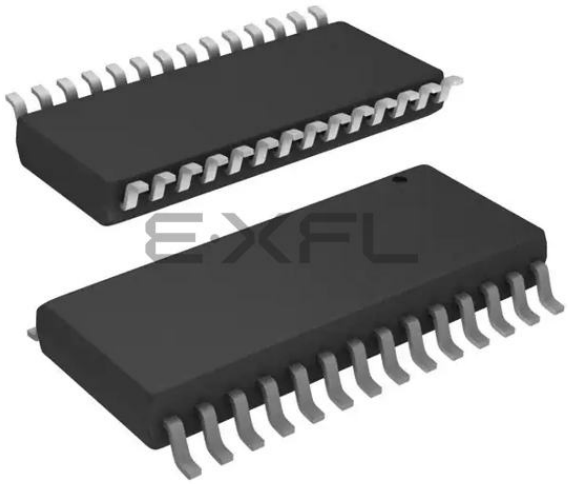


**Welcome to E-XFL.COM**

## 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 | CANbus, I²C, SPI, UART/USART |
| Peripherals | Brown-out Detect/Reset, HLVD, POR, PWM, WDT |
| Number of I/O | 25 |
| Program Memory Size | 48KB (24K x 16) |
| Program Memory Type | FLASH |
| EEPROM Size | 1K x 8 |
| RAM Size | 3.25K x 8 |
| Voltage - Supply (Vcc/Vdd) | 2V ~ 5.5V |
| Data Converters | A/D 8x10b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 85°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 28-SOIC (0.295", 7.50mm Width) |
| Supplier Device Package | 28-SOIC |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/pic18lf2585-i-so |

**TABLE 1-2: PIC18F2585/2680 PINOUT I/O DESCRIPTIONS (CONTINUED)**

| Pin Name | Pin Number | Pin Type | Buffer Type | Description |
|---|---|---|---|---|
| | PDIP, SOIC | | | |
| | | | | PORTC is a bidirectional I/O port. |
| RC0/T1OSO/T13CKI | 11 | | | |
| RC0 | | I/O | ST | Digital I/O. |
| T1OSO | | O | — | Timer1 oscillator output. |
| T13CKI | | I | ST | Timer1/Timer3 external clock input. |
| RC1/T1OSI | 12 | | | |
| RC1 | | I/O | ST | Digital I/O. |
| T1OSI | | I | CMOS | Timer1 oscillator input. |
| RC2/CCP1 | 13 | | | |
| RC2 | | I/O | ST | Digital I/O. |
| CCP1 | | I/O | ST | Capture1 input/Compare1 output/PWM1 output. |
| RC3/SCK/SCL | 14 | | | |
| 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 | | | |
| 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 | | | |
| RC5 | | I/O | ST | Digital I/O. |
| SDO | | O | — | SPI data out. |
| RC6/TX/CK | 17 | | | |
| 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 | | | |
| 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 $\overline{MCLR}$/$V_{PP}$/RE3 pin. |
| $V_{SS}$ | 8, 19 | P | — | Ground reference for logic and I/O pins. |
| $V_{DD}$ | 20 | 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

If the IRCF bits and the INTSRC bit are all clear, the INTOSC output is not enabled and the IOFS bit will remain clear; there will be no indication of the current clock source. The INTRC source is providing the device clocks.

If the IRCF bits are changed from all clear (thus, enabling the INTOSC output) or if INTSRC is set, the IOFS bit becomes set after the INTOSC output becomes stable. Clocks to the device continue while the INTOSC source stabilizes after an interval of TIOBST.

If the IRCF bits were previously at a non-zero value or if INTSRC was set before setting SCS1 and the INTOSC source was already stable, the IOFS bit will remain set.

On transitions from RC_RUN mode to PRI_RUN mode, the device continues to be clocked from the INTOSC multiplexer while the primary clock is started. When the primary clock becomes ready, a clock switch to the primary clock occurs (see Figure 3-4). When the clock switch is complete, the IOFS bit is cleared, the OSTS bit is set and the primary clock is providing the device clock. The IDLEN and SCS bits are not affected by the switch. The INTRC source will continue to run if either the WDT or the Fail-Safe Clock Monitor is enabled.

**FIGURE 3-3: TRANSITION TIMING TO RC_RUN MODE**

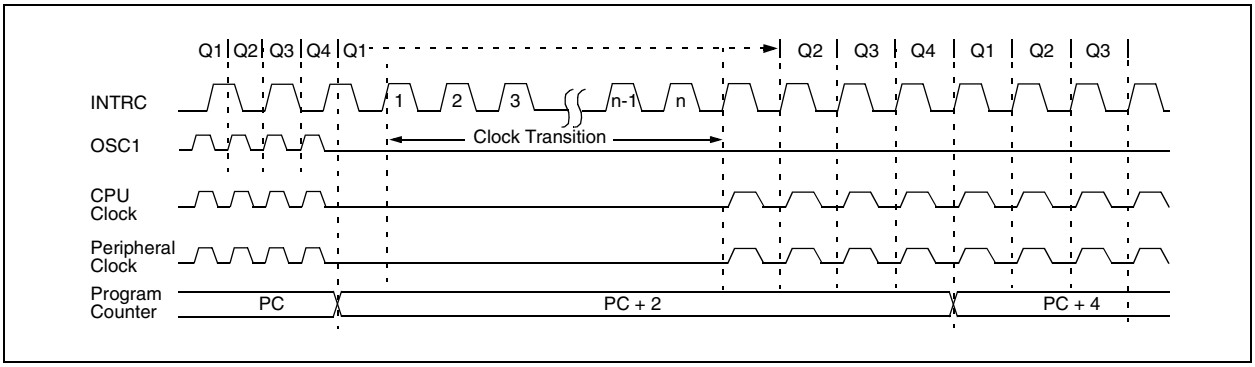
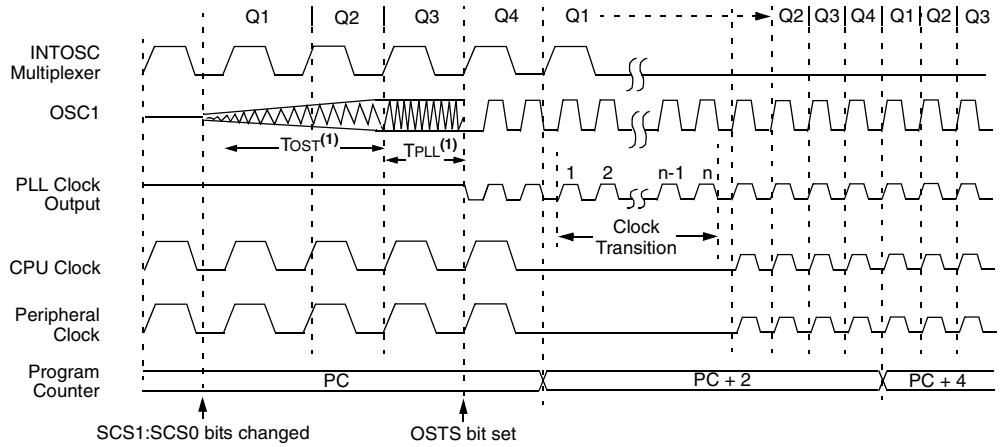



**FIGURE 3-4: TRANSITION TIMING FROM RC_RUN MODE TO PRI_RUN MODE**




**Note 1:** TOST = 1024 TOSC; TPLL = 2 ms (approx). These intervals are not shown to scale.

**TABLE 4-4: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)**

| Register | Applicable Devices | | | | Power-on Reset, Brown-out Reset | $\overline{\text{MCLR}}$ Resets, WDT Reset, RESET Instruction, Stack Resets | Wake-up via WDT or Interrupt |
|---|---|---|---|---|---|---|---|
| TXB2D1 | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | 0uuu uuuu |
| TXB2D0 | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | 0uuu uuuu |
| TXB2DLC | 2585 | 2680 | 4585 | 4680 | -x-- xxxx | -u-- uuuu | -u-- uuuu |
| TXB2EIDL | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| TXB2EIDH | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| TXB2SIDL | 2585 | 2680 | 4585 | 4680 | xxxx x-xx | uuuu u-uu | -uuu uuuu |
| TXB2SIDH | 2585 | 2680 | 4585 | 4680 | xxx- x-xx | uuu- u-uu | uuu- u-uu |
| TXB2CON | 2585 | 2680 | 4585 | 4680 | 0000 0-00 | 0000 0-00 | uuuu u-uu |
| RXM1EIDL | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| RXM1EIDH | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| RXM1SIDL | 2585 | 2680 | 4585 | 4680 | xxx- x-xx | uuu- u-uu | uuu- u-uu |
| RXM1SIDH | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| RXM0EIDL | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| RXM0EIDH | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| RXM0SIDL | 2585 | 2680 | 4585 | 4680 | xxx- x-xx | uuu- u-uu | uuu- u-uu |
| RXM0SIDH | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| RXF5EIDL | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| RXF5EIDH | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| RXF5SIDL | 2585 | 2680 | 4585 | 4680 | xxx- x-xx | uuu- u-uu | uuu- u-uu |
| RXF5SIDH | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| RXF4EIDL | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| RXF4EIDH | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| RXF4SIDL | 2585 | 2680 | 4585 | 4680 | xxx- x-xx | uuu- u-uu | uuu- u-uu |
| RXF4SIDH | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| RXF3EIDL | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| RXF3EIDH | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| RXF3SIDL | 2585 | 2680 | 4585 | 4680 | xxx- x-xx | uuu- u-uu | uuu- u-uu |
| RXF3SIDH | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| RXF2EIDL | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| RXF2EIDH | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| RXF2SIDL | 2585 | 2680 | 4585 | 4680 | xxx- x-xx | uuu- u-uu | uuu- u-uu |
| RXF2SIDH | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |

**Legend:** u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition. Shaded cells indicate conditions do not apply for the designated device.

**Note 1:** One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

**2:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

**3:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

**4:** See Table 4-3 for Reset value for specific condition.

**5:** Bits 6 and 7 of PORTA, LATA and TRISA are enabled, depending on the oscillator mode selected. When not enabled as PORTA pins, they are disabled and read '0'.

**6:** This register reads all '0's until the ECAN™ technology is set up in Mode 1 or Mode 2.

**TABLE 4-4: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)**

| Register | Applicable Devices | | | | Power-on Reset, Brown-out Reset | MCLR Resets, WDT Reset, RESET Instruction, Stack Resets | Wake-up via WDT or Interrupt |
|---|---|---|---|---|---|---|---|
| B4EIDH(6) | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| B4SIDL(6) | 2585 | 2680 | 4585 | 4680 | xxxx x-xx | uuuu u-uu | uuuu u-uu |
| B4SIDH(6) | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| B4CON(6) | 2585 | 2680 | 4585 | 4680 | 0000 0000 | 0000 0000 | uuuu uuuu |
| B3D7(6) | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| B3D6(6) | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| B3D5(6) | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| B3D4(6) | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| B3D3(6) | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| B3D2(6) | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| B3D1(6) | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| B3D0(6) | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| B3DLC(6) | 2585 | 2680 | 4585 | 4680 | -xxx xxxx | -uuu uuuu | -uuu uuuu |
| B3EIDL(6) | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| B3EIDH(6) | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| B3SIDL(6) | 2585 | 2680 | 4585 | 4680 | xxxx x-xx | uuuu u-uu | uuuu u-uu |
| B3SIDH(6) | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| B3CON(6) | 2585 | 2680 | 4585 | 4680 | 0000 0000 | 0000 0000 | uuuu uuuu |
| B2D7(6) | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| B2D6(6) | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| B2D5(6) | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| B2D4(6) | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| B2D3(6) | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| B2D2(6) | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| B2D1(6) | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| B2D0(6) | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| B2DLC(6) | 2585 | 2680 | 4585 | 4680 | -xxx xxxx | -uuu uuuu | -uuu uuuu |
| B2EIDL(6) | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| B2EIDH(6) | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| B2SIDL(6) | 2585 | 2680 | 4585 | 4680 | xxxx x-xx | uuuu u-uu | uuuu u-uu |
| B2SIDH(6) | 2585 | 2680 | 4585 | 4680 | xxxx xxxx | uuuu uuuu | uuuu uuuu |

**Legend:** u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition. Shaded cells indicate conditions do not apply for the designated device.

**Note 1:** One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

**2:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

**3:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

**4:** See Table 4-3 for Reset value for specific condition.

**5:** Bits 6 and 7 of PORTA, LATA and TRISA are enabled, depending on the oscillator mode selected. When not enabled as PORTA pins, they are disabled and read '0'.

**6:** This register reads all '0's until the ECAN™ technology is set up in Mode 1 or Mode 2.

**TABLE 5-2: REGISTER FILE SUMMARY (PIC18F2585/2680/4585/4680) (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 High Byte | | | | | | | | 0000 0000 | 51, 231 |
| SPBRG | EUSART Baud Rate Generator | | | | | | | | 0000 0000 | 51, 231 |
| RCREG | EUSART Receive Register | | | | | | | | 0000 0000 | 51, 238 |
| TXREG | EUSART Transmit Register | | | | | | | | 0000 0000 | 51, 236 |
| TXSTA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 0000 0010 | 51, 237 |
| RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 0000 000x | 51, 237 |
| EEADRH | — | — | — | — | — | — | EEPROM Addr Register High | | ---- --00 | 51, 108 |
| EEADR | EEPROM Address Register | | | | | | | | 0000 0000 | 51, 105 |
| EEDATA | EEPROM Data Register | | | | | | | | 0000 0000 | 51, 105 |
| EECON2 | EEPROM Control Register 2 (not a physical register) | | | | | | | | 0000 0000 | 51, 105 |
| EECON1 | EEPGD | CFGS | — | FREE | WRERR | WREN | WR | RD | xx-0 x000 | 51, 105 |
| IPR3 Mode 0 | IRXIP | WAKIP | ERRIP | TXB2IP | TXB1IP | TXB0IP | RXB1IP | RXB0IP | 1111 1111 | 51, 126 |
| IPR3 Mode 1, 2 | IRXIP | WAKIP | ERRIP | TXBnIP | TXB1IP(8) | TXB0IP(8) | RXBnIP | FIFOWMIP | 1111 1111 | 51, 126 |
| PIR3 Mode 0 | IRXIF | WAKIF | ERRIF | TXB2IF | TXB1IF | TXB0IF | RXB1IF | RXB0IF | 0000 0000 | 51, 120 |
| PIR3 Mode 1, 2 | IRXIF | WAKIF | ERRIF | TXBnIF | TXB1IF(8) | TXB0IF(8) | RXBnIF | FIFOWMIF | 0000 0000 | 51, 120 |
| PIE3 Mode 0 | IRXIE | WAKIE | ERRIE | TXB2IE | TXB1IE | TXB0IE | RXB1IE | RXB0IE | 0000 0000 | 51, 123 |
| PIE3 Mode 1, 2 | IRXIE | WAKIE | ERRIE | TXBnIE | TXB1IE(8) | TXB0IE(8) | RXBnIE | FIFOMWIE | 0000 0000 | 51, 123 |
| IPR2 | OSCFIP | CMIP(9) | — | EEIP | BCLIP | HLVDIP | TMR3IP | ECCP1IP(9) | 11-1 1111 | 51, 125 |
| PIR2 | OSCFIF | CMIF(9) | — | EEIF | BCLIF | HLVDIF | TMR3IF | ECCP1IF(9) | 00-0 0000 | 51, 119 |
| PIE2 | OSCFIE | CMIE(9) | — | EEIE | BCLIE | HLVDIE | TMR3IE | ECCP1IE(9) | 00-0 0000 | 52, 122 |
| IPR1 | PSPIP(3) | ADIP | RCIP | TXIP | SSPIP | CCP1IP | TMR2IP | TMR1IP | 1111 1111 | 52, 124 |
| PIR1 | PSPIF(3) | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 0000 0000 | 52, 118 |
| PIE1 | PSPIE(3) | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 0000 0000 | 52, 121 |
| OSCTUNE | INTSRC | PLLEN(4) | — | TUN4 | TUN3 | TUN2 | TUN1 | TUN0 | 0q-0 0000 | 27, 52 |
| TRISE(3) | IBF | OBF | IBOV | PSPMODE | — | TRISE2 | TRISE1 | TRISE0 | 0000 -111 | 52, 141 |
| TRISD(3) | Data Direction Control Register for PORTD | | | | | | | | 1111 1111 | 52, 138 |
| TRISC | Data Direction Control Register for PORTC | | | | | | | | 1111 1111 | 52, 135 |
| TRISB | Data Direction Control Register for PORTB | | | | | | | | 1111 1111 | 52, 132 |
| TRISA | TRISA7(6) | TRISA6(6) | Data Direction Control Register for PORTA | | | | | | 1111 1111 | 52, 129 |
| LATE(3) | — | — | — | — | — | LATE2 | LATE1 | LATE0 | ---- -xxx | 52, 141 |
| LATD(3) | Read PORTD Data Latch, Write PORTD Data Latch | | | | | | | | xxxx xxxx | 52, 138 |
| LATC | Read PORTC Data Latch, Write PORTC Data Latch | | | | | | | | xxxx xxxx | 52, 135 |
| LATB | Read PORTB Data Latch, Write PORTB Data Latch | | | | | | | | xxxx xxxx | 52, 132 |
| LATA | LATA7(6) | LATA6(6) | Read PORTA Data Latch, Write PORTA Data Latch | | | | | | xxxx xxxx | 52, 129 |

**Legend:** x = unknown, u = unchanged, - = unimplemented, q = value depends on condition

**Note 1:** Bit 21 of the PC is only available in Test mode and Serial Programming modes.

**2:** The SBOREN bit is only available when CONFIG2L<1:0> = 01; otherwise, it is disabled and reads as '0'. See **Section 4.4 "Brown-out Reset (BOR)"**.

**3:** These registers and/or bits are not implemented on PIC18F2X8X devices and are read as '0'. Reset values are shown for PIC18F4X8X devices; individual unimplemented bits should be interpreted as '—'.

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

**5:** The RE3 bit is only available when Master Clear Reset is disabled (CONFIG3H<7> = 0); otherwise, RE3 reads as '0'. This bit is read-only.

**6:** 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'.

**7:** CAN bits have multiple functions depending on the selected mode of the CAN module.

**8:** This register reads all '0's until the ECAN™ technology is set up in Mode 1 or Mode 2.

**9:** These registers are available on PIC18F4X8X devices only.

## 5.4 Data Addressing Modes

**Note:** The execution of some instructions in the core PIC18 instruction set are changed when the PIC18 extended instruction set is enabled. See **Section 5.6 "Data Memory and the Extended Instruction Set"** for more information.

While the program memory can be addressed in only one way – through the program counter – information in the data memory space can be addressed in several ways. For most instructions, the addressing mode is fixed. Other instructions may use up to three modes, depending on which operands are used and whether or not the extended instruction set is enabled.

The addressing modes are:

- Inherent
- Literal
- Direct
- Indirect

An additional addressing mode, Indexed Literal Offset, is available when the extended instruction set is enabled (XINST Configuration bit = 1). Its operation is discussed in greater detail in **Section 5.6.1 "Indexed Addressing with Literal Offset"**.

### 5.4.1 INHERENT AND LITERAL ADDRESSING

Many PIC18 control instructions do not need any argument at all; they either perform an operation that globally affects the device or they operate implicitly on one register. This addressing mode is known as Inherent Addressing. Examples include SLEEP, RESET and DAW.

Other instructions work in a similar way but require an additional explicit argument in the opcode. This is known as Literal Addressing mode because they require some literal value as an argument. Examples include ADDLW and MOVLW which, respectively, add or move a literal value to the W register. Other examples include CALL and GOTO, which include a 20-bit program memory address.

### 5.4.2 DIRECT ADDRESSING

Direct addressing specifies all or part of the source and/or destination address of the operation within the opcode itself. The options are specified by the arguments accompanying the instruction.

In the core PIC18 instruction set, bit-oriented and byte-oriented instructions use some version of direct addressing by default. All of these instructions include some 8-bit literal address as their Least Significant Byte. This address specifies either a register address in one of the banks of data RAM (**Section 5.3.3 "General Purpose Register File"**) or a location in the Access Bank (**Section 5.3.2 "Access Bank"**) as the data source for the instruction.

The Access RAM bit 'a' determines how the address is interpreted. When 'a' is '1', the contents of the BSR (**Section 5.3.1 "Bank Select Register (BSR)"**) are used with the address to determine the complete 12-bit address of the register. When 'a' is '0', the address is interpreted as being a register in the Access Bank. Addressing that uses the Access RAM is sometimes also known as Direct Forced Addressing mode.

A few instructions, such as MOVFF, include the entire 12-bit address (either source or destination) in their opcodes. In those cases, the BSR is ignored entirely.

The destination of the operation's results is determined by the destination bit 'd'. When 'd' is '1', the results are stored back in the source register, overwriting its original contents. When 'd' is '0', the results are stored in the W register. Instructions without the 'd' argument have a destination that is implicit in the instruction; their destination is either the target register being operated on or the W register.

### 5.4.3 INDIRECT ADDRESSING

Indirect addressing allows the user to access a location in data memory without giving a fixed address in the instruction. This is done by using File Select Registers (FSRs) as pointers to the locations to be read or written to. Since the FSRs are themselves located in RAM as Special File Registers, they can also be directly manipulated under program control. This makes FSRs very useful in implementing data structures, such as tables and arrays in data memory.

The registers for indirect addressing are also implemented with Indirect File Operands (INDFs) that permit automatic manipulation of the pointer value with auto-incrementing, auto-decrementing or offsetting with another value. This allows for efficient code, using loops, such as the example of clearing an entire RAM bank in Example 5-5.

**EXAMPLE 5-5: HOW TO CLEAR RAM (BANK 1) USING INDIRECT ADDRESSING**

```
        LFSR   FSR0, 100h  ;
NEXT    CLRF   POSTINC0    ; Clear INDF
                           ; register then
                           ; inc pointer
        BTFSS  FSR0H, 1    ; All done with
                           ; Bank1?
        BRA    NEXT        ; NO, clear next
CONTINUE                   ; YES, continue
```

**TABLE 10-3: PORTB I/O SUMMARY**

| Pin Name | Function | I/O | TRIS | Buffer | Description |
|---|---|---|---|---|---|
| RB0/INT0/FLT0/AN10 | RB0 | OUT | 0 | DIG | LATB<0> data output. |
| | | IN | 1 | TTL | PORTB<0> data input. Weak pull-up available only in this mode. |
| | INT0 | IN | 1 | ST | External interrupt 0 input. |
| | FLT0 | IN | 1 | ST | Enhanced PWM Fault input. |
| | AN10 | IN | 1 | ANA | A/D input channel 10. Enabled on POR, this analog input overrides the digital input (read as clear – low level). |
| RB1/INT1/AN8 | RB1 | OUT | 0 | DIG | LATB<1> data output. |
| | | IN | 1 | TTL | PORTB<1> data input. Weak pull-up available only in this mode. |
| | INT1 | IN | 1 | ST | External interrupt 1 input. |
| | AN8 | IN | 1 | ANA | A/D input channel 8. Enabled on POR, this analog input overrides the digital input (read as clear – low level). |
| RB2/INT2/CANTX | RB2 | OUT | x | DIG | LATB<2> data output. |
| | | IN | 1 | TTL | PORTB<2> data input. Weak pull-up available only in this mode. |
| | INT2 | IN | 1 | ST | External interrupt 2 input. |
| | CANTX | OUT | 1 | DIG | CAN transmit signal output. The CAN interface overrides the TRIS<2> control when enabled. |
| RB3/CANRX | RB3 | OUT | 0 | DIG | LATB<3> data output. |
| | | IN | 1 | TTL | PORTB<3> data input. Weak pull-up available only in this mode. |
| | CANRX | IN | 1 | ST | CAN receive signal input. Pin must be configured as a digital input by setting TRISB<3>. |
| RB4/KBI0/AN9 | RB4 | OUT | 0 | DIG | LATB<4> data output. |
| | | IN | 1 | TTL | PORTB<4> data input. Weak pull-up available only in this mode. |
| | KBI0 | IN | 1 | TTL | Interrupt-on-pin change. |
| | AN9 | IN | 1 | ANA | A/D input channel 9. Enabled on POR, this analog input overrides the digital input (read as clear – low level). |
| RB5/KBI1/PGM | RB5 | OUT | 0 | DIG | LATB<5> data output. |
| | | IN | 1 | TTL | PORTB<5> data input. Weak pull-up available only in this mode. |
| | KBI1 | IN | 1 | TTL | Interrupt-on-pin change. |
| | PGM | IN | x | ST | Low-Voltage Programming mode entry (ICSP™). Enabling this function overrides digital output. |
| RB6/KBI2/PGC | RB6 | OUT | 0 | DIG | LATB<6> data output. |
| | | IN | 1 | TTL | PORTB<6> data input. Weak pull-up available only in this mode. |
| | KBI2 | IN | 1 | TTL | Interrupt-on-pin change. |
| | PGC | IN | x | ST | Low-Voltage Programming mode entry (ICSP) clock input. |
| RB7/KBI3/PGD | RB7 | OUT | 0 | DIG | LATB<7> data output. |
| | | IN | 1 | TTL | PORTB<7> data input. Weak pull-up available only in this mode. |
| | KBI3 | IN | 1 | TTL | Interrupt-on-pin change. |
| | PGD | OUT | x | DIG | Low-Voltage Programming mode entry (ICSP) clock output. |
| | | IN | x | ST | Low-Voltage Programming mode entry (ICSP) clock input. |

**Legend:** PWR = Power Supply; OUT = Output; IN = Input; ANA = Analog Signal; DIG = Digital Output; ST = Schmitt Buffer Input; TTL – TTL Buffer Input

## 14.0 TIMER3 MODULE

The Timer3 module timer/counter incorporates these features:

- Software selectable operation as a 16-bit timer or counter
- Readable and writable 8-bit registers (TMR3H and TMR3L)
- Selectable clock source (internal or external) with device clock or Timer1 oscillator internal options
- Interrupt-on-overflow
- Module Reset on CCP1 special event trigger

A simplified block diagram of the Timer3 module is shown in Figure 14-1. A block diagram of the module's operation in Read/Write mode is shown in Figure 14-2.

The Timer3 module is controlled through the T3CON register (Register 14-1). It also selects the clock source options for the CCP1 modules (see **Section 15.1.1 "CCP1 Modules and Timer Resources"** for more information).

**REGISTER 14-1: T3CON: TIMER3 CONTROL REGISTER**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| RD16 | T3ECCP1(1) | T3CKPS1 | T3CKPS0 | T3CCP1(1) | $\overline{\text{T3SYNC}}$ | TMR3CS | TMR3ON |
| bit 7 | | | | | | | bit 0 |

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

bit 6,3 **T3ECCP1:T3CCP1:** Timer3 and Timer1 to ECCP1/CCP1 Enable bits(1)
1x = Timer3 is the capture/compare clock source for both CCP1 and ECCP1 modules
01 = Timer3 is the capture/compare clock source for ECCP1;
Timer1 is the capture/compare clock source for CCP1
00 = Timer1 is the capture/compare clock source for both CCP1 and ECCP1 modules

**Note 1:** These bits are available on PIC18F4X8X devices only.

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

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

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

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

| **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 |

**TABLE 16-3: REGISTERS ASSOCIATED WITH ECCP1 MODULE AND TIMER1 TO TIMER3**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 49 |
| RCON | IPEN | SBOREN | — | $\overline{RI}$ | $\overline{TO}$ | $\overline{PD}$ | $\overline{POR}$ | $\overline{BOR}$ | 50 |
| IPR1 | PSPIP | ADIP | RCIP | TXIP | SSPIP | CCP1IP | TMR2IP | TMR1IP | 52 |
| PIR1 | PSPIF | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 52 |
| PIE1 | PSPIE | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 52 |
| IPR2 | OSCFIP | CMIP(3) | — | EEIP | BCLIP | HLVDIP | TMR3IP | ECCP1IP(3) | 51 |
| PIR2 | OSCFIF | CMIF(3) | — | EEIF | BCLIF | HLVDIF | TMR3IF | ECCP1IF(3) | 51 |
| PIE2 | OSCFIE | CMIE(3) | — | EEIE | BCLIE | HLVDIE | TMR3IE | ECCP1IE(3) | 52 |
| TRISB | PORTB Data Direction Register | | | | | | | | 52 |
| TRISC | PORTC Data Direction Register | | | | | | | | 52 |
| TRISD(1) | PORTD Data Direction Register | | | | | | | | 52 |
| TMR1L | Holding Register for the Least Significant Byte of the 16-bit TMR1 Register | | | | | | | | 50 |
| TMR1H | Holding Register for the Most Significant Byte of the 16-bit TMR1 Register | | | | | | | | 50 |
| T1CON | RD16 | T1RUN | T1CKPS1 | T1CKPS0 | T1OSCEN | $\overline{T1SYNC}$ | TMR1CS | TMR1ON | 50 |
| TMR2 | Timer2 Module Register | | | | | | | | 50 |
| T2CON | — | T2OUTPS3 | T2OUTPS2 | T2OUTPS1 | T2OUTPS0 | TMR2ON | T2CKPS1 | T2CKPS0 | 50 |
| PR2 | Timer2 Period Register | | | | | | | | 50 |
| TMR3L | Holding Register for the Least Significant Byte of the 16-bit TMR3 Register | | | | | | | | 51 |
| TMR3H | Holding Register for the Most Significant Byte of the 16-bit TMR3 Register | | | | | | | | 51 |
| T3CON | RD16 | T3ECCP1(1) | T3CKPS1 | T3CKPS0 | T3CCP1(1) | $\overline{T3SYNC}$ | TMR3CS | TMR3ON | 51 |
| ECCPR1L(2) | Enhanced Capture/Compare/PWM Register 1 (LSB) | | | | | | | | 51 |
| ECCPR1H(2) | Enhanced Capture/Compare/PWM Register 1 (MSB) | | | | | | | | 51 |
| ECCP1CON(2) | EPWM1M1 | EPWM1M0 | EDC1B1 | EDC1B0 | ECCP1M3 | ECCP1M2 | ECCP1M1 | ECCP1M0 | 51 |
| ECCP1AS(2) | ECCPASE | ECCPAS2 | ECCPAS1 | ECCPAS0 | PSSAC1 | PSSAC0 | PSSBD1(2) | PSSBD0(2) | 51 |
| ECCP1DEL(2) | PRSEN | PDC6(2) | PDC5(2) | PDC4(2) | PDC3(2) | PDC2(2) | PDC1(2) | PDC0(2) | 51 |

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used during ECCP1 operation.

**Note 1:** These bits are available on PIC18F4X8X devices only.

**2:** These bits or registers are unimplemented in PIC18F2X8X devices; always maintain these bit clear.

**3:** These bits are available on PIC18F4X8X and reserved on PIC18F2X8X devices.

### 17.4.4 CLOCK STRETCHING

Both 7 and 10-bit Slave modes implement automatic clock stretching during a transmit sequence.

The SEN bit (SSPCON2<0>) allows clock stretching to be enabled during receives. Setting SEN will cause the SCL pin to be held low at the end of each data receive sequence.

#### 17.4.4.1 Clock Stretching for 7-bit Slave Receive Mode (SEN = 1)

In 7-bit Slave Receive mode, on the falling edge of the ninth clock at the end of the ACK sequence if the BF bit is set, the CKP bit in the SSPCON1 register is automatically cleared, forcing the SCL output to be held low. The CKP being cleared to '0' will assert the SCL line low. The CKP bit must be set in the user's ISR before reception is allowed to continue. By holding the SCL line low, the user has time to service the ISR and read the contents of the SSPBUF before the master device can initiate another receive sequence. This will prevent buffer overruns from occurring (see Figure 17-13).

| | |
|---|---|
| **Note 1:** | If the user reads the contents of the SSPBUF before the falling edge of the ninth clock, thus clearing the BF bit, the CKP bit will not be cleared and clock stretching will not occur. |
| **2:** | The CKP bit can be set in software regardless of the state of the BF bit. The user should be careful to clear the BF bit in the ISR before the next receive sequence in order to prevent an overflow condition. |

#### 17.4.4.2 Clock Stretching for 10-bit Slave Receive Mode (SEN = 1)

In 10-bit Slave Receive mode during the address sequence, clock stretching automatically takes place but CKP is not cleared. During this time, if the UA bit is set after the ninth clock, clock stretching is initiated. The UA bit is set after receiving the upper byte of the 10-bit address and following the receive of the second byte of the 10-bit address with the R/$\overline{W}$ bit cleared to '0'. The release of the clock line occurs upon updating SSPADD. Clock stretching will occur on each data receive sequence as described in 7-bit mode.

| | |
|---|---|
| **Note:** | If the user polls the UA bit and clears it by updating the SSPADD register before the falling edge of the ninth clock occurs and if the user hasn't cleared the BF bit by reading the SSPBUF register before that time, then the CKP bit will still NOT be asserted low. Clock stretching on the basis of the state of the BF bit only occurs during a data sequence, not an address sequence. |

#### 17.4.4.3 Clock Stretching for 7-bit Slave Transmit Mode

7-bit Slave Transmit mode implements clock stretching by clearing the CKP bit after the falling edge of the ninth clock if the BF bit is clear. This occurs regardless of the state of the SEN bit.

The user's ISR must set the CKP bit before transmission is allowed to continue. By holding the SCL line low, the user has time to service the ISR and load the contents of the SSPBUF before the master device can initiate another transmit sequence (see Figure 17-9).

| | |
|---|---|
| **Note 1:** | If the user loads the contents of SSPBUF, setting the BF bit before the falling edge of the ninth clock, the CKP bit will not be cleared and clock stretching will not occur. |
| **2:** | The CKP bit can be set in software regardless of the state of the BF bit. |

#### 17.4.4.4 Clock Stretching for 10-bit Slave Transmit Mode

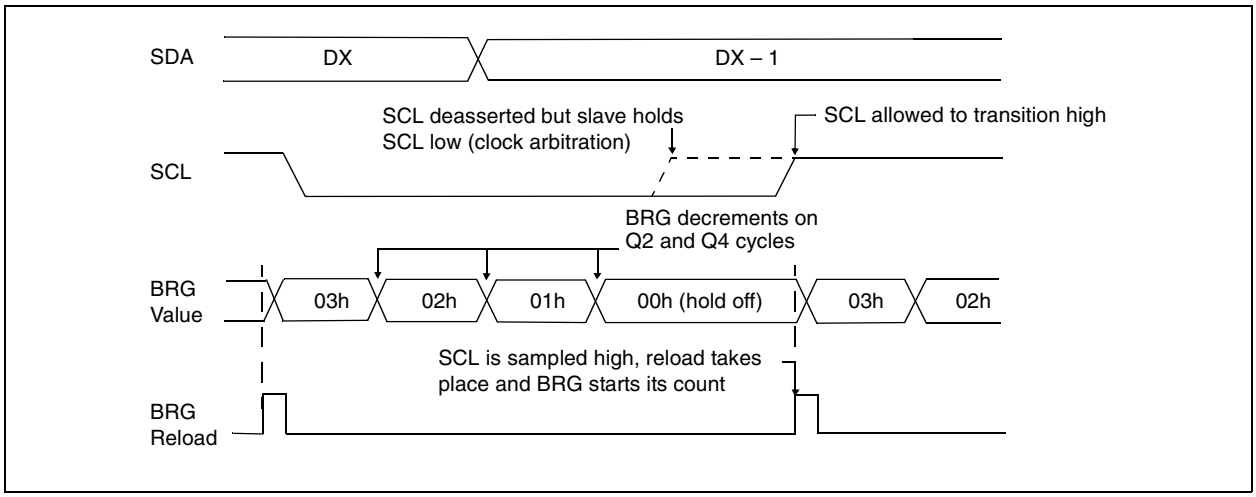
In 10-bit Slave Transmit mode, clock stretching is controlled during the first two address sequences by the state of the UA bit, just as it is in 10-bit Slave Receive mode. The first two addresses are followed by a third address sequence which contains the high-order bits of the 10-bit address and the R/$\overline{W}$ bit set to '1'. After the third address sequence is performed, the UA bit is not set, the module is now configured in Transmit mode and clock stretching is controlled by the BF flag as in 7-bit Slave Transmit mode (see Figure 17-11).

#### 17.4.7.1 Clock Arbitration

Clock arbitration occurs when the master, during any receive, transmit or Repeated Start/Stop condition, deasserts the SCL pin (SCL allowed to float high). When the SCL pin is allowed to float high, the Baud Rate Generator (BRG) is suspended from counting until the SCL pin is actually sampled high. When the SCL pin is sampled high, the Baud Rate Generator is reloaded with the contents of SSPADD<6:0> and begins counting. This ensures that the SCL high time will always be at least one BRG rollover count in the event that the clock is held low by an external device (Figure 17-18).

**FIGURE 17-18: BAUD RATE GENERATOR TIMING WITH CLOCK ARBITRATION**



#### 17.4.17.3 Bus Collision During a Stop Condition

Bus collision occurs during a Stop condition if:

a) After the SDA pin has been deasserted and allowed to float high, SDA is sampled low after the BRG has timed out.

b) After the SCL pin is deasserted, SCL is sampled low before SDA goes high.

The Stop condition begins with SDA asserted low. When SDA is sampled low, the SCL pin is allowed to float. When the pin is sampled high (clock arbitration), the Baud Rate Generator is loaded with SSPADD<6:0> and counts down to 0. After the BRG times out, SDA is sampled. If SDA is sampled low, a bus collision has occurred. This is due to another master attempting to drive a data '0' (Figure 17-31). If the SCL pin is sampled low before SDA is allowed to float high, a bus collision occurs. This is another case of another master attempting to drive a data '0' (Figure 17-32).

**FIGURE 17-31: BUS COLLISION DURING A STOP CONDITION (CASE 1)**

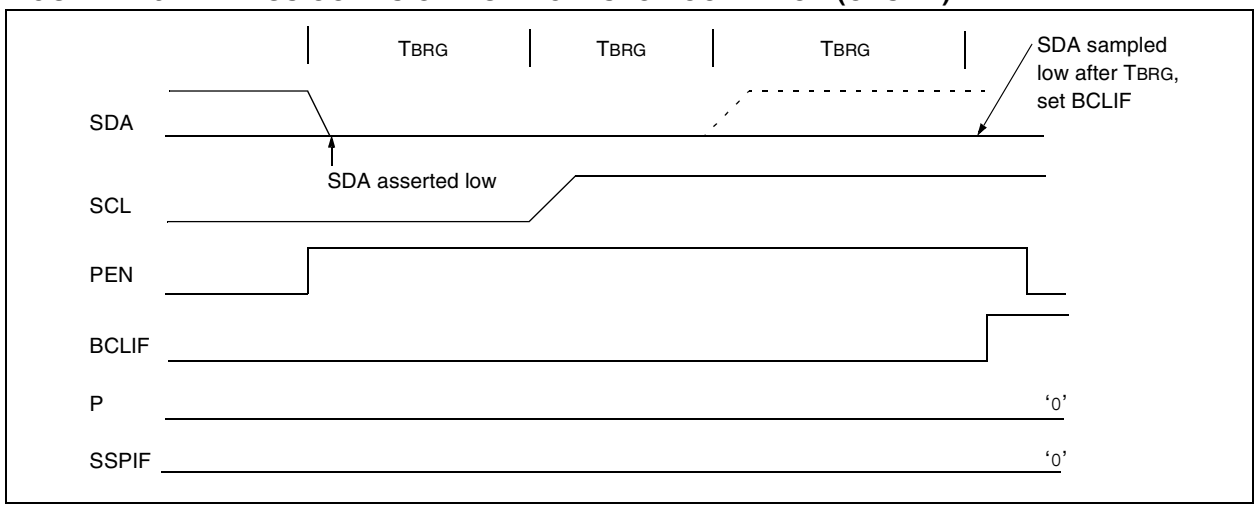
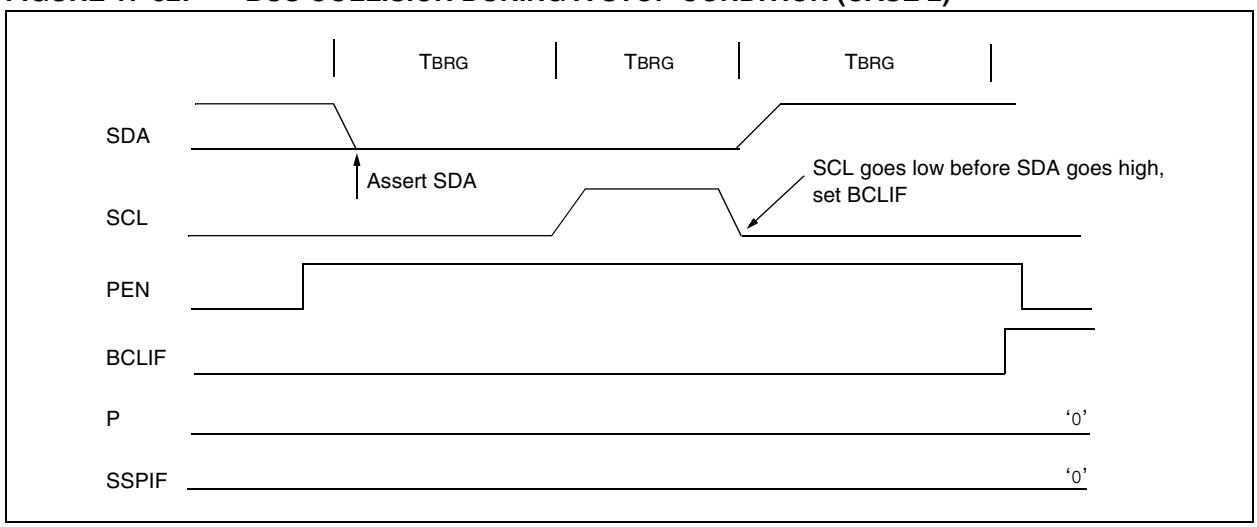



**FIGURE 17-32: BUS COLLISION DURING A STOP CONDITION (CASE 2)**



## 18.2 EUSART Asynchronous Mode

The Asynchronous mode of operation is selected by clearing the SYNC bit (TXSTA<4>). In this mode, the EUSART uses standard Non-Return-to-Zero (NRZ) format (one Start bit, eight or nine data bits and one Stop bit). The most common data format is 8 bits. An on-chip dedicated 8-bit/16-bit Baud Rate Generator can be used to derive standard baud rate frequencies from the oscillator.

The EUSART transmits and receives the LSb first. The EUSART's transmitter and receiver are functionally independent but use the same data format and baud rate. The Baud Rate Generator produces a clock, either x16 or x64 of the bit shift rate depending on the BRGH and BRG16 bits (TXSTA<2> and BAUDCON<3>). Parity is not supported by the hardware, but can be implemented in software and stored as the 9th data bit.

When operating in Asynchronous mode, the EUSART module consists of the following important elements:

- Baud Rate Generator
- Sampling Circuit
- Asynchronous Transmitter
- Asynchronous Receiver
- Auto-Wake-up on Sync Break Character
- 12-bit Break Character Transmit
- Auto-Baud Rate Detection

### 18.2.1 EUSART ASYNCHRONOUS TRANSMITTER

The EUSART transmitter block diagram is shown in Figure 18-3. The heart of the transmitter is the Transmit (Serial) Shift Register (TSR). The Shift register obtains its data from the Read/Write Transmit Buffer register, TXREG. The TXREG register is loaded with data in software. The TSR register is not loaded until the Stop bit has been transmitted from the previous load. As soon as the Stop bit is transmitted, the TSR is loaded with new data from the TXREG register (if available).

Once the TXREG register transfers the data to the TSR register (occurs in one TCY), the TXREG register is empty and the TXIF flag bit (PIR1<4>) is set. This interrupt can be enabled or disabled by setting or clearing the interrupt enable bit, TXIE (PIE1<4>). TXIF will be set regardless of the state of TXIE; it cannot be cleared in software. TXIF is also not cleared immediately upon loading TXREG, but becomes valid in the second instruction cycle following the load instruction. Polling TXIF immediately following a load of TXREG will return invalid results.

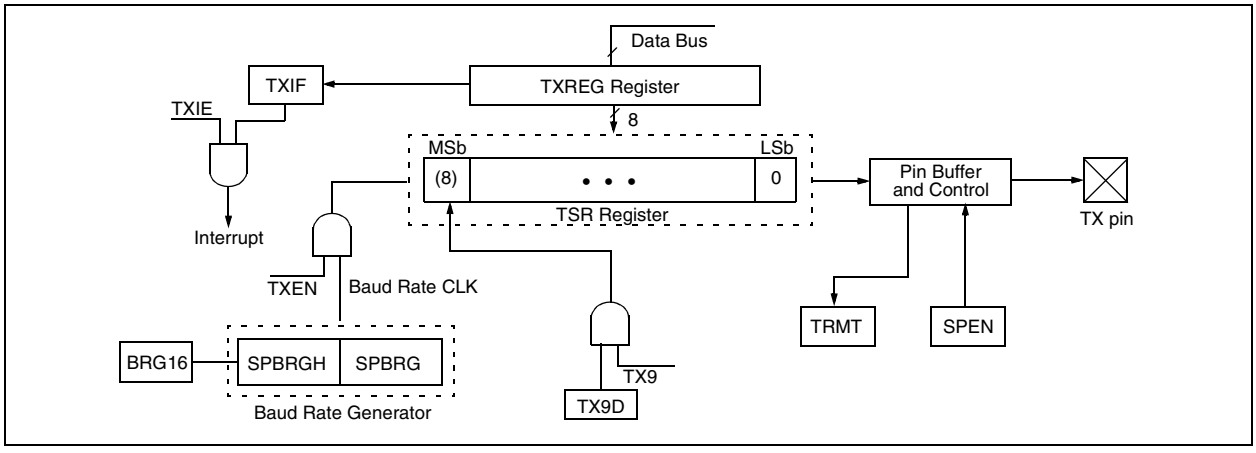
While TXIF indicates the status of the TXREG register, another bit, TRMT (TXSTA<1>), shows the status of the TSR register. TRMT is a read-only bit which is set when the TSR register is empty. No interrupt logic is tied to this bit so the user has to poll this bit in order to determine if the TSR register is empty.

| | |
|---|---|
| **Note 1:** | The TSR register is not mapped in data memory so it is not available to the user. |
| **2:** | Flag bit TXIF is set when enable bit TXEN is set. |

To set up an Asynchronous Transmission:

1. Initialize the SPBRGH:SPBRG registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the asynchronous serial port by clearing bit SYNC and setting bit SPEN.
3. If interrupts are desired, set enable bit TXIE.
4. If 9-bit transmission is desired, set transmit bit TX9. Can be used as address/data bit.
5. Enable the transmission by setting bit TXEN which will also set bit TXIF.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Load data to the TXREG register (starts transmission).
8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

**FIGURE 18-3: EUSART TRANSMIT BLOCK DIAGRAM**



**REGISTER 19-3: ADCON2: A/D CONTROL REGISTER 2**

| R/W-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| ADFM | — | ACQT2 | ACQT1 | ACQT0 | ADCS2 | ADCS1 | ADCS0 |
| bit 7 | | | | | | | bit 0 |

bit 7 **ADFM:** A/D Result Format Select bit
1 = Right justified
0 = Left justified

bit 6 **Unimplemented:** Read as '0'

bit 5-3 **ACQT2:ACQT0:** A/D Acquisition Time Select bits
111 = 20 TAD
110 = 16 TAD
101 = 12 TAD
100 = 8 TAD
011 = 6 TAD
010 = 4 TAD
001 = 2 TAD
000 = 0 TAD(1)

bit 2-0 **ADCS2:ADCS0:** A/D Conversion Clock Select bits
111 = FRC (clock derived from A/D RC oscillator)(1)
110 = FOSC/64
101 = FOSC/16
100 = FOSC/4
011 = FRC (clock derived from A/D RC oscillator)(1)
010 = FOSC/32
001 = FOSC/8
000 = FOSC/2

**Note 1:** If the A/D FRC clock source is selected, a delay of one TCY (instruction cycle) is added before the A/D clock starts. This allows the SLEEP instruction to be executed before starting a conversion.

| **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 |

## 21.0 COMPARATOR VOLTAGE REFERENCE MODULE

The comparator voltage reference is a 16-tap resistor ladder network that provides a selectable reference voltage. Although its primary purpose is to provide a reference for the analog comparators, it may also be used independently of them.

A block diagram is of the module shown in Figure 21-1. The resistor ladder is segmented to provide two ranges of CVREF values and has a power-down function to conserve power when the reference is not being used. The module's supply reference can be provided from either device VDD/VSS or an external voltage reference.

### 21.1 Configuring the Comparator Voltage Reference

The voltage reference module is controlled through the CVRCON register (Register 21-1). The comparator voltage reference provides two ranges of output voltage, each with 16 distinct levels. The range to be used is selected by the CVRR bit (CVRCON<5>). The primary difference between the ranges is the size of the steps selected by the CVREF Selection bits (CVR3:CVR0), with one range offering finer resolution. The equations used to calculate the output of the comparator voltage reference are as follows:

If CVRR = 1:
CVREF = ((CVR3:CVR0)/24) x CVRSRC

If CVRR = 0:
CVREF = (CVDD x 1/4) + (((CVR3:CVR0)/32) x CVRSRC)

The comparator reference supply voltage can come from either VDD and VSS, or the external VREF+ and VREF- that are multiplexed with RA2 and RA3. The voltage source is selected by the CVRSS bit (CVRCON<4>).

The settling time of the comparator voltage reference must be considered when changing the CVREF output (see Table 27-3 in **Section 27.0 "Electrical Characteristics"**).

**REGISTER 21-1: CVRCON: COMPARATOR VOLTAGE REFERENCE CONTROL REGISTER**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| CVREN | CVROE(1) | CVRR | CVRSS | CVR3 | CVR2 | CVR1 | CVR0 |
| bit 7 | | | | | | | bit 0 |

bit 7 **CVREN**: Comparator Voltage Reference Enable bit
1 = CVREF circuit powered on
0 = CVREF circuit powered down

bit 6 **CVROE**: Comparator VREF Output Enable bit(1)
1 = CVREF voltage level is also output on the RA0/AN0/CVREF pin
0 = CVREF voltage is disconnected from the RA0/AN0/CVREF pin

**Note 1:** CVROE overrides the TRISA<0> bit setting. If enabled for output, RA2 must also be configured as an input by setting TRISA<2> to '1'.

bit 5 **CVRR**: Comparator VREF Range Selection bit
1 = 0.00 CVRSRC to 0.75 CVRSRC, with CVRSRC/24 step size
0 = 0.25 CVRSRC to 0.75 CVRSRC, with CVRSRC/32 step size

bit 4 **CVRSS**: Comparator VREF Source Selection bit
1 = Comparator reference source, CVRSRC = (VREF+) – (VREF-)
0 = Comparator reference source, CVRSRC = VDD – VSS

bit 3-0 **CVR3:CVR0:** Comparator VREF Value Selection bits (0 ≤ (CVR3:CVR0) ≤ 15)
When CVRR = 1:
CVREF = ((CVR3:CVR0)/24) • (CVRSRC)
When CVRR = 0:
CVREF = (CVRSRC/4) + ((CVR3:CVR0)/32) • (CVRSRC)

| 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 |

**NOTES:**

## 23.2 CAN Module Registers

| **Note:** | Not all CAN registers are available in the access bank. |
|---|---|

There are many control and data registers associated with the CAN module. For convenience, their descriptions have been grouped into the following sections:

- Control and Status Registers
- Dedicated Transmit Buffer Registers
- Dedicated Receive Buffer Registers
- Programmable TX/RX and Auto RTR Buffers
- Baud Rate Control Registers
- I/O Control Register
- Interrupt Status and Control Registers

Detailed descriptions of each register and their usage are described in the following sections.

### 23.2.1 CAN CONTROL AND STATUS REGISTERS

The registers described in this section control the overall operation of the CAN module and show its operational status.

**REGISTER 23-19: RXBnDLC: RECEIVE BUFFER n DATA LENGTH CODE REGISTERS [0 ≤ n ≤ 1]**

| U-0 | R-x | R-x | R-x | R-x | R-x | R-x | R-x |
|---|---|---|---|---|---|---|---|
| — | RXRTR | RB1 | RB0 | DLC3 | DLC2 | DLC1 | DLC0 |
| bit 7 | | | | | | | bit 0 |

bit 7 **Unimplemented:** Read as '0'

bit 6 **RXRTR:** Receiver Remote Transmission Request bit
1 = Remote transfer request
0 = No remote transfer request

bit 5 **RB1:** Reserved bit 1
Reserved by CAN Spec and read as '0'.

bit 4 **RB0:** Reserved bit 0
Reserved by CAN Spec and read as '0'.

bit 3-0 **DLC3:DLC0:** Data Length Code bits
1111 = Invalid
1110 = Invalid
1101 = Invalid
1100 = Invalid
1011 = Invalid
1010 = Invalid
1001 = Invalid
1000 = Data length = 8 bytes
0111 = Data length = 7 bytes
0110 = Data length = 6 bytes
0101 = Data length = 5 bytes
0100 = Data length = 4 bytes
0011 = Data length = 3 bytes
0010 = Data length = 2 bytes
0001 = Data length = 1 bytes
0000 = Data length = 0 bytes

| **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 |

**REGISTER 23-20: RXBnDm: RECEIVE BUFFER n DATA FIELD BYTE m REGISTERS [0 ≤ n ≤ 1, 0 ≤ m ≤ 7]**

| R-x | R-x | R-x | R-x | R-x | R-x | R-x | R-x |
|---|---|---|---|---|---|---|---|
| RXBnDm7 | RXBnDm6 | RXBnDm5 | RXBnDm4 | RXBnDm3 | RXBnDm2 | RXBnDm1 | RXBnDm0 |
| bit 7 | | | | | | | bit 0 |

bit 7-0 **RXBnDm7:RXBnDm0:** Receive Buffer n Data Field Byte m bits (where 0 ≤ n < 1 and 0 < m < 7)
Each receive buffer has an array of registers. For example, Receive Buffer 0 has 8 registers: RXB0D0 to RXB0D7.

| **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 |

**COMF** **Complement f**

Syntax: COMF f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: $(\bar{f}) \rightarrow$ dest

Status Affected: N, Z

Encoding:

| | | | |
|---|---|---|---|
| 0001 | 11da | ffff | ffff |

Description: The contents of register 'f' are complemented. If 'd' is '1', the result is stored in W. If 'd' is '0', the result is stored back in register 'f' (default).

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 25.2.3 "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 to destination |

Example: `COMF REG, 0, 0`

Before Instruction
REG = 13h
After Instruction
REG = 13h
W = ECh

**CPFSEQ** **Compare f with W, Skip if f = W**

Syntax: CPFSEQ f {,a}

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: (f) – (W),
skip if (f) = (W)
(unsigned comparison)

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 0110 | 001a | ffff | ffff |

Description: Compares the contents of data memory location 'f' to the contents of W by performing an unsigned subtraction.

If 'f' = W, then the fetched instruction is discarded and a NOP is executed instead, making this a two-cycle instruction.

If 'a' is '0', the Access Bank is selected. If 'a' is '0', the BSR is used to select the GPR bank (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 25.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)

**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 |

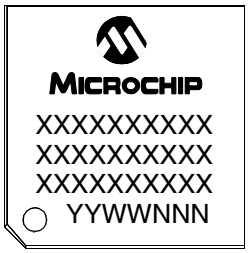
Example:
```
HERE    CPFSEQ REG, 0
NEQUAL  :
EQUAL   :
```

Before Instruction
PC Address = HERE
W = ?
REG = ?
After Instruction
If REG = W;
PC = Address (EQUAL)
If REG ≠ W;
PC = Address (NEQUAL)

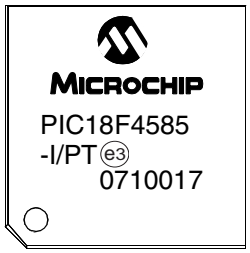
 

### 29.1 Package Marking Information (Continued)

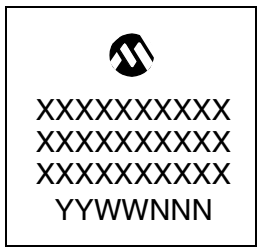
44-Lead TQFP




Example




44-Lead QFN




Example