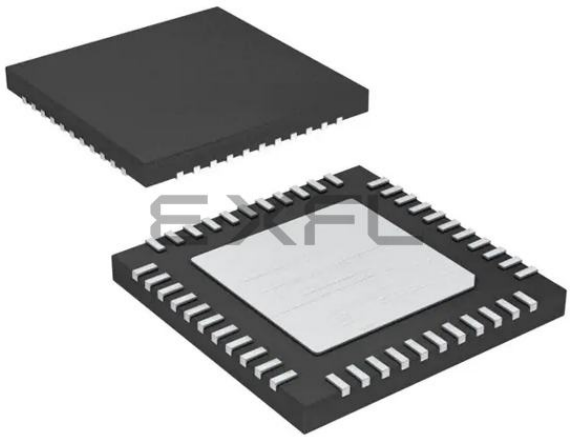


Welcome to E-XFL.COM

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	25MHz
Connectivity	CANbus, I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, HLVD, POR, PWM, WDT
Number of I/O	36
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)	4.2V ~ 5.5V
Data Converters	A/D 11x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°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/pic18f4585-e-ml

3.5.4 EXIT WITHOUT AN OSCILLATOR START-UP DELAY

Certain exits from power managed modes do not invoke the OST at all. There are two cases:

- PRI_IDLE mode where the primary clock source is not stopped; and
- the primary clock source is not any of the LP, XT, HS or HSPLL modes.

In these instances, the primary clock source either does not require an oscillator start-up delay, since it is already running (PRI_IDLE), or normally does not require an oscillator start-up delay (RC, EC and INTIO Oscillator modes). However, a fixed delay of interval TCSD following the wake event is still required when leaving Sleep and Idle modes to allow the CPU to prepare for execution. Instruction execution resumes on the first clock cycle following this delay.

TABLE 3-2: EXIT DELAY ON WAKE-UP BY RESET FROM SLEEP MODE OR ANY IDLE MODE (BY CLOCK SOURCES)

Clock Source Before Wake-up	Clock Source After Wake-up	Exit Delay	Clock Ready Status Bit (OSCCON)
Primary Device Clock (PRI_IDLE mode)	LP, XT, HS	T_{CSD}(2)	OSTS
	HSPLL		
	EC, RC		
	INTRC(1)		—
	INTOSC(3)		IOFS
T1OSC or INTRC(1)	LP, XT, HS	T_{OST}(4)	OSTS
	HSPLL	$T_{OST} + t_{rc}$(4)	
	EC, RC	T_{CSD}(2)	
	INTRC(1)		—
	INTOSC(2)	T_{IOBST}(5)	IOFS
INTOSC(3)	LP, XT, HS	T_{OST}(5)	OSTS
	HSPLL	$T_{OST} + t_{rc}$(4)	
	EC, RC	T_{CSD}(2)	
	INTRC(1)		—
	INTOSC(2)	None	IOFS
None (Sleep mode)	LP, XT, HS	T_{OST}(4)	OSTS
	HSPLL	$T_{OST} + t_{rc}$(4)	
	EC, RC	T_{CSD}(2)	
	INTRC(1)		—
	INTOSC(2)	T_{IOBST}(5)	IOFS

Note 1: In this instance, refers specifically to the 31 kHz INTRC clock source.

2: TCSD (parameter 38) is a required delay when waking from Sleep and all Idle modes and runs concurrently with any other required delays (see **Section 3.4 "Idle Modes"**).

3: Includes both the INTOSC 8 MHz source and postscaler derived frequencies.

4: TOST is the Oscillator Start-up Timer (parameter 32). t_{rc} is the PLL Lock-out Timer (parameter F12); it is also designated as TPLL.

5: Execution continues during TIOBST (parameter 39), the INTOSC stabilization period.

4.0 RESET

The PIC18F2585/2680/4585/4680 devices differentiate between various kinds of Reset:

a) Power-on Reset (POR)
b) $\overline{\text{MCLR}}$ Reset during normal operation
c) $\overline{\text{MCLR}}$ Reset during power managed modes
d) Watchdog Timer (WDT) Reset (during execution)
e) Programmable Brown-out Reset (BOR)
f) RESET Instruction
g) Stack Full Reset
h) Stack Underflow Reset

This section discusses Resets generated by $\overline{\text{MCLR}}$, POR and BOR and covers the operation of the various start-up timers. Stack Reset events are covered in **Section 5.1.2.4 "Stack Full and Underflow Resets"**. WDT Resets are covered in **Section 24.2 "Watchdog Timer (WDT)"**.

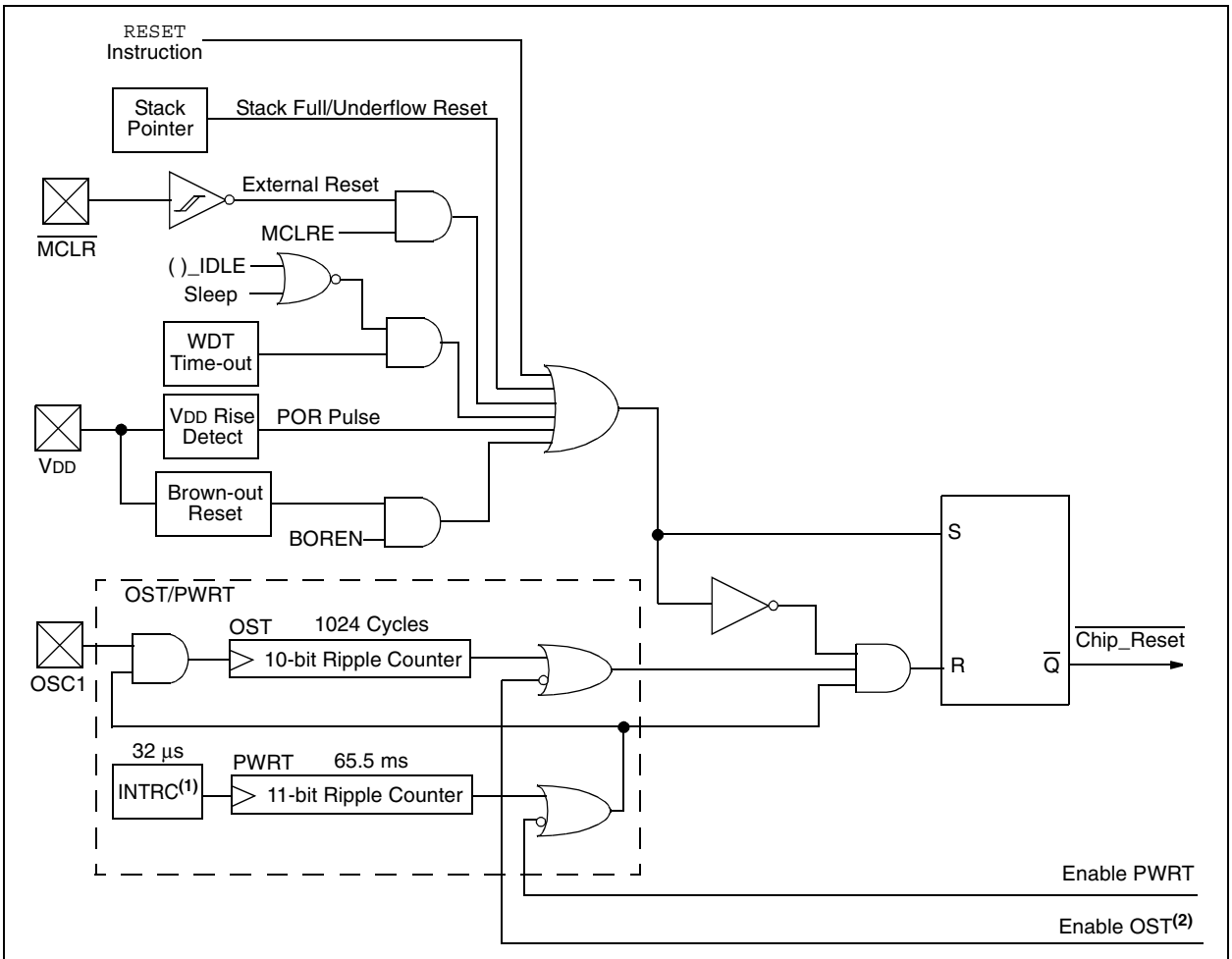
A simplified block diagram of the On-Chip Reset Circuit is shown in Figure 4-1.

4.1 RCON Register

Device Reset events are tracked through the RCON register (Register 4-1). The lower five bits of the register indicate that a specific Reset event has occurred. In most cases, these bits can only be cleared by the event and must be set by the application after the event. The state of these flag bits, taken together, can be read to indicate the type of Reset that just occurred. This is described in more detail in **Section 4.6 "Reset State of Registers"**.

The RCON register also has control bits for setting interrupt priority (IPEN) and software control of the BOR (SBOREN). Interrupt priority is discussed in **Section 9.0 "Interrupts"**. BOR is covered in **Section 4.4 "Brown-out Reset (BOR)"**.

FIGURE 4-1: SIMPLIFIED BLOCK DIAGRAM OF ON-CHIP RESET CIRCUIT



Note 1: This is the INTRC source from the internal oscillator block and is separate from the RC oscillator of the CLKI pin.
2: See Table 4-2 for time-out situations.

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
BSEL0(6)	2585	2680	4585	4680	0000 00--	0000 00--	uuuu uu--
MSEL3(6)	2585	2680	4585	4680	0000 0000	0000 0000	uuuu uuuu
MSEL2(6)	2585	2680	4585	4680	0000 0000	0000 0000	uuuu uuuu
MSEL1(6)	2585	2680	4585	4680	0000 0101	0000 0101	uuuu uuuu
MSEL0(6)	2585	2680	4585	4680	0101 0000	0101 0000	uuuu uuuu
SDFLC(6)	2585	2680	4585	4680	---0 0000	---0 0000	-u-- uuuu
RXFCON1(6)	2585	2680	4585	4680	0000 0000	0000 0000	uuuu uuuu
RXFCON0(6)	2585	2680	4585	4680	0000 0000	0000 0000	uuuu uuuu
RXFBCON7(6)	2585	2680	4585	4680	0000 0000	0000 0000	uuuu uuuu
RXFBCON6(6)	2585	2680	4585	4680	0000 0000	0000 0000	uuuu uuuu
RXFBCON5(6)	2585	2680	4585	4680	0000 0000	0000 0000	uuuu uuuu
RXFBCON4(6)	2585	2680	4585	4680	0000 0000	0000 0000	uuuu uuuu
RXFBCON3(6)	2585	2680	4585	4680	0000 0000	0000 0000	uuuu uuuu
RXFBCON2(6)	2585	2680	4585	4680	0001 0001	0001 0001	uuuu uuuu
RXFBCON1(6)	2585	2680	4585	4680	0001 0001	0001 0001	uuuu uuuu
RXFBCON0(6)	2585	2680	4585	4680	0000 0000	0000 0000	uuuu uuuu
RXF15EIDL(6)	2585	2680	4585	4680	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF15EIDH(6)	2585	2680	4585	4680	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF15SIDL(6)	2585	2680	4585	4680	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXF15SIDH(6)	2585	2680	4585	4680	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF14EIDL(6)	2585	2680	4585	4680	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF14EIDH(6)	2585	2680	4585	4680	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF14SIDL(6)	2585	2680	4585	4680	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXF14SIDH(6)	2585	2680	4585	4680	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF13EIDL(6)	2585	2680	4585	4680	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF13EIDH(6)	2585	2680	4585	4680	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF13SIDL(6)	2585	2680	4585	4680	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXF13SIDH(6)	2585	2680	4585	4680	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF12EIDL(6)	2585	2680	4585	4680	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF12EIDH(6)	2585	2680	4585	4680	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF12SIDL(6)	2585	2680	4585	4680	xxx- x-xx	uuu- u-uu	uuu- u-uu

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).

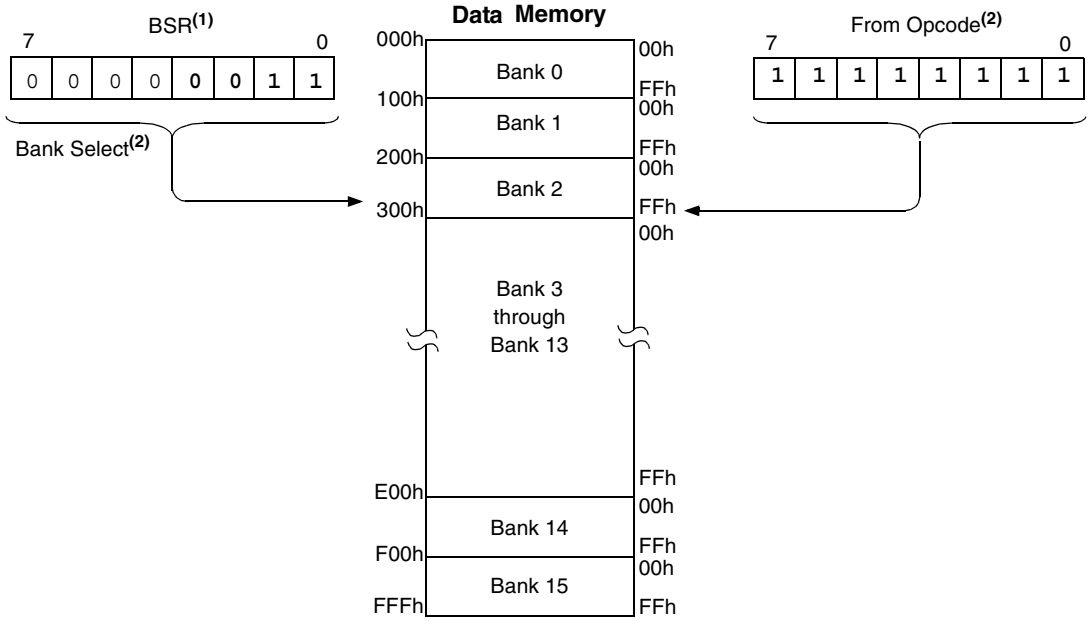
3: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

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.

FIGURE 5-6: 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. The 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 5.6.3 “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: 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:
TXB1SIDL	SID2	SID1	SID0	—	EXIDE	—	EID17	EID16	xxx- x-xx	54, 283
TXB1SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	xxxx xxxx	54, 283
TXB1CON	TXBIF	TXABT	TXLARB	TXERR	TXREQ	—	TXPRI1	TXPRI0	0000 0-00	54, 282
TXB2D7	TXB2D77	TXB2D76	TXB2D75	TXB2D74	TXB2D73	TXB2D72	TXB2D71	TXB2D70	xxxx xxxx	54, 284
TXB2D6	TXB2D67	TXB2D66	TXB2D65	TXB2D64	TXB2D63	TXB2D62	TXB2D61	TXB2D60	xxxx xxxx	54, 284
TXB2D5	TXB2D57	TXB2D56	TXB2D55	TXB2D54	TXB2D53	TXB2D52	TXB2D51	TXB2D50	xxxx xxxx	54, 284
TXB2D4	TXB2D47	TXB2D46	TXB2D45	TXB2D44	TXB2D43	TXB2D42	TXB2D41	TXB2D40	xxxx xxxx	54, 284
TXB2D3	TXB2D37	TXB2D36	TXB2D35	TXB2D34	TXB2D33	TXB2D32	TXB2D31	TXB2D30	xxxx xxxx	54, 284
TXB2D2	TXB2D27	TXB2D26	TXB2D25	TXB2D24	TXB2D23	TXB2D22	TXB2D21	TXB2D20	xxxx xxxx	54, 284
TXB2D1	TXB2D17	TXB2D16	TXB2D15	TXB2D14	TXB2D13	TXB2D12	TXB2D11	TXB2D10	xxxx xxxx	55, 284
TXB2D0	TXB2D07	TXB2D06	TXB2D05	TXB2D04	TXB2D03	TXB2D02	TXB2D01	TXB2D00	xxxx xxxx	55, 284
TXB2DLC	—	TXRTR	—	—	DLC3	DLC2	DLC1	DLC0	-x-- xxxx	55, 285
TXB2EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	xxxx xxxx	55, 284
TXB2EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	xxxx xxxx	55, 283
TXB2SIDL	SID2	SID1	SID0	—	EXIDE	—	EID17	EID16	xxxx x-xx	55, 283
TXB2SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	xxx- x-xx	55, 283
TXB2CON	TXBIF	TXABT	TXLARB	TXERR	TXREQ	—	TXPRI1	TXPRI0	0000 0-00	55, 282
RXM1EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	xxxx xxxx	55, 304
RXM1EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	xxxx xxxx	55, 304
RXM1SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	xxx- x-xx	55, 304
RXM1SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	xxxx xxxx	55, 304
RXM0EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	xxxx xxxx	55, 304
RXM0EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	xxxx xxxx	55, 304
RXM0SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	xxx- x-xx	55, 304
RXM0SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	xxxx xxxx	55, 303
RXF5EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	xxxx xxxx	55, 303
RXF5EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	xxxx xxxx	55, 303
RXF5SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	xxx- x-xx	55, 302
RXF5SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	xxxx xxxx	55, 302
RXF4EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	xxxx xxxx	55, 303
RXF4EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	xxxx xxxx	55, 303
RXF4SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	xxx- x-xx	55, 302
RXF4SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	xxxx xxxx	55, 302
RXF3EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	xxxx xxxx	55, 303
RXF3EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	xxxx xxxx	55, 303
RXF3SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	xxx- x-xx	55, 302
RXF3SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	xxxx xxxx	55, 302
RXF2EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	xxxx xxxx	55, 303
RXF2EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	xxxx xxxx	55, 303
RXF2SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	xxx- x-xx	55, 302
RXF2SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	xxxx xxxx	55, 302

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.

NOTES:

9.6 INTn Pin Interrupts

External interrupts on the RB0/INT0, RB1/INT1 and RB2/INT2 pins are edge-triggered. If the corresponding INTEDGx bit in the INTCON2 register is set (= 1), the interrupt is triggered by a rising edge; if the bit is clear, the trigger is on the falling edge. When a valid edge appears on the RBx/INTx pin, the corresponding flag bit INTxF is set. This interrupt can be disabled by clearing the corresponding enable bit INTxE. Flag bit INTxF must be cleared in software in the Interrupt Service Routine before re-enabling the interrupt.

All external interrupts (INT0, INT1 and INT2) can wake-up the processor from the power managed modes, if bit INTxE was set prior to going into power managed modes. If the Global Interrupt Enable bit, GIE, is set, the processor will branch to the interrupt vector following wake-up.

Interrupt priority for INT1 and INT2 is determined by the value contained in the interrupt priority bits, INT1IP (INTCON3<6>) and INT2IP (INTCON3<7>). There is no priority bit associated with INT0. It is always a high priority interrupt source.

9.7 TMR0 Interrupt

In 8-bit mode (which is the default), an overflow in the TMR0 register (FFh → 00h) will set flag bit TMR0IF. In 16-bit mode, an overflow in the TMR0H:TMR0L register pair (FFFFh → 0000h) will set TMR0IF. The interrupt can be enabled/disabled by setting/clearing enable bit TMR0IE (INTCON<5>). Interrupt priority for Timer0 is determined by the value contained in the interrupt priority bit, TMR0IP (INTCON2<2>). See **Section 11.0 "Timer0 Module"** for further details on the Timer0 module.

9.8 PORTB Interrupt-on-Change

An input change on PORTB<7:4> sets flag bit, RBIF (INTCON<0>). The interrupt can be enabled/disabled by setting/clearing enable bit, RBIE (INTCON<3>). Interrupt priority for PORTB interrupt-on-change is determined by the value contained in the interrupt priority bit, RBIP (INTCON2<0>).

9.9 Context Saving During Interrupts

During interrupts, the return PC address is saved on the stack. Additionally, the WREG, STATUS and BSR registers are saved on the fast return stack. If a fast return from interrupt is not used (See **Section 5.3 "Data Memory Organization"**), the user may need to save the WREG, STATUS and BSR registers on entry to the Interrupt Service Routine. Depending on the user's application, other registers may also need to be saved. Example 9-1 saves and restores the WREG, STATUS and BSR registers during an Interrupt Service Routine.

EXAMPLE 9-1: SAVING STATUS, WREG AND BSR REGISTERS IN RAM

```
MOVWF   W_TEMP                          ; W_TEMP is in virtual bank
MOVFF   STATUS, STATUS_TEMP             ; STATUS_TEMP located anywhere
MOVFF   BSR, BSR_TEMP                   ; BSR_TMEP located anywhere
;
; USER ISR CODE
;
MOVFF   BSR_TEMP, BSR                   ; Restore BSR
MOVF    W_TEMP, W                       ; Restore WREG
MOVFF   STATUS_TEMP, STATUS             ; Restore STATUS
```


TABLE 10-9: PORTE I/O SUMMARY

Pin Name	Function	I/O	TRIS	Buffer	Description
RE0/$\overline{RD}$/AN5	RE0	OUT	0	DIG	LATE<0> data output.
		IN	1	ST	PORTE<0> data input.
	$\overline{RD}$	IN	1	TTL	PSP read enable input.
	AN5	IN	1	ANA	A/D input channel 5. Enabled on POR, this analog input overrides the digital input (read as clear – low level).
RE1/$\overline{WR}$/AN6/C1OUT	RE1	OUT	0	DIG	LATE<1> data output.
		IN	1	ST	PORTE<1> data input.
	$\overline{WR}$	IN	1	TTL	PSP write enable input.
	AN6	IN	1	ANA	A/D input channel 6. Enabled on POR, this analog input overrides the digital input (read as clear – low level).
	C1OUT	OUT	0	DIG	Comparator 1 output.
RE2/$\overline{CS}$/AN7/C2OUT	RE2	OUT	0	DIG	LATE<2> data output.
		IN	1	ST	PORTE<2> data input.
	$\overline{CS}$	IN	1	TTL	PSP chip select input.
	AN7	IN	1	ANA	A/D input channel 7. Enabled on POR, this analog input overrides the digital input (read as clear – low level).
	C2OUT	OUT	0	DIG	Comparator 2 output.
$\overline{MCLR}$/VPP/RE3	$\overline{MCLR}$	IN	x	ST	External Reset input. Disabled when MCLRE Configuration bit is '1'.
	VPP	IN	x	ANA	High-voltage detection; used by ICSP™ operation.
	RE3	IN	1	ST	PORTE<3> data input. Disabled when MCLRE Configuration bit is '0'.

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

TABLE 10-10: SUMMARY OF REGISTERS ASSOCIATED WITH PORTE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
PORTE(3)	—	—	—	—	RE3(1,2)	RE2	RE1	RE0	52
LATE(2)	—	—	—	—	—	LATE Data Output Register			52
TRISE(3)	IBF	OBF	IBOV	PSPMODE	—	TRISE2	TRISE1	TRISE0	52
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	50
CMCON(3)	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	51

Legend: — = unimplemented, read as '0'. Shaded cells are not used by PORTE.

Note 1: Implemented only when Master Clear functionality is disabled (MCLRE Configuration bit = 0).

2: RE3 is the only PORTE bit implemented on both PIC18F2X8X and PIC18F4X8X devices. All other bits are implemented only when PORTE is implemented (i.e., PIC18F4X8X devices).

3: These registers are unimplemented on PIC18F2X8X devices.

15.4 PWM Mode

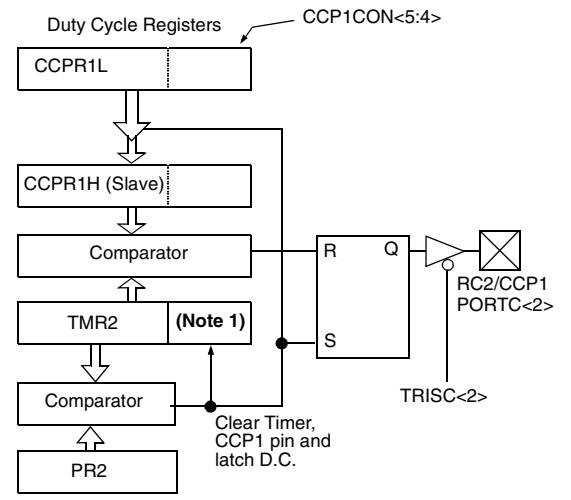
In Pulse-Width Modulation (PWM) mode, the CCP1 pin produces up to a 10-bit resolution PWM output. Since the CCP1 pin is multiplexed with a PORTB or PORTC data latch, the appropriate TRIS bit must be cleared to make the CCP1 pin an output.

Note:	Clearing the CCP1CON register will force the RC2 output latch (depending on device configuration) to the default low level. This is not the PORTC I/O data latch.

Figure 15-3 shows a simplified block diagram of the CCP1 module in PWM mode.

For a step-by-step procedure on how to set up the CCP1 module for PWM operation, see **Section 15.4.4 "Setup for PWM Operation"**.

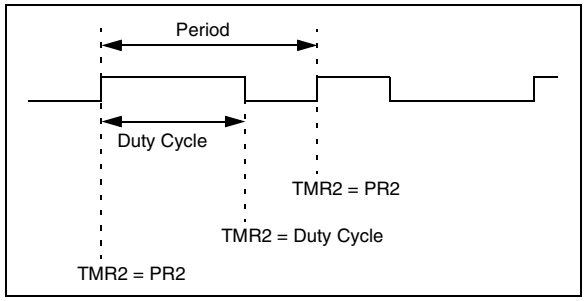
FIGURE 15-3: SIMPLIFIED PWM BLOCK DIAGRAM



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 (PR4) register. The PWM period can be calculated using the following formula.

EQUATION 15-1:

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

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

When TMR1 (TMR3) is equal to PR2 (PR2), the following three events occur on the next increment cycle:

- TMR2 is cleared
- The CCP1 pin is set (exception: if PWM duty cycle = 0%, the CCP1 pin will not be set)
- The PWM duty cycle is latched from ECCPR1L into ECCPR1H

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 ECCPR1L register and to the CCP1CON<5:4> bits. Up to 10-bit resolution is available. The ECCPR1L contains the eight MSbs and the CCP1CON<5:4> contains the two LSbs. This 10-bit value is represented by ECCPR1L:ECCP1CON<5:4>. The following equation is used to calculate the PWM duty cycle in time.

EQUATION 15-2:

$$\text{PWM Duty Cycle} = (\text{ECCPR1L:ECCP1CON<5:4>}) \cdot T_{OSC} \cdot (\text{TMR2 Prescale Value})$$

ECCPR1L and ECCP1CON<5:4> can be written to at any time, but the duty cycle value is not latched into ECCPR1H until after a match between PR2 and TMR2 occurs (i.e., the period is complete). In PWM mode, ECCPR1H is a read-only register.

16.0 ENHANCED CAPTURE/COMPARE/PWM (ECCP1) MODULE

Note:	The ECCP1 module is implemented only in PIC18F4X8X (40/44-pin) devices.

In PIC18F4585/4680 devices, ECCP1 is implemented as a standard CCP1 module with Enhanced PWM capabilities. These include the provision for 2 or 4 output channels, user selectable polarity, dead-band control and automatic shutdown and restart. The enhanced features are discussed in detail in **Section 16.4 "Enhanced PWM Mode"**. Capture, Compare and single output PWM functions of the ECCP1 module are the same as described for the standard CCP1 module.

The control register for the Enhanced CCP1 module is shown in Register 16-1. It differs from the CCP1CON register in the PIC18F2585/2680 devices in that the two Most Significant bits are implemented to control PWM functionality.

REGISTER 16-1: ECCP1CON REGISTER (ECCP1 MODULE, PIC18F4585/4680 DEVICES)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
EPWM1M1	EPWM1M0	EDC1B1	EDC1B0	ECCP1M3	ECCP1M2	ECCP1M1	ECCP1M0
bit 7							bit 0

bit 7-6 **EPWM1M1:EPWM1M0:** Enhanced PWM Output Configuration bits

If ECCP1M3:ECCP1M2 = 00, 01, 10:

xx = P1A assigned as Capture/Compare input/output; P1B, P1C, P1D assigned as port pins

If ECCP1M3:ECCP1M2 = 11:

00 = Single output: P1A modulated; P1B, P1C, P1D assigned as port pins
01 = Full-bridge output forward: P1D modulated; P1A active; P1B, P1C inactive
10 = Half-bridge output: P1A, P1B modulated with dead-band control; P1C, P1D assigned as port pins
11 = Full-bridge output reverse: P1B modulated; P1C active; P1A, P1D inactive

bit 5-4 **EDC1B1:EDC1B0**: PWM Duty Cycle bit 1 and bit 0

Capture mode:
Unused.

Compare mode:
Unused.

PWM mode:
These bits are the two LSbs of the 10-bit PWM duty cycle. The eight MSbs of the duty cycle are found in CCPR1L/ECCPR1L.

bit 3-0 **ECCP1M3:ECCP1M0**: Enhanced CCP1 Mode Select bits

0000 = Capture/Compare/PWM off (resets ECCP1 module)
0001 = Reserved
0010 = Compare mode, toggle output on match
0011 = Reserved
0100 = Capture mode, every falling edge
0101 = Capture mode, every rising edge
0110 = Capture mode, every 4th rising edge
0111 = Capture mode, every 16th rising edge
1000 = Compare mode, initialize ECCP1 pin low, set output on compare match (set ECCP1IF)
1001 = Compare mode, initialize ECCP1 pin high, clear output on compare match (set ECCP1IF)
1010 = Compare mode, generate software interrupt only, ECCP1 pin reverts to I/O state
1011 = Compare mode, trigger special event (ECCP1 resets TMR1 or TMR3, sets ECCP1IF bit and starts the A/D conversion on ECCP1 match)
1100 = PWM mode; P1A, P1C active-high; P1B, P1D active-high
1101 = PWM mode; P1A, P1C active-high; P1B, P1D active-low
1110 = PWM mode; P1A, P1C active-low; P1B, P1D active-high
1111 = PWM mode; P1A, P1C active-low; P1B, P1D active-low

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

17.4.8 I²C MASTER MODE START CONDITION TIMING

To initiate a Start condition, the user sets the Start Enable bit, SEN (SSPCON2<0>). If the SDA and SCL pins are sampled high, the Baud Rate Generator is reloaded with the contents of SSPADD<6:0> and starts its count. If SCL and SDA are both sampled high when the Baud Rate Generator times out (TBRG), the SDA pin is driven low. The action of the SDA being driven low while SCL is high is the Start condition and causes the S bit (SSPSTAT<3>) to be set. Following this, the Baud Rate Generator is reloaded with the contents of SSPADD<6:0> and resumes its count. When the Baud Rate Generator times out (TBRG), the SEN bit (SSPCON2<0>) will be automatically cleared by hardware, the Baud Rate Generator is suspended, leaving the SDA line held low and the Start condition is complete.

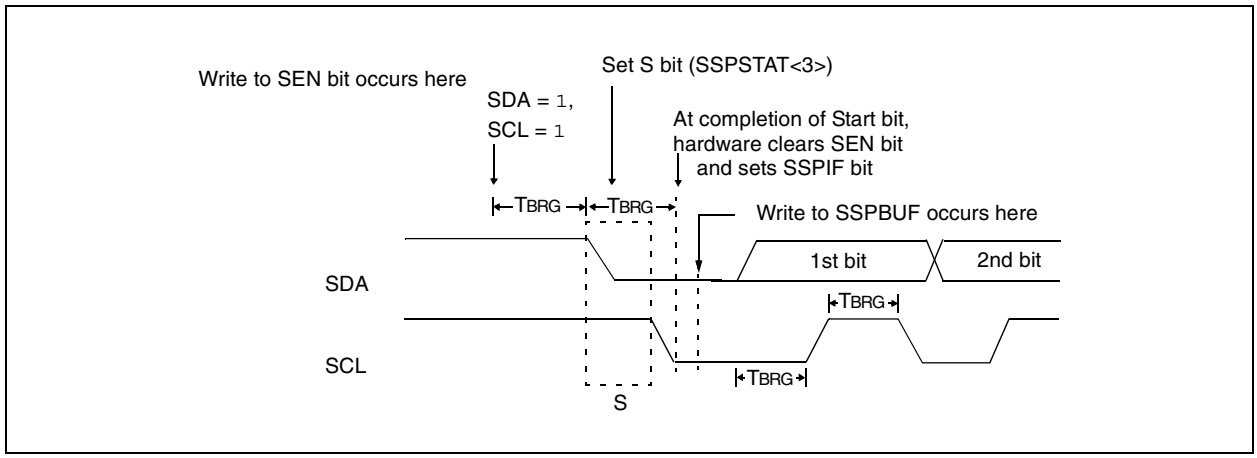
Note:	If at the beginning of the Start condition, the SDA and SCL pins are already sampled low, or if during the Start condition, the SCL line is sampled low before the SDA line is driven low, a bus collision occurs, the Bus Collision Interrupt Flag, BCLIF, is set, the Start condition is aborted and the I²C module is reset into its Idle state.

17.4.8.1 WCOL Status Flag

If the user writes the SSPBUF when a 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 to the lower 5 bits of SSPCON2 is disabled until the Start condition is complete.

FIGURE 17-19: FIRST START BIT TIMING



REGISTER 18-3: BAUDCON: BAUD RATE CONTROL REGISTER

R/W-0	R-1	U-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN
bit 7							bit 0

bit 7 **ABDOVF:** Auto-Baud Acquisition Rollover Status bit
1 = A BRG rollover has occurred during Auto-Baud Rate Detect mode (must be cleared in software)
0 = No BRG rollover has occurred

bit 6 **RCIDL:** Receive Operation Idle Status bit
1 = Receive operation is Idle
0 = Receive operation is active

bit 5 **Unimplemented:** Read as '0'

bit 4 **SCKP:** Synchronous Clock Polarity Select bit
Asynchronous mode:
Unused in this mode.
Synchronous mode:
1 = Idle state for clock (CK) is a high level
0 = Idle state for clock (CK) is a low level

bit 3 **BRG16:** 16-bit Baud Rate Register Enable bit
1 = 16-bit Baud Rate Generator – SPBRGH and SPBRG
0 = 8-bit Baud Rate Generator – SPBRG only (Compatible mode), SPBRGH value ignored

bit 2 **Unimplemented:** Read as '0'

bit 1 **WUE:** Wake-up Enable bit
Asynchronous mode:
1 = EUSART will continue to sample the RX pin – interrupt generated on falling edge; bit cleared in hardware on following rising edge
0 = RX pin not monitored or rising edge detected
Synchronous mode:
Unused in this mode.

bit 0 **ABDEN**: Auto-Baud Detect Enable bit
Asynchronous mode:
1 = Enable baud rate measurement on the next character. Requires reception of a Sync field (55h); cleared in hardware upon completion
0 = Baud rate measurement disabled or completed
Synchronous mode:
Unused in this mode.

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

22.0 HIGH/LOW-VOLTAGE DETECT (HLVD)

PIC18F2585/2680/4585/4680 devices have a High/Low-Voltage Detect module (HLVD). This is a programmable circuit that allows the user to specify both a device voltage trip point and the direction of change from that point. If the device experiences an excursion past the trip point in that direction, an interrupt flag is set. If the interrupt is enabled, the program execution will branch to the interrupt vector address and the software can then respond to the interrupt.

The High/Low-Voltage Detect Control register (Register 22-1) completely controls the operation of the HLVD module. This allows the circuitry to be "turned off" by the user under software control, which minimizes the current consumption for the device.

The block diagram for the HLVD module is shown in Figure 22-1.

REGISTER 22-1: HLVDCON: HIGH/LOW-VOLTAGE DETECT CONTROL REGISTER

R/W-0	U-0	R-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-1
VDIRMAG	—	IRVST	HLVDEN	HLVDL3(1)	HLVDL2(1)	HLVDL1(1)	HLVDL0(1)
bit 7							bit 0

bit 7 **VDIRMAG:** Voltage Direction Magnitude Select bit
1 = Event occurs when voltage equals or exceeds trip point (HLVDL3:HLDVL0)
0 = Event occurs when voltage equals or falls below trip point (HLVDL3:HLVDL0)

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

bit 5 **IRVST:** Internal Reference Voltage Stable Flag bit
1 = Indicates that the voltage detect logic will generate the interrupt flag at the specified voltage range
0 = Indicates that the voltage detect logic will not generate the interrupt flag at the specified voltage range and the HLVD interrupt should not be enabled

bit 4 **HLVDEN:** High/Low-Voltage Detect Power Enable bit
1 = HLVD enabled
0 = HLVD disabled

bit 3-0 **HLVDL3:HLVDL0:** Voltage Detection Limit bits(1)
1111 = External analog input is used (input comes from the HLVDIN pin)
1110 = 4.48V-4.69V
1101 = 4.23V-4.43V
1100 = 4.01V-4.20V
1011 = 3.81V-3.99V
1010 = 3.63V-3.80V
1001 = 3.46V-3.63V
1000 = 3.31V-3.47V
0111 = 3.05V-3.19V
0110 = 2.82V-2.95V
0101 = 2.72V-2.85V
0100 = 2.54V-2.66V
0011 = 2.38V-2.49V
0010 = 2.31V-2.42V
0001 = 2.18V-2.28V
0000 = 2.12V-2.22V

Note 1: HLVDL3:HLVDL0 modes that result in a trip point below the valid operating voltage of the device are not tested.

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 23-26: BnSIDL: TX/RX BUFFER n STANDARD IDENTIFIER REGISTERS, LOW BYTE IN RECEIVE MODE [0 ≤ n ≤ 5, TXnEN (BSEL0<n>) = 0](1)

R-x	R-x	R-x	R-x	R-x	U-0	R-x	R-x
SID2	SID1	SID0	SRR	EXID	—	EID17	EID16
bit 7							bit 0

bit 7-5 **SID2:SID0:** Standard Identifier bits (if EXID = 0)
Extended Identifier bits EID20:EID18 (if EXID = 1).

bit 4 **SRR:** Substitute Remote Transmission Request bit (only when EXID = 1)
1 = Remote transmission request occurred
0 = No remote transmission request occurred

bit 3 **EXID:** Extended Identifier Enable bit
1 = Received message is an extended identifier frame (SID10:SID0 are EID28:EID18)
0 = Received message is a standard identifier frame

bit 2 **Unimplemented:** Read as '0'

bit 1-0 **EID17:EID16:** Extended Identifier bits

Note 1: These registers are available in Mode 1 and 2 only.

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 23-27: BnSIDL: TX/RX BUFFER n STANDARD IDENTIFIER REGISTERS, LOW BYTE IN TRANSMIT MODE [0 ≤ n ≤ 5, TXnEN (BSEL0<n>) = 1](1)

R/W-x	R/W-x	R/W-x	U-0	R/W-x	U-0	R/W-x	R/W-x
SID2	SID1	SID0	—	EXIDE	—	EID17	EID16
bit 7							bit 0

bit 7-5 **SID2:SID0:** Standard Identifier bits (if EXIDE = 0)
Extended Identifier bits EID20:EID18 (if EXIDE = 1).

bit 4 **Unimplemented:** Read as '0'

bit 3 **EXIDE:** Extended Identifier Enable bit
1 = Received message is an extended identifier frame (SID10:SID0 are EID28:EID18)
0 = Received message is a standard identifier frame

bit 2 **Unimplemented:** Read as '0'

bit 1-0 **EID17:EID16:** Extended Identifier bits

Note 1: These registers are available in Mode 1 and 2 only.

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 23-31: BnEIDL: TX/RX BUFFER n EXTENDED IDENTIFIER REGISTERS, LOW BYTE IN TRANSMIT MODE [0 ≤ n ≤ 5, TXnEN (BSEL<n>) = 1](1)

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0
bit 7							bit 0

bit 7-0 **EID7:EID0:** Extended Identifier bits

Note 1: These registers are available in Mode 1 and 2 only.

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 23-32: BnDm: TX/RX BUFFER n DATA FIELD BYTE m REGISTERS IN RECEIVE MODE [0 ≤ n ≤ 5, 0 ≤ m ≤ 7, TXnEN (BSEL<n>) = 0](1)

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
BnDm7	BnDm6	BnDm5	BnDm4	BnDm3	BnDm2	BnDm1	BnDm0
bit 7							bit 0

bit 7-0 **BnDm7:BnDm0:** Receive Buffer n Data Field Byte m bits (where 0 ≤ n < 3 and 0 < m < 8)
Each receive buffer has an array of registers. For example, Receive Buffer 0 has 7 registers: B0D0 to B0D7.

Note 1: These registers are available in Mode 1 and 2 only.

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 23-33: BnDm: TX/RX BUFFER n DATA FIELD BYTE m REGISTERS IN TRANSMIT MODE [0 ≤ n ≤ 5, 0 ≤ m ≤ 7, TXnEN (BSEL<n>) = 1](1)

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
BnDm7	BnDm6	BnDm5	BnDm4	BnDm3	BnDm2	BnDm1	BnDm0
bit 7							bit 0

bit 7-0 **BnDm7:BnDm0:** Transmit Buffer n Data Field Byte m bits (where 0 ≤ n < 3 and 0 < m < 8)
Each transmit buffer has an array of registers. For example, Transmit Buffer 0 has 7 registers: TXB0D0 to TXB0D7.

Note 1: These registers are available in Mode 1 and 2 only.

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 23-54: BRGCON3: BAUD RATE CONTROL REGISTER 3

R/W-0	R/W-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
WAKDIS	WAKFIL	—	—	—	SEG2PH2(1)	SEG2PH1(1)	SEG2PH0(1)
bit 7							bit 0

bit 7 **WAKDIS:** Wake-up Disable bit
1 = Disable CAN bus activity wake-up feature
0 = Enable CAN bus activity wake-up feature

bit 6 **WAKFIL:** Selects CAN bus Line Filter for Wake-up bit
1 = Use CAN bus line filter for wake-up
0 = CAN bus line filter is not used for wake-up

bit 5-3 **Unimplemented:** Read as '0'

bit 2-0 **SEG2PH2:SEG2PH0:** Phase Segment 2 Time Select bits(1)
111 = Phase Segment 2 time = 8 x TQ
110 = Phase Segment 2 time = 7 x TQ
101 = Phase Segment 2 time = 6 x TQ
100 = Phase Segment 2 time = 5 x TQ
011 = Phase Segment 2 time = 4 x TQ
010 = Phase Segment 2 time = 3 x TQ
001 = Phase Segment 2 time = 2 x TQ
000 = Phase Segment 2 time = 1 x TQ

Note 1: Ignored if SEG2PHTS bit (BRGCON2<7>) is '0'.

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 23-57: PIE3: PERIPHERAL INTERRUPT ENABLE REGISTER

Mode 0

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
IRXIE	WAKIE	ERRIE	TXB2IE	TXB1IE(1)	TXB0IE(1)	RXB1IE	RXB0IE

Mode 1, 2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
IRXIE	WAKIE	ERRIE	TXBnIE	TXB1IE(1)	TXB0IE(1)	RXBnIE	FIFOWMIE

bit 7 bit 0

bit 7 **IRXIE:** CAN Invalid Received Message Interrupt Enable bit
1 = Enable invalid message received interrupt
0 = Disable invalid message received interrupt

bit 6 **WAKIE:** CAN bus Activity Wake-up Interrupt Enable bit
1 = Enable bus activity wake-up interrupt
0 = Disable bus activity wake-up interrupt

bit 5 **ERRIE:** CAN bus Error Interrupt Enable bit
1 = Enable CAN bus error interrupt
0 = Disable CAN bus error interrupt

bit 4 When CAN is in Mode 0:
TXB2IE: CAN Transmit Buffer 2 Interrupt Enable bit
1 = Enable Transmit Buffer 2 interrupt
0 = Disable Transmit Buffer 2 interrupt
When CAN is in Mode 1 or 2:
TXBnIE: CAN Transmit Buffer Interrupts Enable bit
1 = Enable transmit buffer interrupt; individual interrupt is enabled by TXBIE and BIE0
0 = Disable all transmit buffer interrupts

bit 3 **TXB1IE:** CAN Transmit Buffer 1 Interrupt Enable bit(1)
1 = Enable Transmit Buffer 1 interrupt
0 = Disable Transmit Buffer 1 interrupt

bit 2 **TXB0IE:** CAN Transmit Buffer 0 Interrupt Enable bit(1)
1 = Enable Transmit Buffer 0 interrupt
0 = Disable Transmit Buffer 0 interrupt

bit 1 When CAN is in Mode 0:
RXB1IE: CAN Receive Buffer 1 Interrupt Enable bit
1 = Enable Receive Buffer 1 interrupt
0 = Disable Receive Buffer 1 interrupt
When CAN is in Mode 1 or 2:
RXBnIE: CAN Receive Buffer Interrupts Enable bit
1 = Enable receive buffer interrupt; individual interrupt is enabled by BIE0
0 = Disable all receive buffer interrupts

bit 0 When CAN is in Mode 0:
RXB0IE: CAN Receive Buffer 0 Interrupt Enable bit
1 = Enable Receive Buffer 0 interrupt
0 = Disable Receive Buffer 0 interrupt
When CAN is in Mode 1:
Unimplemented: Read as '0'
When CAN is in Mode 2:
FIFOWMIE: FIFO Watermark Interrupt Enable bit
1 = Enable FIFO watermark interrupt
0 = Disable FIFO watermark interrupt

Note 1: In CAN Mode 1 and 2, this bit is forced to '0'.

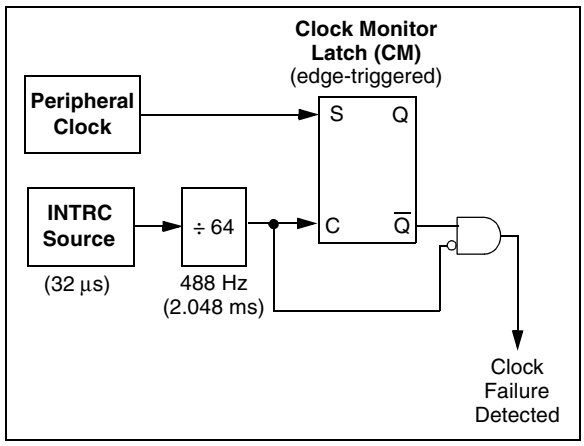
Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

24.4 Fail-Safe Clock Monitor

The Fail-Safe Clock Monitor (FSCM) allows the microcontroller to continue operation in the event of an external oscillator failure by automatically switching the device clock to the internal oscillator block. The FSCM function is enabled by setting the FCMEN Configuration bit.

When FSCM is enabled, the INTRC oscillator runs at all times to monitor clocks to peripherals and provide a backup clock in the event of a clock failure. Clock monitoring (shown in Figure 24-3) is accomplished by creating a sample clock signal, which is the INTRC output divided by 64. This allows ample time between FSCM sample clocks for a peripheral clock edge to occur. The peripheral device clock and the sample clock are presented as inputs to the Clock Monitor latch (CM). The CM is set on the falling edge of the device clock source, but cleared on the rising edge of the sample clock.

FIGURE 24-3: FSCM BLOCK DIAGRAM



Clock failure is tested for on the falling edge of the sample clock. If a sample clock falling edge occurs while CM is still set, a clock failure has been detected (Figure 24-4). This causes the following:

- the FSCM generates an oscillator fail interrupt by setting bit OSCFIF (PIR2<7>);
- the device clock source is switched to the internal oscillator block (OSCCON is not updated to show the current clock source – this is the fail-safe condition); and
- the WDT is reset.

During switchover, the postscaler frequency from the internal oscillator block may not be sufficiently stable for timing sensitive applications. In these cases, it may be desirable to select another clock configuration and enter an alternate power managed mode. This can be done to attempt a partial recovery or execute a controlled shutdown. See **Section 3.1.4 "Multiple Sleep Commands"** and **Section 24.3.1 "Special Considerations for Using Two-Speed Start-up"** for more details.

To use a higher clock speed on wake-up, the INTOSC or postscaler clock sources can be selected to provide a higher clock speed by setting bits, IRCF2:IRCF0, immediately after Reset. For wake-ups from Sleep, the INTOSC or postscaler clock sources can be selected by setting the IRCF2:IRCF0 bits prior to entering Sleep mode.

The FSCM will detect failures of the primary or secondary clock sources only. If the internal oscillator block fails, no failure would be detected, nor would any action be possible.

24.4.1 FSCM AND THE WATCHDOG TIMER

Both the FSCM and the WDT are clocked by the INTRC oscillator. Since the WDT operates with a separate divider and counter, disabling the WDT has no effect on the operation of the INTRC oscillator when the FSCM is enabled.

As already noted, the clock source is switched to the INTOSC clock when a clock failure is detected. Depending on the frequency selected by the IRCF2:IRCF0 bits, this may mean a substantial change in the speed of code execution. If the WDT is enabled with a small prescale value, a decrease in clock speed allows a WDT time-out to occur and a subsequent device Reset. For this reason, fail-safe clock events also reset the WDT and postscaler, allowing it to start timing from when execution speed was changed and decreasing the likelihood of an erroneous time-out.

24.4.2 EXITING FAIL-SAFE OPERATION

The fail-safe condition is terminated by either a device Reset or by entering a power managed mode. On Reset, the controller starts the primary clock source specified in Configuration Register 1H (with any required start-up delays that are required for the oscillator mode, such as OST or PLL timer). The INTOSC multiplexer provides the device clock until the primary clock source becomes ready (similar to a Two-Speed Start-up). The clock source is then switched to the primary clock (indicated by the OSTS bit in the OSCCON register becoming set). The Fail-Safe Clock Monitor then resumes monitoring the peripheral clock.

The primary clock source may never become ready during start-up. In this case, operation is clocked by the INTOSC multiplexer. The OSCCON register will remain in its Reset state until a power managed mode is entered.

MOVSS **Move Indexed to Indexed**

Syntax: MOVSS [z_s], [z_d]

Operands: $0 \le z_s \le 127$
$0 \le z_d \le 127$

Operation: ((FSR2) + z_s) → ((FSR2) + z_d)

Status Affected: None

Encoding:

1st word (source)	1110	1011	1zzz	$zzzz_s$
2nd word (dest.)	1111	xxxx	xzzz	$zzzz_d$

Description: The contents of the source register are moved to the destination register. The addresses of the source and destination registers are determined by adding the 7-bit literal offsets 'z_s' or 'z_d', respectively, to the value of FSR2. Both registers can be located anywhere in the 4096-byte data memory space (000h to FFFh).

The MOVSS instruction cannot use the PCL, TOSU, TOSH or TOSL as the destination register.

If the resultant source address points to an indirect addressing register, the value returned will be 00h. If the resultant destination address points to an indirect addressing register, the instruction will execute as a NOP.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Determine source addr	Determine source addr	Read source reg
Decode	Determine dest addr	Determine dest addr	Write to dest reg

Example: MOVSS [05h], [06h]

Before Instruction
FSR2 = 80h
Contents of 85h = 33h
Contents of 86h = 11h
After Instruction
FSR2 = 80h
Contents of 85h = 33h
Contents of 86h = 33h

PUSHL **Store Literal at FSR2, Decrement FSR2**

Syntax: PUSHL k

Operands: $0 \le k \le 255$

Operation: k → (FSR2),
FSR2 – 1 → FSR2

Status Affected: None

Encoding:

1111	1010	kkkk	kkkk

Description: The 8-bit literal 'k' is written to the data memory address specified by FSR2. FSR2 is decremented by 1 after the operation.

This instruction allows users to push values onto a software stack.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read 'k'	Process data	Write to destination

Example: PUSHL 08h

Before Instruction
FSR2H:FSR2L = 01ECh
Memory (01ECh) = 00h

After Instruction
FSR2H:FSR2L = 01EBh
Memory (01ECh) = 08h

25.2.3 BYTE-ORIENTED AND BIT-ORIENTED INSTRUCTIONS IN INDEXED LITERAL OFFSET MODE

Note:	Enabling the PIC18 instruction set extension may cause legacy applications to behave erratically or fail entirely.

In addition to eight new commands in the extended set, enabling the extended instruction set also enables Indexed Literal Offset Addressing mode (**Section 5.6.1 "Indexed Addressing with Literal Offset"**). This has a significant impact on the way that many commands of the standard PIC18 instruction set are interpreted.

When the extended set is disabled, addresses embedded in opcodes are treated as literal memory locations: either as a location in the Access Bank (a = 0), or in a GPR bank designated by the BSR (a = 1). When the extended instruction set is enabled and a = 0, however, a file register argument of 5Fh or less is interpreted as an offset from the pointer value in FSR2 and not as a literal address. For practical purposes, this means that all instructions that use the Access RAM bit as an argument – that is, all bit-oriented and byte-oriented instructions, or almost half of the core PIC18 instructions – may behave differently when the extended instruction set is enabled.

When the content of FSR2 is 00h, the boundaries of the Access RAM are essentially remapped to their original values. This may be useful in creating backward compatible code. If this technique is used, it may be necessary to save the value of FSR2 and restore it when moving back and forth between 'C' and assembly routines in order to preserve the Stack Pointer. Users must also keep in mind the syntax requirements of the extended instruction set (see **Section 25.2.3.1 "Extended Instruction Syntax with Standard PIC18 Commands"**).

Although the Indexed Literal Offset Addressing mode can be very useful for dynamic stack and pointer manipulation, it can also be very annoying if a simple arithmetic operation is carried out on the wrong register. Users who are accustomed to the PIC18 programming must keep in mind that, when the extended instruction set is enabled, register addresses of 5Fh or less are used for Indexed Literal Offset Addressing.

Representative examples of typical byte-oriented and bit-oriented instructions in the Indexed Literal Offset Addressing mode are provided on the following page to show how execution is affected. The operand conditions shown in the examples are applicable to all instructions of these types.

25.2.3.1 Extended Instruction Syntax with Standard PIC18 Commands

When the extended instruction set is enabled, the file register argument, 'f', in the standard byte-oriented and bit-oriented commands, is replaced with the literal offset value, 'k'. As already noted, this occurs only when 'f' is less than or equal to 5Fh. When an offset value is used, it must be indicated by square brackets ("[ ]"). As with the extended instructions, the use of brackets indicates to the compiler that the value is to be interpreted as an index or an offset. Omitting the brackets, or using a value greater than 5Fh within brackets, will generate an error in the MPASM™ Assembler.

If the index argument is properly bracketed for Indexed Literal Offset Addressing, the Access RAM argument is never specified; it will automatically be assumed to be '0'. This is in contrast to standard operation (extended instruction set disabled) when 'a' is set on the basis of the target address. Declaring the Access RAM bit in this mode will also generate an error in the MPASM Assembler.

The destination argument, 'd', functions as before.

In the latest versions of the MPASM Assembler, language support for the extended instruction set must be explicitly invoked. This is done with either the command line option, /y, or the PE directive in the source listing.

25.2.4 CONSIDERATIONS WHEN ENABLING THE EXTENDED INSTRUCTION SET

It is important to note that the extensions to the instruction set may not be beneficial to all users. In particular, users who are not writing code that uses a software stack may not benefit from using the extensions to the instruction set.

Additionally, the Indexed Literal Offset Addressing mode may create issues with legacy applications written to the PIC18 assembler. This is because instructions in the legacy code may attempt to address registers in the Access Bank below 5Fh. Since these addresses are interpreted as literal offsets to FSR2 when the instruction set extension is enabled, the application may read or write to the wrong data addresses.

When porting an application to the PIC18F2585/2680/4585/4680, it is very important to consider the type of code. A large, re-entrant application that is written in 'C' and would benefit from efficient compilation will do well when using the instruction set extensions. Legacy applications that heavily use the Access Bank will most likely not benefit from using the extended instruction set.

