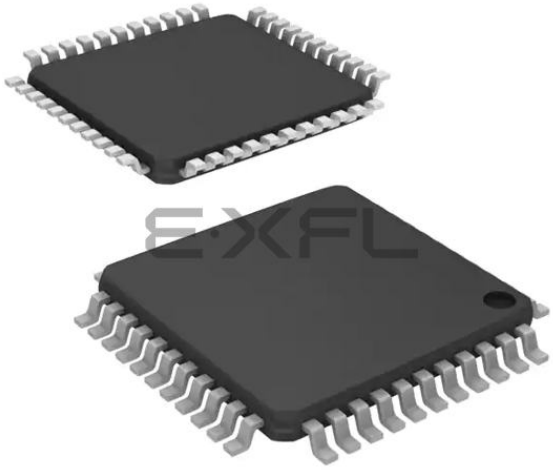


Welcome to E-XFL.COM

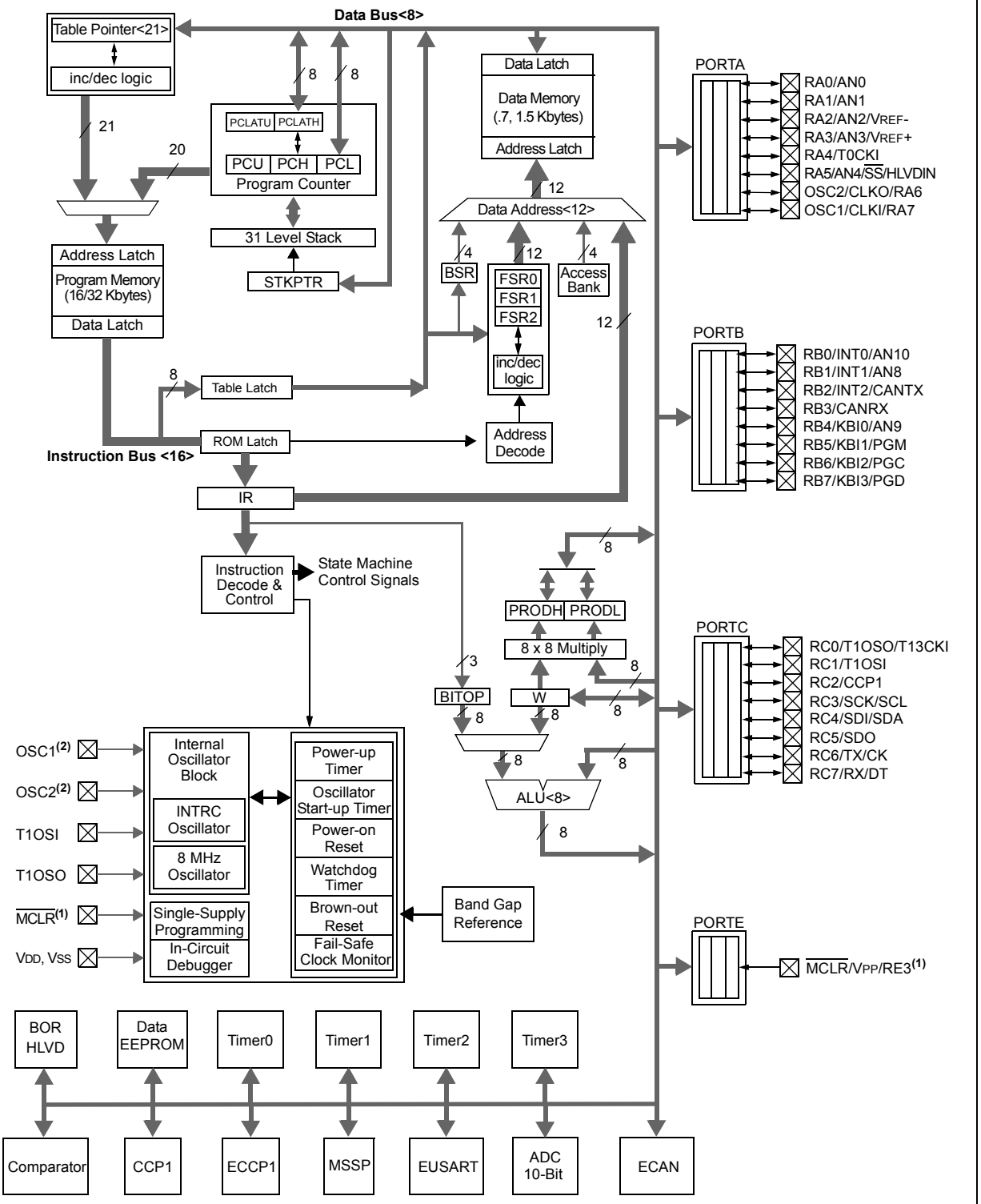
What is "Embedded - Microcontrollers"?

"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	36
Program Memory Size	32KB (16K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	1.5K x 8
Voltage - Supply (Vcc/Vdd)	4.2V ~ 5.5V
Data Converters	A/D 11x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-TQFP
Supplier Device Package	44-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f4580t-i-pt

FIGURE 1-1: PIC18F2480/2580 (28-PIN) BLOCK DIAGRAM



Note 1: RE3 is multiplexed with $\overline{\text{MCLR}}$ and is only available when the $\overline{\text{MCLR}}$ Resets are disabled.

2: OSC1/CLKI and OSC2/CLKO are only available in select oscillator modes and when these pins are not being used as digital I/O. Refer to **Section 3.0 "Oscillator Configurations"** for additional information.

TABLE 1-2: PIC18F2480/2580 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	SPDIP, SOIC	QFN			
					PORTB is a bidirectional I/O port. PORTB can be software programmed for internal weak pull-ups on all inputs.
RB0/INT0/ AN10	21	18			
RB0			I/O	TTL	Digital I/O.
INT0			I	ST	External Interrupt 0.
AN10			I	Analog	Analog Input 10.
RB1/INT1/AN8	22	19			
RB1			I/O	TTL	Digital I/O.
INT1			I	ST	External Interrupt 1.
AN8			I	Analog	Analog Input 8.
RB2/INT2/CANTX	23	20			
RB2			I/O	TTL	Digital I/O.
INT2			I	ST	External Interrupt 2.
CANTX			O	TTL	CAN bus TX.
RB3/CANRX	24	21			
RB3			I/O	TTL	Digital I/O.
CANRX			I	TTL	CAN bus RX.
RB4/KBI0/AN9	25	22			
RB4			I/O	TTL	Digital I/O.
KBI0			I	TTL	Interrupt-on-change pin.
AN9			I	Analog	Analog Input 9.
RB5/KBI1/PGM	26	23			
RB5			I/O	TTL	Digital I/O.
KBI1			I	TTL	Interrupt-on-change pin.
PGM			I/O	ST	Low-Voltage ICSP™ Programming enable pin.
RB6/KBI2/PGC	27	24			
RB6			I/O	TTL	Digital I/O.
KBI2			I	TTL	Interrupt-on-change pin.
PGC			I/O	ST	In-Circuit Debugger and ICSP programming clock pin.
RB7/KBI3/PGD	28	25			
RB7			I/O	TTL	Digital I/O.
KBI3			I	TTL	Interrupt-on-change pin.
PGD			I/O	ST	In-Circuit Debugger and ICSP programming data pin.

Legend: TTL = TTL compatible input
CMOS = CMOS compatible input or output
ST = Schmitt Trigger input with CMOS levels
I = Input
O = Output
P = Power
I^2C = I^2C™/SMBus input buffer

TABLE 5-4: INITIALIZATION CONDITIONS FOR ALL REGISTERS

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
TOSU	2480	2580	4480	4580	---0 0000	---0 0000	---0 uuuu(3)
TOSH	2480	2580	4480	4580	0000 0000	0000 0000	uuuu uuuu(3)
TOSL	2480	2580	4480	4580	0000 0000	0000 0000	uuuu uuuu(3)
STKPTR	2480	2580	4480	4580	00-0 0000	uu-0 0000	uu-u uuuu(3)
PCLATU	2480	2580	4480	4580	---0 0000	---0 0000	---u uuuu
PCLATH	2480	2580	4480	4580	0000 0000	0000 0000	uuuu uuuu
PCL	2480	2580	4480	4580	0000 0000	0000 0000	PC + 2(2)
TBLPTRU	2480	2580	4480	4580	--00 0000	--00 0000	--uu uuuu
TBLPTRH	2480	2580	4480	4580	0000 0000	0000 0000	uuuu uuuu
TBLPTRL	2480	2580	4480	4580	0000 0000	0000 0000	uuuu uuuu
TABLAT	2480	2580	4480	4580	0000 0000	0000 0000	uuuu uuuu
PRODH	2480	2580	4480	4580	xxxx xxxx	uuuu uuuu	uuuu uuuu
PRODL	2480	2580	4480	4580	xxxx xxxx	uuuu uuuu	uuuu uuuu
INTCON	2480	2580	4480	4580	0000 000x	0000 000u	uuuu uuuu(1)
INTCON2	2480	2580	4480	4580	1111 -1-1	1111 -1-1	uuuu -u-u(1)
INTCON3	2480	2580	4480	4580	11-0 0-00	11-0 0-00	uu-u u-uu(1)
INDF0	2480	2580	4480	4580	N/A	N/A	N/A
POSTINC0	2480	2580	4480	4580	N/A	N/A	N/A
POSTDEC0	2480	2580	4480	4580	N/A	N/A	N/A
PREINC0	2480	2580	4480	4580	N/A	N/A	N/A
PLUSW0	2480	2580	4480	4580	N/A	N/A	N/A
FSR0H	2480	2580	4480	4580	---- 0000	---- 0000	---- uuuu
FSR0L	2480	2580	4480	4580	xxxx xxxx	uuuu uuuu	uuuu uuuu
WREG	2480	2580	4480	4580	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF1	2480	2580	4480	4580	N/A	N/A	N/A
POSTINC1	2480	2580	4480	4580	N/A	N/A	N/A
POSTDEC1	2480	2580	4480	4580	N/A	N/A	N/A
PREINC1	2480	2580	4480	4580	N/A	N/A	N/A
PLUSW1	2480	2580	4480	4580	N/A	N/A	N/A
FSR1H	2480	2580	4480	4580	---- 0000	---- 0000	---- uuuu
FSR1L	2480	2580	4480	4580	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 5-3 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 ECAN™ technology is set up in Mode 1 or Mode 2.

6.1.2.4 Stack Full and Underflow Resets

Device Resets on stack overflow and stack underflow conditions are enabled by setting the STVREN bit in Configuration Register 4L. When STVREN is set, a full or underflow will set the appropriate STKFUL or STKUNF bit and then cause a device Reset. When STVREN is cleared, a full or underflow condition will set the appropriate STKFUL or STKUNF bit but not cause a device Reset. The STKFUL or STKUNF bits are cleared by the user software or a Power-on Reset.

6.1.3 FAST REGISTER STACK

A Fast Register Stack is provided for the STATUS, WREG and BSR registers, to provide a “fast return” option for interrupts. Each stack is only one level deep and is neither readable nor writable. It is loaded with the current value of the corresponding register when the processor vectors for an interrupt. All interrupt sources will push values into the stack registers. The values in the registers are then loaded back into their associated registers, if the `RETFIE, FAST` instruction is used to return from the interrupt.

If both low and high-priority interrupts are enabled, the stack registers cannot be used reliably to return from low-priority interrupts. If a high-priority interrupt occurs while servicing a low-priority interrupt, the stack register values stored by the low-priority interrupt will be overwritten. In these cases, users must save the key registers in software during a low-priority interrupt.

If interrupt priority is not used, all interrupts may use the Fast Register Stack for returns from interrupt. If no interrupts are used, the Fast Register Stack can be used to restore the STATUS, WREG and BSR registers at the end of a subroutine call. To use the Fast Register Stack for a subroutine call, a `CALL label, FAST` instruction must be executed to save the STATUS, WREG and BSR registers to the Fast Register Stack. A `RETURN, FAST` instruction is then executed to restore these registers from the Fast Register Stack.

Example 6-1 shows a source code example that uses the Fast Register Stack during a subroutine call and return.

EXAMPLE 6-1: FAST REGISTER STACK CODE EXAMPLE

```
CALL  SUB1, FAST     ;STATUS, WREG, BSR
                     ;SAVED IN FAST REGISTER
                     ;STACK
      •
      •

SUB1  •
      •
      RETURN, FAST   ;RESTORE VALUES SAVED
                     ;IN FAST REGISTER STACK
```

6.1.4 LOOK-UP TABLES IN PROGRAM MEMORY

There may be programming situations that require the creation of data structures, or look-up tables, in program memory. For PIC18 devices, look-up tables can be implemented in two ways:

- Computed `GOTO`
- Table Reads

6.1.4.1 Computed `GOTO`

A computed `GOTO` is accomplished by adding an offset to the program counter. An example is shown in Example 6-2.

A look-up table can be formed with an `ADDWF PCL` instruction and a group of `RETLW nn` instructions. The W register is loaded with an offset into the table before executing a call to that table. The first instruction of the called routine is the `ADDWF PCL` instruction. The next instruction executed will be one of the `RETLW nn` instructions, that returns the value ‘`nn`’ to the calling function.

The offset value (in WREG) specifies the number of bytes that the program counter should advance and should be multiples of 2 (LSb = `0`).

In this method, only one data byte may be stored in each instruction location and room on the return address stack is required.

EXAMPLE 6-2: COMPUTED `GOTO` USING AN OFFSET VALUE

```
        MOVF   OFFSET, W
        CALL   TABLE
ORG     nn00h
TABLE   ADDWF  PCL
        RETLW  nnh
        RETLW  nnh
        RETLW  nnh
        .
        .
        .
```

6.1.4.2 Table Reads and Table Writes

A better method of storing data in program memory allows two bytes of data to be stored in each instruction location.

Look-up table data may be stored two bytes per program word by using table reads and writes. The Table Pointer (TBLPTR) register specifies the byte address and the Table Latch (TABLAT) register contains the data that is read from or written to program memory. Data is transferred to or from program memory one byte at a time.

Table read and table write operations are discussed further in **Section 7.1 “Table Reads and Table Writes”**.

TABLE 6-2: REGISTER FILE SUMMARY (PIC18F2480/2580/4480/4580)

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:
TOSU	—	—	—	Top-of-Stack Upper Byte (TOS<20:16>)					---0 0000	55, 68
TOSH	Top-of-Stack High Byte (TOS<15:8>)								0000 0000	55, 68
TOSL	Top-of-Stack Low Byte (TOS<7:0>)								0000 0000	55, 68
STKPTR	STKFUL	STKUNF	—	Return Stack Pointer					00-0 0000	55, 69
PCLATU	—	—	bit 21(1)	Holding Register for PC<20:16>					---0 0000	55, 68
PCLATH	Holding Register for PC<15:8>								0000 0000	55, 68
PCL	PC Low Byte (PC<7:0>)								0000 0000	55, 68
TBLPTRU	—	—	bit 21	Program Memory Table Pointer Upper Byte (TBLPTR<20:16>)					--00 0000	55, 109
TBLPTRH	Program Memory Table Pointer High Byte (TBLPTR<15:8>)								0000 0000	55, 109
TBLPTRL	Program Memory Table Pointer Low Byte (TBLPTR<7:0>)								0000 0000	55, 109
TABLAT	Program Memory Table Latch								0000 0000	55, 109
PRODH	Product Register High Byte								xxxx xxxx	55, 117
PRODL	Product Register Low Byte								xxxx xxxx	55, 117
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	55, 121
INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	—	TMR0IP	—	RBIP	1111 -1-1	55, 122
INTCON3	INT2IP	INT1IP	—	INT2IE	INT1IE	—	INT2IF	INT1IF	11-0 0-00	55, 123
INDF0	Uses contents of FSR0 to address data memory – value of FSR0 not changed (not a physical register)								N/A	55, 96
POSTINC0	Uses contents of FSR0 to address data memory – value of FSR0 post-incremented (not a physical register)								N/A	55, 97
POSTDEC0	Uses contents of FSR0 to address data memory – value of FSR0 post-decremented (not a physical register)								N/A	55, 97
PREINC0	Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented (not a physical register)								N/A	55, 97
PLUSW0	Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented (not a physical register), value of FSR0 offset by W								N/A	55, 97
FSR0H	—	—	—	—	Indirect Data Memory Address Pointer 0 High				---- xxxx	55, 96
FSR0L	Indirect Data Memory Address Pointer 0 Low Byte								xxxx xxxx	55, 96
WREG	Working Register								xxxx xxxx	55
INDF1	Uses contents of FSR1 to address data memory – value of FSR1 not changed (not a physical register)								N/A	55, 96
POSTINC1	Uses contents of FSR1 to address data memory – value of FSR1 post-incremented (not a physical register)								N/A	55, 97
POSTDEC1	Uses contents of FSR1 to address data memory – value of FSR1 post-decremented (not a physical register)								N/A	55, 97
PREINC1	Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented (not a physical register)								N/A	55, 97
PLUSW1	Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented (not a physical register), value of FSR1 offset by W								N/A	55, 97
FSR1H	—	—	—	—	Indirect Data Memory Address Pointer 1 High				---- xxxx	55, 96
FSR1L	Indirect Data Memory Address Pointer 1 Low Byte								xxxx xxxx	55, 96
BSR	—	—	—	—	Bank Select Register				---- 0000	56, 73
INDF2	Uses contents of FSR2 to address data memory – value of FSR2 not changed (not a physical register)								N/A	56, 96
POSTINC2	Uses contents of FSR2 to address data memory – value of FSR2 post-incremented (not a physical register)								N/A	56, 97
POSTDEC2	Uses contents of FSR2 to address data memory – value of FSR2 post-decremented (not a physical register)								N/A	56, 97
PREINC2	Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented (not a physical register)								N/A	56, 97
PLUSW2	Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented (not a physical register), value of FSR2 offset by W								N/A	56, 97

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 5.4 "Brown-out Reset (BOR)"**.

3: These registers and/or bits are not implemented on PIC18F2X80 devices and are read as '0'. Reset values are shown for PIC18F4X80 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 3.6.4 "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 PIC18F4X80 devices only.

REGISTER 10-6: PIR3: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 3

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
	IRXIF	WAKIF	ERRIF	TXB2IF	TXB1IF(1)	TXB0IF(1)	RXB1IF	RXB0IF

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
	IRXIF	WAKIF	ERRIF	TXBnIF	TXB1IF(1)	TXB0IF(1)	RXBnIF	FIFOWMIF(1)
	bit 7							bit 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

bit 7 **IRXIF:** CAN Invalid Received Message Interrupt Flag bit
1 = An invalid message has occurred on the CAN bus
0 = No invalid message on CAN bus

bit 6 **WAKIF:** CAN bus Activity Wake-up Interrupt Flag bit
1 = Activity on CAN bus has occurred
0 = No activity on CAN bus

bit 5 **ERRIF:** CAN bus Error Interrupt Flag bit
1 = An error has occurred in the CAN module (multiple sources)
0 = No CAN module errors

bit 4 When CAN is in Mode 0:
TXB2IF: CAN Transmit Buffer 2 Interrupt Flag bit
1 = Transmit Buffer 2 has completed transmission of a message and may be reloaded
0 = Transmit Buffer 2 has not completed transmission of a message
When CAN is in Mode 1 or 2:
TXBnIF: Any Transmit Buffer Interrupt Flag bit
1 = One or more transmit buffers have completed transmission of a message and may be reloaded
0 = No transmit buffer is ready for reload

bit 3 **TXB1IF:** CAN Transmit Buffer 1 Interrupt Flag bit(1)
1 = Transmit Buffer 1 has completed transmission of a message and may be reloaded
0 = Transmit Buffer 1 has not completed transmission of a message

bit 2 **TXB0IF:** CAN Transmit Buffer 0 Interrupt Flag bit(1)
1 = Transmit Buffer 0 has completed transmission of a message and may be reloaded
0 = Transmit Buffer 0 has not completed transmission of a message

bit 1 When CAN is in Mode 0:
RXB1IF: CAN Receive Buffer 1 Interrupt Flag bit
1 = Receive Buffer 1 has received a new message
0 = Receive Buffer 1 has not received a new message
When CAN is in Mode 1 or 2:
RXBnIF: Any Receive Buffer Interrupt Flag bit
1 = One or more receive buffers has received a new message
0 = No receive buffer has received a new message

bit 0 When CAN is in Mode 0:
RXB0IF: CAN Receive Buffer 0 Interrupt Flag bit
1 = Receive Buffer 0 has received a new message
0 = Receive Buffer 0 has not received a new message
When CAN is in Mode 1:
Unimplemented: Read as '0'
When CAN is in Mode 2:
FIFOWMIF: FIFO Watermark Interrupt Flag bit(1)
1 = FIFO high watermark is reached
0 = FIFO high watermark is not reached

Note 1: In CAN Mode 1 and 2, these bits are forced to '0'.

TABLE 11-3: 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(1)	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: OUT = Output, IN = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Buffer Input, TTL = TTL Buffer Input

Note 1: Available on 40/44-pin devices only.

18.4.3.2 Reception

When the R/$\overline{W}$ bit of the address byte is clear and an address match occurs, the R/$\overline{W}$ bit of the SSPSTAT register is cleared. The received address is loaded into the SSPBUF register and the SDA line is held low ($\overline{ACK}$).

When the address byte overflow condition exists, then the no Acknowledge ($\overline{ACK}$) pulse is given. An overflow condition is defined as either bit, BF (SSPSTAT<0>), is set, or bit, SSPOV (SSPCON1<6>), is set.

An MSSP interrupt is generated for each data transfer byte. Flag bit, SSPIF (PIR1<3>), must be cleared in software. The SSPSTAT register is used to determine the status of the byte.

If SEN is enabled (SSPCON2<0> = 1), RC3/SCK/SCL will be held low (clock stretch) following each data transfer. The clock must be released by setting bit, CKP (SSPCON<4>). See **Section 18.4.4 "Clock Stretching"** for more details.

18.4.3.3 Transmission

When the R/$\overline{W}$ bit of the incoming address byte is set and an address match occurs, the R/$\overline{W}$ bit of the SSPSTAT register is set. The received address is loaded into the SSPBUF register. The $\overline{ACK}$ pulse will be sent on the ninth bit and pin RC3/SCK/SCL is held low regardless of SEN (see **Section 18.4.4 "Clock Stretching"** for more details). By stretching the clock, the master will be unable to assert another clock pulse until the slave is done preparing the transmit data. The transmit data must be loaded into the SSPBUF register which also loads the SSPSR register. Then, the RC3/SCK/SCL pin should be enabled by setting bit, CKP (SSPCON1<4>). The eight data bits are shifted out on the falling edge of the SCL input. This ensures that the SDA signal is valid during the SCL high time (Figure 18-9).

The $\overline{ACK}$ pulse from the master-receiver is latched on the rising edge of the ninth SCL input pulse. If the SDA line is high (not $\overline{ACK}$), then the data transfer is complete. In this case, when the $\overline{ACK}$ is latched by the slave, the slave logic is reset and the slave monitors for another occurrence of the Start bit. If the SDA line was low ($\overline{ACK}$), the next transmit data must be loaded into the SSPBUF register. Again, pin, RC3/SCK/SCL, must be enabled by setting bit, CKP.

An MSSP interrupt is generated for each data transfer byte. The SSPIF bit must be cleared in software and the SSPSTAT register is used to determine the status of the byte. The SSPIF bit is set on the falling edge of the ninth clock pulse.

REGISTER 24-9: TXBnEIDL: TRANSMIT BUFFER n EXTENDED IDENTIFIER REGISTERS, LOW BYTE [0 ≤ n ≤ 2]

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

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

bit 7-0 **EID<7:0>:** Extended Identifier bits (not used when transmitting standard identifier message)

REGISTER 24-10: TXBnDm: TRANSMIT BUFFER n DATA FIELD BYTE m REGISTERS [0 ≤ n ≤ 2, 0 ≤ m ≤ 7]

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
TXBnDm7	TXBnDm6	TXBnDm5	TXBnDm4	TXBnDm3	TXBnDm2	TXBnDm1	TXBnDm0
bit 7							bit 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

bit 7-0 **TXBnDm<7:0>:** 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.

REGISTER 24-50: MSEL2: MASK SELECT REGISTER 2(1)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
FIL11_1	FIL11_0	FIL10_1	FIL10_0	FIL9_1	FIL9_0	FIL8_1	FIL8_0
bit 7							bit 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

bit 7-6 **FIL11_<1:0>:** Filter 11 Select bits 1 and 0
- 11 = No mask
- 10 = Filter 15
- 01 = Acceptance Mask 1
- 00 = Acceptance Mask 0

bit 5-4 **FIL10_<1:0>:** Filter 10 Select bits 1 and 0
- 11 = No mask
- 10 = Filter 15
- 01 = Acceptance Mask 1
- 00 = Acceptance Mask 0

bit 3-2 **FIL9_<1:0>:** Filter 9 Select bits 1 and 0
- 11 = No mask
- 10 = Filter 15
- 01 = Acceptance Mask 1
- 00 = Acceptance Mask 0

bit 1-0 **FIL8_<1:0>:** Filter 8 Select bits 1 and 0
- 11 = No mask
- 10 = Filter 15
- 01 = Acceptance Mask 1
- 00 = Acceptance Mask 0

Note 1: This register is available in Mode 1 and 2 only.

REGISTER 24-59: TXBIE: TRANSMIT BUFFERS INTERRUPT ENABLE REGISTER(1)

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	U-0	U-0
—	—	—	TXB2IE(2)	TXB1IE(2)	TXB0IE(2)	—	—
bit 7							bit 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

bit 7-5 **Unimplemented:** Read as '0'

bit 4-2 **TXB2IE:TXB0IE:** Transmit Buffer 2-0 Interrupt Enable bits(2)
1 = Transmit buffer interrupt is enabled
0 = Transmit buffer interrupt is disabled

bit 1-0 **Unimplemented:** Read as '0'

Note 1: This register is available in Mode 1 and 2 only.
2: TXBnIE in PIE3 register must be set to get an interrupt.

REGISTER 24-60: BIE0: BUFFER INTERRUPT ENABLE REGISTER 0(1)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
B5IE(2)	B4IE(2)	B3IE(2)	B2IE(2)	B1IE(2)	B0IE(2)	RXB1IE(2)	RXB0IE(2)
bit 7							bit 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

bit 7-2 **B5IE:B0IE:** Programmable Transmit/Receive Buffer 5-0 Interrupt Enable bits(2)
1 = Interrupt is enabled
0 = Interrupt is disabled

bit 1-0 **RXB1IE:RXB0IE:** Dedicated Receive Buffer 1-0 Interrupt Enable bits(2)
1 = Interrupt is enabled
0 = Interrupt is disabled

Note 1: This register is available in Mode 1 and 2 only.
2: Either TXBnIE or RXBnIE in the PIE3 register must be set to get an interrupt.

TABLE 24-1: CAN CONTROLLER REGISTER MAP

Address[1]	Name	Address	Name	Address	Name	Address	Name
F7Fh	SPBRGH[3]	F5Fh	CANCON_RO0	F3Fh	CANCON_RO2	F1Fh	RXM1EIDL
F7Eh	BAUDCON[3]	F5Eh	CANSTAT_RO0	F3Eh	CANSTAT_RO2	F1Eh	RXM1EIDH
F7Dh	—[4]	F5Dh	RXB1D7	F3Dh	TXB1D7	F1Dh	RXM1SIDL
F7Ch	—[4]	F5Ch	RXB1D6	F3Ch	TXB1D6	F1Ch	RXM1SIDH
F7Bh	—[4]	F5Bh	RXB1D5	F3Bh	TXB1D5	F1Bh	RXM0EIDL
F7Ah	—[4]	F5Ah	RXB1D4	F3Ah	TXB1D4	F1Ah	RXM0EIDH
F79h	ECCP1DEL[3]	F59h	RXB1D3	F39h	TXB1D3	F19h	RXM0SIDL
F78h	—[4]	F58h	RXB1D2	F38h	TXB1D2	F18h	RXM0SIDH
F77h	ECANCON	F57h	RXB1D1	F37h	TXB1D1	F17h	RXF5EIDL
F76h	TXERRCNT	F56h	RXB1D0	F36h	TXB1D0	F16h	RXF5EIDH
F75h	RXERRCNT	F55h	RXB1DLC	F35h	TXB1DLC	F15h	RXF5SIDL
F74h	COMSTAT	F54h	RXB1EIDL	F34h	TXB1EIDL	F14h	RXF5SIDH
F73h	CIOCON	F53h	RXB1EIDH	F33h	TXB1EIDH	F13h	RXF4EIDL
F72h	BRGCON3	F52h	RXB1SIDL	F32h	TXB1SIDL	F12h	RXF4EIDH
F71h	BRGCON2	F51h	RXB1SIDH	F31h	TXB1SIDH	F11h	RXF4SIDL
F70h	BRGCON1	F50h	RXB1CON	F30h	TXB1CON	F10h	RXF4SIDH
F6Fh	CANCON	F4Fh	CANCON_RO1[2]	F2Fh	CANCON_RO3[2]	F0Fh	RXF3EIDL
F6Eh	CANSTAT	F4Eh	CANSTAT_RO1[2]	F2Eh	CANSTAT_RO3[2]	F0Eh	RXF3EIDH
F6Dh	RXB0D7	F4Dh	TXB0D7	F2Dh	TXB2D7	F0Dh	RXF3SIDL
F6Ch	RXB0D6	F4Ch	TXB0D6	F2Ch	TXB2D6	F0Ch	RXF3SIDH
F6Bh	RXB0D5	F4Bh	TXB0D5	F2Bh	TXB2D5	F0Bh	RXF2EIDL
F6Ah	RXB0D4	F4Ah	TXB0D4	F2Ah	TXB2D4	F0Ah	RXF2EIDH
F69h	RXB0D3	F49h	TXB0D3	F29h	TXB2D3	F09h	RXF2SIDL
F68h	RXB0D2	F48h	TXB0D2	F28h	TXB2D2	F08h	RXF2SIDH
F67h	RXB0D1	F47h	TXB0D1	F27h	TXB2D1	F07h	RXF1EIDL
F66h	RXB0D0	F46h	TXB0D0	F26h	TXB2D0	F06h	RXF1EIDH
F65h	RXB0DLC	F45h	TXB0DLC	F25h	TXB2DLC	F05h	RXF1SIDL
F64h	RXB0EIDL	F44h	TXB0EIDL	F24h	TXB2EIDL	F04h	RXF1SIDH
F63h	RXB0EIDH	F43h	TXB0EIDH	F23h	TXB2EIDH	F03h	RXF0EIDL
F62h	RXB0SIDL	F42h	TXB0SIDL	F22h	TXB2SIDL	F02h	RXF0EIDH
F61h	RXB0SIDH	F41h	TXB0SIDH	F21h	TXB2SIDH	F01h	RXF0SIDL
F60h	RXB0CON	F40h	TXB0CON	F20h	TXB2CON	F00h	RXF0SIDH

Note 1: Shaded registers are available in Access Bank low area, while the rest are available in Bank 15.

2: CANSTAT register is repeated in these locations to simplify application firmware. Unique names are given for each instance of the controller register due to the Microchip header file requirement.

3: These registers are not CAN registers.

4: Unimplemented registers are read as '0'.

25.2 Watchdog Timer (WDT)

For PIC18F2480/2580/4480/4580 devices, the WDT is driven by the INTRC source. When the WDT is enabled, the clock source is also enabled. The nominal WDT period is 4 ms and has the same stability as the INTRC oscillator.

The 4 ms period of the WDT is multiplied by a 16-bit postscaler. Any output of the WDT postscaler is selected by a multiplexer, controlled by bits in Configuration Register 2H. Available periods range from 4 ms to 131.072 seconds (2.18 minutes). The WDT and postscaler are cleared when any of the following events occur: a SLEEP or CLRWDT instruction is executed, the IRCF bits (OSCCON<6:4>) are changed or a clock failure has occurred.

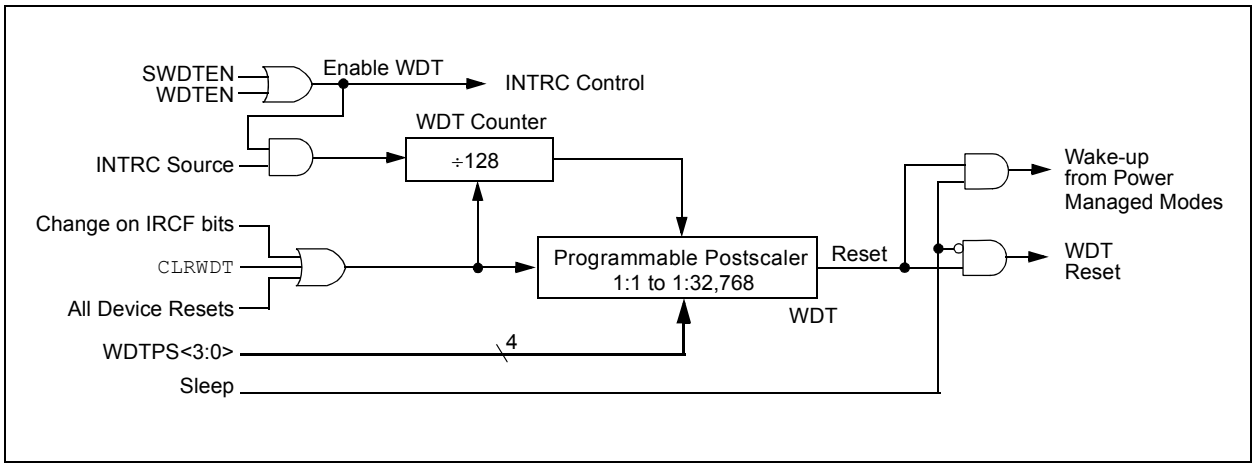
.

Note 1:	The CLRWDT and SLEEP instructions clear the WDT and postscaler counts when executed.
2:	Changing the setting of the IRCF bits (OSCCON<6:4>) clears the WDT and postscaler counts.
3:	When a CLRWDT instruction is executed, the postscaler count will be cleared.

25.2.1 CONTROL REGISTER

Register 25-14 shows the WDTCON register. This is a readable and writable register which contains a control bit that allows software to override the WDT enable Configuration bit, but only if the Configuration bit has disabled the WDT.

FIGURE 25-1: WDT BLOCK DIAGRAM



25.3 Two-Speed Start-up

The Two-Speed Start-up feature helps to minimize the latency period from oscillator start-up to code execution by allowing the microcontroller to use the INTRC oscillator as a clock source until the primary clock source is available. It is enabled by setting the IESO Configuration bit.

Two-Speed Start-up should be enabled only if the primary oscillator mode is LP, XT, HS or HSPLL (Crystal-Based modes). Other sources do not require an Oscillator Start-up Timer delay; for these, Two-Speed Start-up should be disabled.

When enabled, Resets and wake-ups from Sleep mode cause the device to configure itself to run from the internal oscillator block as the clock source, following the time-out of the Power-up Timer after a Power-on Reset is enabled. This allows almost immediate code execution while the primary oscillator starts and the OST is running. Once the OST times out, the device automatically switches to PRI_RUN mode.

Because the OSCCON register is cleared on Reset events, the INTOSC (or postscaler) clock source is not initially available after a Reset event; the INTRC clock is used directly at its base frequency. 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, IRCF<2:0>, 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.

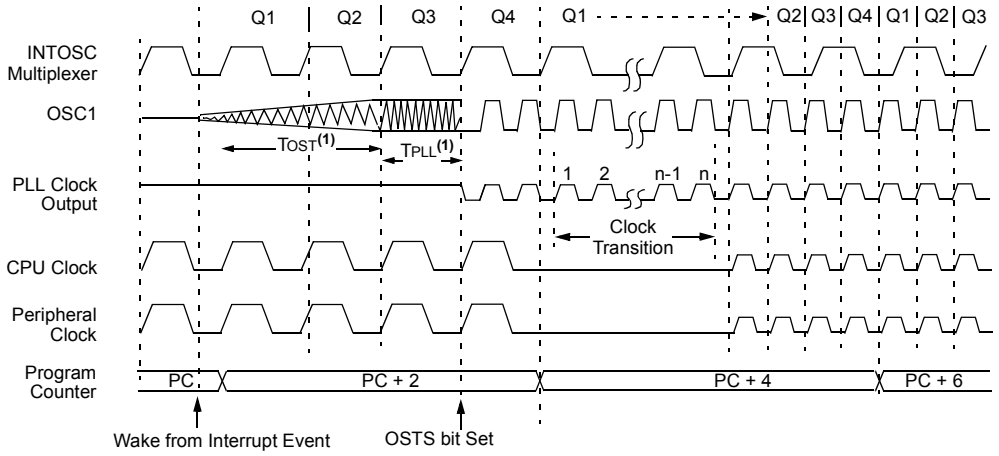
In all other power-managed modes, Two-Speed Start-up is not used. The device will be clocked by the currently selected clock source until the primary clock source becomes available. The setting of the IESO bit is ignored.

25.3.1 SPECIAL CONSIDERATIONS FOR USING TWO-SPEED START-UP

While using the INTRC oscillator in Two-Speed Start-up, the device still obeys the normal command sequences for entering power-managed modes, including serial `SLEEP` instructions (refer to **Section 4.1.4 "Multiple Sleep Commands"**). In practice, this means that user code can change the SCS<1:0> bit settings or issue `SLEEP` instructions before the OST times out. This would allow an application to briefly wake-up, perform routine "housekeeping" tasks and return to Sleep before the device starts to operate from the primary oscillator.

User code can also check if the primary clock source is currently providing the device clocking by checking the status of the OSTS bit (OSCCON<3>). If the bit is set, the primary oscillator is providing the clock. Otherwise, the internal oscillator block is providing the clock during wake-up from Reset or Sleep mode.

FIGURE 25-2: TIMING TRANSITION FOR TWO-SPEED START-UP (INTOSC TO HSPLL)



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

SLEEP — Enter Sleep mode

Syntax: SLEEP

Operands: None

Operation: 00h → WDT,
0 → WDT postscaler,
1 → $\overline{TO}$,
0 → $\overline{PD}$

Status Affected: $\overline{TO}$, $\overline{PD}$

Encoding:

0000	0000	0000	0011

Description: The Power-Down Status bit ($\overline{PD}$) is cleared. The Time-out Status bit ($\overline{TO}$) is set. Watchdog Timer and its postscaler are cleared.

The processor is put into Sleep mode with the oscillator stopped.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	Process Data	Go to Sleep

Example: SLEEP

Before Instruction
$\overline{TO}$ = ?
$\overline{PD}$ = ?

After Instruction
$\overline{TO}$ = 1 †
$\overline{PD}$ = 0

† If WDT causes wake-up, this bit is cleared.

SUBFWB — Subtract f from W with Borrow

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

Operands: 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

Operation: (W) – (f) – ($\overline{C}$) → dest

Status Affected: N, OV, C, DC, Z

Encoding:

0101	01da	ffff	ffff

Description: Subtract register 'f' and Carry flag (borrow) from W (2's complement method). If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored in register 'f'.

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

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 26.2.3 "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 1: SUBFWB REG, 1, 0

Before Instruction
REG = 3
W = 2
C = 1

After Instruction
REG = FF
W = 2
C = 0
Z = 0
N = 1 ; result is negative

Example 2: SUBFWB REG, 0, 0

Before Instruction
REG = 2
W = 5
C = 1

After Instruction
REG = 2
W = 3
C = 1
Z = 0
N = 0 ; result is positive

Example 3: SUBFWB REG, 1, 0

Before Instruction
REG = 1
W = 2
C = 0

After Instruction
REG = 0
W = 2
C = 1
Z = 1 ; result is zero
N = 0

26.2 Extended Instruction Set

In addition to the standard 75 instructions of the PIC18 instruction set, PIC18F2480/2580/4480/4580 devices also provide an optional extension to the core CPU functionality. The added features include eight additional instructions that augment indirect and indexed addressing operations and the implementation of Indexed Literal Offset Addressing mode for many of the standard PIC18 instructions.

The additional features are disabled by default. To enable them, users must set the XINST Configuration bit.

The instructions in the extended set can all be classified as literal operations, which either manipulate the File Select Registers or use them for indexed addressing. Two of the instructions, `ADDFSR` and `SUBFSR`, each have an additional special instantiation for using FSR2. These versions (`ADDULNK` and `SUBULNK`) allow for automatic return after execution.

The extended instructions are specifically implemented to optimize re-entrant program code (that is, code that is recursive or that uses a software stack) written in high-level languages, particularly C. Among other things, they allow users working in high-level languages to perform certain operations on data structures more efficiently. These include:

- dynamic allocation and de-allocation of software stack space when entering and leaving subroutines
- function pointer invocation
- software Stack Pointer manipulation
- manipulation of variables located in a software stack

A summary of the instructions in the extended instruction set is provided in Table 26-3. Detailed descriptions are provided in **Section 26.2.2 "Extended Instruction Set"**. The opcode field descriptions in Table 26-1 apply to both the standard and extended PIC18 instruction sets.

Note:	The instruction set extension and the Indexed Literal Offset Addressing mode were designed for optimizing applications written in C; the user may likely never use these instructions directly in assembler. The syntax for these commands is provided as a reference for users who may be reviewing code that has been generated by a compiler.

26.2.1 EXTENDED INSTRUCTION SYNTAX

Most of the extended instructions use indexed arguments, using one of the File Select Registers and some offset to specify a source or destination register. When an argument for an instruction serves as part of indexed addressing, it is enclosed in square brackets ("[ ]"). This is done to indicate that the argument is used as an index or offset. MPASM™ Assembler will flag an error if it determines that an index or offset value is not bracketed.

When the extended instruction set is enabled, brackets are also used to indicate index arguments in byte-oriented and bit-oriented instructions. This is in addition to other changes in their syntax. For more details, see **Section 26.2.3.1 "Extended Instruction Syntax with Standard PIC18 Commands"**.

Note:	In the past, square brackets have been used to denote optional arguments in the PIC18 and earlier instruction sets. In this text and going forward, optional arguments are denoted by braces ("{ }").

TABLE 26-3: EXTENSIONS TO THE PIC18 INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word				Status Affected
				MSb			LSb	
ADDFSR	f, k	Add Literal to FSR	1	1110	1000	ffkk	kkkk	None
ADDULNK	k	Add Literal to FSR2 and Return	2	1110	1000	11kk	kkkk	None
CALLW		Call Subroutine using WREG	2	0000	0000	0001	0100	None
MOVSF	z_s, f_d	Move z_s (source) to 1st word	2	1110	1011	0zzz	zzzz	None
		f_d (destination) 2nd word		1111	ffff	ffff	ffff	
MOVSS	z_s, z_d	Move z_s (source) to 1st word	2	1110	1011	1zzz	zzzz	None
		z_d (destination) 2nd word		1111	xxxx	xzzz	zzzz	
PUSHL	k	Store Literal at FSR2, Decrement FSR2	1	1110	1010	kkkk	kkkk	None
SUBFSR	f, k	Subtract Literal from FSR	1	1110	1001	ffkk	kkkk	None
SUBULNK	k	Subtract Literal from FSR2 and Return	2	1110	1001	11kk	kkkk	None

SUBFSR **Subtract Literal from FSR**

Syntax: SUBFSR f, k

Operands: $0 \le k \le 63$
$f \in [0, 1, 2]$

Operation: FSRf – k → FSRf

Status Affected: None

Encoding:

1110	1001	ffkk	kkkk

Description: The 6-bit literal 'k' is subtracted from the contents of the FSR specified by 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: `SUBFSR 2, 23h`

Before Instruction
FSR2 = 03FFh

After Instruction
FSR2 = 03DCh

SUBULNK **Subtract Literal from FSR2 and Return**

Syntax: SUBULNK k

Operands: $0 \le k \le 63$

Operation: FSR2 – k → FSR2
(TOS) → PC

Status Affected: None

Encoding:

1110	1001	11kk	kkkk

Description: The 6-bit literal 'k' is subtracted from the contents of the FSR2. A `RETURN` is then executed by loading the PC with the TOS.

The instruction takes two cycles to execute; a `NOP` is performed during the second cycle.

This may be thought of as a special case of the `SUBFSR` instruction, where f = 3 (binary '`11`'); it operates only on FSR2.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination
No Operation	No Operation	No Operation	No Operation

Example: `SUBULNK 23h`

Before Instruction
FSR2 = 03FFh
PC = 0100h

After Instruction
FSR2 = 03DCh
PC = (TOS)

28.3 DC Characteristics: PIC18F2480/2580/4480/4580 (Industrial) PIC18LF2480/2580/4480/4580 (Industrial)

DC CHARACTERISTICS			Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended			
Param No.	**Symbol**	**Characteristic**	**Min**	**Max**	**Units**	**Conditions**
	VIL	**Input Low Voltage**				
		I/O Ports:				
D030		with TTL Buffer	VSS	0.15 VDD	V	VDD < 4.5V
D030A			—	0.8	V	4.5V ≤ VDD ≤ 5.5V
D031		with Schmitt Trigger Buffer	VSS	0.2 VDD	V	
D031A		RC3 and RC4	VSS	0.3 Vdd	V	I²C™ enabled
D031B			VSS	0.8	V	SMBus enabled
D032		$\overline{MCLR}$	VSS	0.2 VDD	V	
D033		OSC1	VSS	0.3 VDD	V	HS, HSPLL modes
D033A		OSC1	VSS	0.2 VDD	V	RC, EC modes(1)
D033B		OSC1	VSS	0.3	V	XT, LP modes
D034		T13CKI	VSS	0.3	V	
	VIH	**Input High Voltage**				
		I/O Ports:				
D040		with TTL Buffer	0.25 VDD + 0.8V	VDD	V	VDD < 4.5V
D040A			2.0	VDD	V	4.5V ≤ VDD ≤ 5.5V
D041		with Schmitt Trigger Buffer	0.8 VDD	VDD	V	
D041A		RC3 and RC4	0.7 VDD	VDD	V	I²C™ enabled
D041B			2.1	VDD	V	SMBus enabled, VDD ≥ 3V
D042		$\overline{MCLR}$	0.8 VDD	VDD	V	
D043		OSC1	0.7 VDD	VDD	V	HS, HSPLL modes
D043A		OSC1	0.8 VDD	VDD	V	EC mode
D043B		OSC1	0.9 VDD	VDD	V	RC mode(1)
D043C		OSC1	1.6	VDD	V	XT, LP modes
D044		T13CKI	1.6	VDD	V	
	IIL	**Input Leakage Current(2,3)**				
D060		I/O Ports	—	±200	nA	VDD < 5.5V, VSS ≤ VPIN ≤ VDD, Pin at high-impedance
			—	±50	nA	VDD < 3V, VSS ≤ VPIN ≤ VDD, Pin at high-impedance
D061		$\overline{MCLR}$	—	±1	µA	Vss ≤ VPIN ≤ VDD
D063		OSC1	—	±1	µA	Vss ≤ VPIN ≤ VDD
	IPU	**Weak Pull-up Current**				
D070	IPURB	PORTB Weak Pull-up Current	50	400	µA	VDD = 5V, VPIN = VSS

Note 1: In RC oscillator configuration, the OSC1/CLKI pin is a Schmitt Trigger input. It is not recommended that the PIC® device be driven with an external clock while in RC mode.

2: The leakage current on the $\overline{MCLR}$ pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.

3: Negative current is defined as current sourced by the pin.

FIGURE 28-11: PARALLEL SLAVE PORT TIMING (PIC18F4480/4580)

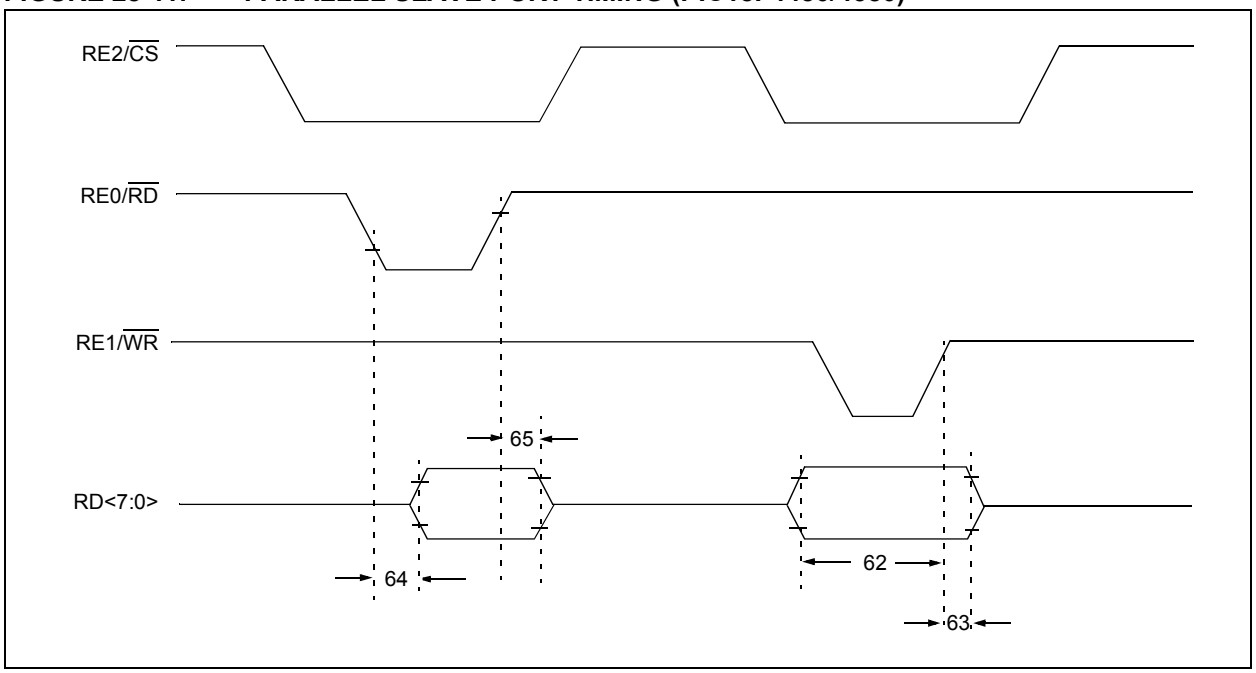


TABLE 28-13: PARALLEL SLAVE PORT REQUIREMENTS (PIC18F4480/4580)

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
62	TDTV2WRH	Data In Valid before $\overline{WR}$ ↑ or $\overline{CS}$ ↑ (setup time)		20	—	ns	
63	TWRH2DTI	$\overline{WR}$ ↑ or $\overline{CS}$ ↑ to Data–In Invalid (hold time)	PIC18**F**XXXX	20	—	ns	
			PIC18**LF**XXXX	35	—	ns	VDD = 2.0V
64	TRDL2DTV	$\overline{RD}$ ↓ and $\overline{CS}$ ↓ to Data–Out Valid		—	80	ns	
65	TRDH2DTI	$\overline{RD}$ ↑ or $\overline{CS}$ ↓ to Data–Out Invalid		10	30	ns	
66	TIBFINH	Inhibit of the IBF Flag bit being Cleared from $\overline{WR}$ ↑ or $\overline{CS}$ ↑		—	3 TCY		