C | VDD = 4.5-5.5V |
| | | -5 | +/-1 | 5 | % | -40°C to +85°C | VDD = 4.5-5.5V |
| | **INTRC Accuracy @ Freq = 31 kHz** | | | | | | |
| | PIC18LF2420/2520/4420/4520 | 26.562 | — | 35.938 | kHz | -40°C to +85°C | VDD = 2.7-3.3V |
| | PIC18F2420/2520/4420/4520 | 26.562 | — | 35.938 | kHz | -40°C to +85°C | VDD = 4.5-5.5V |

**Legend:** Shading of rows is to assist in readability of the table.

**Note 1:** Frequency calibrated at 25°C. OSCTUNE register can be used to compensate for temperature drift.