Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

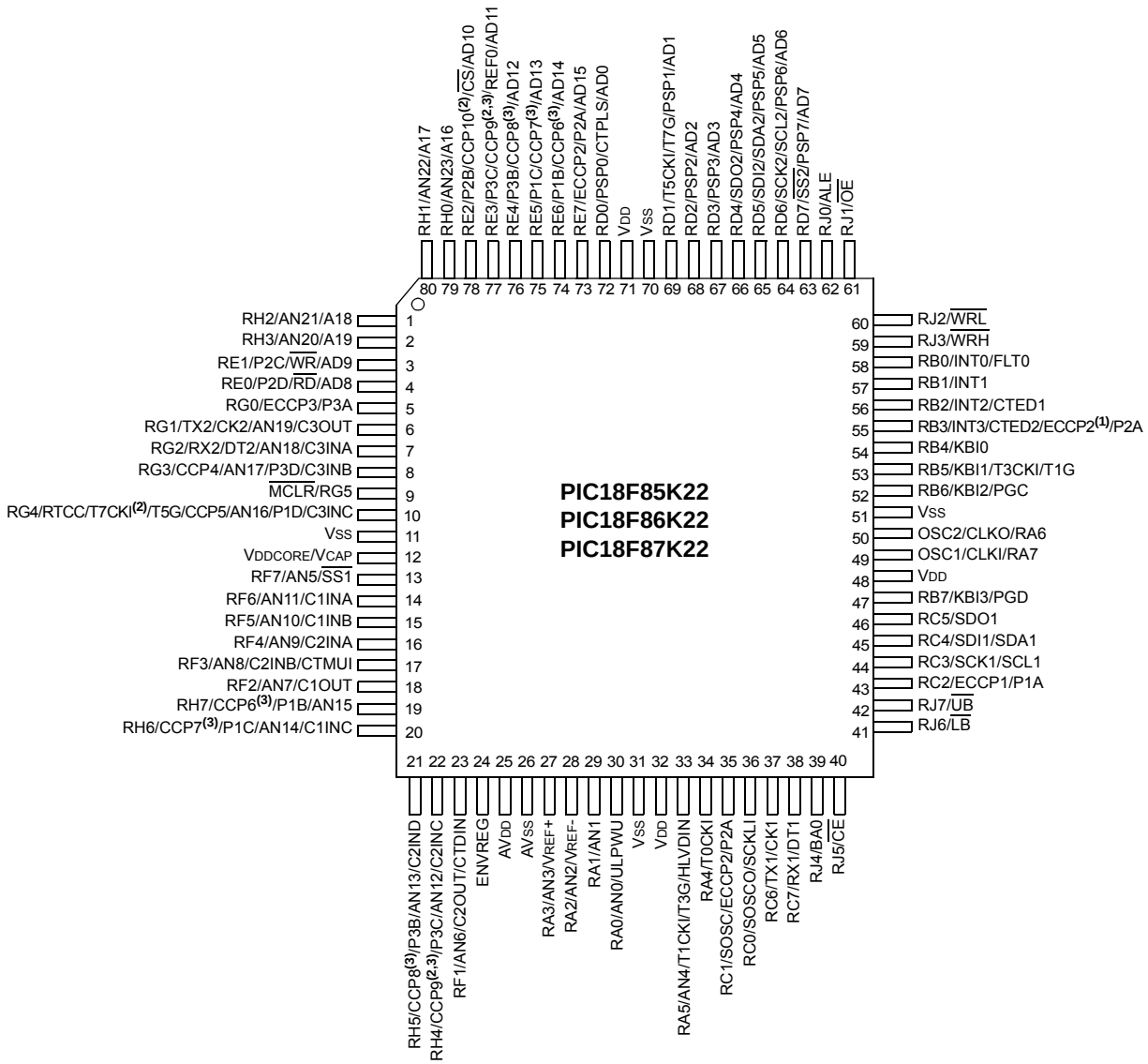
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" 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	64MHz
Connectivity	EBI/EMI, I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	69
Program Memory Size	128KB (64K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	4K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.5V
Data Converters	A/D 24x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	80-TQFP
Supplier Device Package	80-TQFP (12x12)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f87k22t-i-pt

Pin Diagrams – PIC18F8XK22

80-Pin TQFP



Note 1: The ECCP2 pin placement depends on the CCP2MX Configuration bit setting and whether the device is in Microcontroller or Extended Microcontroller mode.

2: Not available on the PIC18F65K22 and PIC18F85K22 devices.

3: The CC6, CCP7, CCP8 and CCP9 pin placement depends on the setting of the ECCPMX Configuration bit (CONFIG3H<1>).

1.3 Details on Individual Family Members

Devices in the PIC18F87K22 family are available in 64-pin and 80-pin packages. Block diagrams for the two groups are shown in Figure 1-1 and Figure 1-2.

The devices are differentiated from each other in these ways:

- Flash Program Memory:
  - PIC18FX5K22 (PIC18F65K22 and PIC18F85K22) – 32 Kbytes
  - PIC18FX6K22 (PIC18F66K22 and PIC18F86K22) – 64 Kbytes
  - PIC18FX7K22 (PIC18F67K22 and PIC18F87K22) – 128 Kbytes
- Data RAM:
  - All devices except PIC18FX5K22 – 4 Kbytes
  - PIC18FX5K22 – 2 Kbytes
- I/O Ports:
  - PIC18F6XK22 (64-pin devices) – 7 bidirectional ports
  - PIC18F8XK22 (80-pin devices) – 9 bidirectional ports
- CCP modules:
  - PIC18FX5K22 (PIC18F65K22 and PIC18F85K22) – 5 CCP modules
  - PIC18FX6K22 and PIC18FX7K22 (PIC18F66K22, PIC18F86K22, PIC18F67K22, and PIC18F87K22) – 7 CCP modules
- Timer modules:
  - PIC18FX5K22 (PIC18F65K22 and PIC18F85K22) – Four 8-bit timer/counters and four 16-bit timer/counters
  - PIC18FX6K22 and PIC18FX7K22 (PIC18F66K22, PIC18F86K22, PIC18F67K22, and PIC18F87K22) – Six 8-bit timer/counters and five 16-bit timer/counters
- A/D Channels:
  - PIC18F6XK22 (64-pin devices) – 24 channels
  - PIC18F8XK22 (80-pin devices) – 16 channels

All other features for devices in this family are identical. These are summarized in Table 1-1 and Table 1-2.

The pinouts for all devices are listed in Table 1-3 and Table 1-4.

TABLE 1-3: PIC18F6XK22 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	QFN/TQFP			
				PORTC is a bidirectional I/O port.
RC0/SOSCO/SCLKI	30			
RC0		I/O	ST	Digital I/O.
SOSCO		O	—	SOSC oscillator output.
SCLKI		I	ST	Digital SOSC input.
RC1/SOSCI/ECCP2/P2A	29			
RC1		I/O	ST	Digital I/O.
SOSCI		I	CMOS	SOSC oscillator input.
ECCP2(1)		I/O	ST	Capture 2 input/Compare 2 output/PWM2 output.
P2A		O	—	Enhanced PWM2 Output A.
RC2/ECCP1/P1A	33			
RC2		I/O	ST	Digital I/O.
ECCP1		I/O	ST	Capture 1 input/Compare 1 output/PWM1 output.
P1A		O	—	Enhanced PWM1 Output A.
RC3/SCK1/SCL1	34			
RC3		I/O	ST	Digital I/O.
SCK1		I/O	ST	Synchronous serial clock input/output for SPI mode.
SCL1(4)		I/O	I^2C	Synchronous serial clock input/output for I^2C™ mode.
RC4/SDI1/SDA1	35			
RC4		I/O	ST	Digital I/O.
SDI1		I	ST	SPI data in.
SDA1(4)		I/O	I^2C	I^2C data I/O.
RC5/SDO1	36			
RC5		I/O	ST	Digital I/O.
SDO1		O	—	SPI data out.
RC6/TX1/CK1	31			
RC6		I/O	ST	Digital I/O.
TX1		O	—	EUSART asynchronous transmit.
CK1		I/O	ST	EUSART synchronous clock (see related RX1/DT1).
RC7/RX1/DT1	32			
RC7		I/O	ST	Digital I/O.
RX1		I	ST	EUSART asynchronous receive.
DT1		I/O	ST	EUSART synchronous data (see related TX1/CK1).

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; Analog = Analog input; I = Input; O = Output; P = Power; OD = Open-Drain (no P diode to V_{DD}); I^2C = I^2C™/SMBus

Note 1: Default assignment for ECCP2 when the CCP2MX Configuration bit is set.

2: Alternate assignment for ECCP2 when the CCP2MX Configuration bit is cleared.

3: Not available on PIC18F65K22 and PIC18F85K22 devices.

4: The CC6, CCP7, CCP8 and CCP9 pin placement depends on the setting of the ECCPMX Configuration bit (CONFIG3H<1>).

3.0 OSCILLATOR CONFIGURATIONS

3.1 Oscillator Types

The PIC18F87K22 family of devices can be operated in the following oscillator modes:

- EC External clock, RA6 available
- ECIO External clock, clock out RA6 (FOSC/4 on RA6)
- HS High-Speed Crystal/Resonator
- XT Crystal/Resonator
- LP Low-Power Crystal
- RC External Resistor/Capacitor, RA6 available
- RCIO External Resistor/Capacitor, clock out RA6 (FOSC/4 on RA6)
- INTIO2 Internal Oscillator with I/O on RA6 and RA7
- INTIO1 Internal Oscillator with FOSC/4 output on RA6 and I/O on RA7

There is also an option for running the 4xPLL on any of the clock sources in the input frequency range of 4 to 16 MHz.

The PLL is enabled by setting the PLLCFG bit (CONFIG1H<4>) or the PLLEN bit (OSCTUNE<6>).

For the EC and HS mode, the PLLEN (software) or PLLCFG (CONFIG) bit can be used to enable the PLL.

For the INTIOx modes (HF-INTOSC):

- Only the PLLEN can enable the PLL (PLLCFG is ignored).
- When the oscillator is configured for the internal oscillator (FOSC<3:0> = 100x), the PLL can be enabled only when the HF-INTOSC frequency is 8 or 16 MHz.

When the RA6 and RA7 pins are not used for an oscillator function or CLKOUT function, they are available as general purpose I/Os.

To optimize power consumption when using EC/HS/XT/LP/RC as the primary oscillator, the frequency input range can be configured to yield an optimized power bias:

- Low-Power Bias – External frequency less than 160 kHz
- Medium Power Bias – External frequency between 160 kHz and 16 MHz
- High-Power Bias – External frequency greater than 16 MHz

All of these modes are selected by the user by programming the FOSC<3:0> Configuration bits (CONFIG1H<3:0>). In addition, PIC18F87K22 family devices can switch between different clock sources, either under software control or, under certain conditions, automatically. This allows for additional power savings by managing device clock speed in real time without resetting the application. The clock sources for the PIC18F87K22 family of devices are shown in Figure 3-1.

For the HS and EC mode, there are additional power modes of operation – depending on the frequency of operation.

HS1 is the Medium Power mode with a frequency range of 4 MHz to 16 MHz. HS2 is the High-Power mode, where the oscillator frequency can go from 16 MHz to 25 MHz. HS1 and HS2 are achieved by setting the CONFIG1H<3:0> correctly. (For details, see Register 28-2 on page 406.)

EC mode has these modes of operation:

- EC1 – For low power with a frequency range up to 160 kHz
- EC2 – Medium power with a frequency range of 160 kHz to 16 MHz
- EC3 – High power with a frequency range of 16 MHz to 64 MHz

EC1, EC2 and EC3 are achieved by setting the CONFIG1H<3:0> correctly. (For details, see Register 28-2 on page 406.)

Table 3-1 shows the HS and EC modes' frequency range and FOSC<3:0> settings.

4.0 POWER-MANAGED MODES

The PIC18F87K22 family of devices offers a total of seven operating modes for more efficient power management. These modes provide a variety of options for selective power conservation in applications where resources may be limited (such as battery-powered devices).

There are three categories of power-managed mode:

- Run modes
- Idle modes
- Sleep mode

There is an Ultra Low-Power Wake-up (ULPWU) for waking from the Sleep mode.

These categories define which portions of the device are clocked, and sometimes, at what speed. The Run and Idle modes may use any of the three available clock sources (primary, secondary or internal oscillator block). The Sleep mode does not use a clock source.

The ULPWU mode, on the RA0 pin, enables a slow falling voltage to generate a wake-up, even from Sleep, without excess current consumption. (See **Section 4.7 "Ultra Low-Power Wake-up"**.)

The power-managed modes include several power-saving features offered on previous PIC® devices. One is the clock switching feature, offered in other PIC18 devices. This feature allows the controller to use the SOSC oscillator instead of the primary one. Another power-saving feature is Sleep mode, offered by all PIC devices, where all device clocks are stopped.

4.1 Selecting Power-Managed Modes

Selecting a power-managed mode requires two decisions:

- Will the CPU be clocked or not
- What will be the clock source

The IDLEN bit (OSCCON<7>) controls CPU clocking, while the SCS<1:0> bits (OSCCON<1:0>) select the clock source. The individual modes, bit settings, clock sources and affected modules are summarized in Table 4-1.

4.1.1 CLOCK SOURCES

The SCS<1:0> bits select one of three clock sources for power-managed modes. Those sources are:

- The primary clock as defined by the FOSC<3:0> Configuration bits
- The secondary clock (the SOSC oscillator)
- The internal oscillator block (for LF-INTOSC modes)

4.1.2 ENTERING POWER-MANAGED MODES

Switching from one power-managed mode to another begins by loading the OSCCON register. The SCS<1:0> bits select the clock source and determine which Run or Idle mode is used. Changing these bits causes an immediate switch to the new clock source, assuming that it is running. The switch may also be subject to clock transition delays. These considerations are discussed in **Section 4.1.3 "Clock Transitions and Status Indicators"** and subsequent sections.

Entering the power-managed Idle or Sleep modes is triggered by the execution of a SLEEP instruction. The actual mode that results depends on the status of the IDLEN bit.

Depending on the current and impending mode, a change to a power-managed mode does not always require setting all of the previously discussed bits. Many transitions can be done by changing the oscillator select bits, or changing the IDLEN bit, prior to issuing a SLEEP instruction. If the IDLEN bit is already configured as desired, it may only be necessary to perform a SLEEP instruction to switch to the desired mode.

TABLE 4-1: POWER-MANAGED MODES

Mode	OSCCON Bits		Module Clocking		Available Clock and Oscillator Source
	IDLEN<7>(1)	SCS<1:0>	CPU	Peripherals	
Sleep	0	N/A	Off	Off	None – All clocks are disabled
PRI_RUN	N/A	00	Clocked	Clocked	Primary – XT, LP, HS, EC, RC and PLL modes. This is the normal, Full-Power Execution mode.
SEC_RUN	N/A	01	Clocked	Clocked	Secondary – SOSC Oscillator
RC_RUN	N/A	1x	Clocked	Clocked	Internal oscillator block(2)
PRI_IDLE	1	00	Off	Clocked	Primary – LP, XT, HS, RC, EC
SEC_IDLE	1	01	Off	Clocked	Secondary – SOSC oscillator
RC_IDLE	1	1x	Off	Clocked	Internal oscillator block(2)

Note 1: IDLEN reflects its value when the SLEEP instruction is executed.

2: Includes INTOSC (HF-INTOSC and MG-INTOSC) and INTOSC postscaler, as well as the LF-INTOSC source.

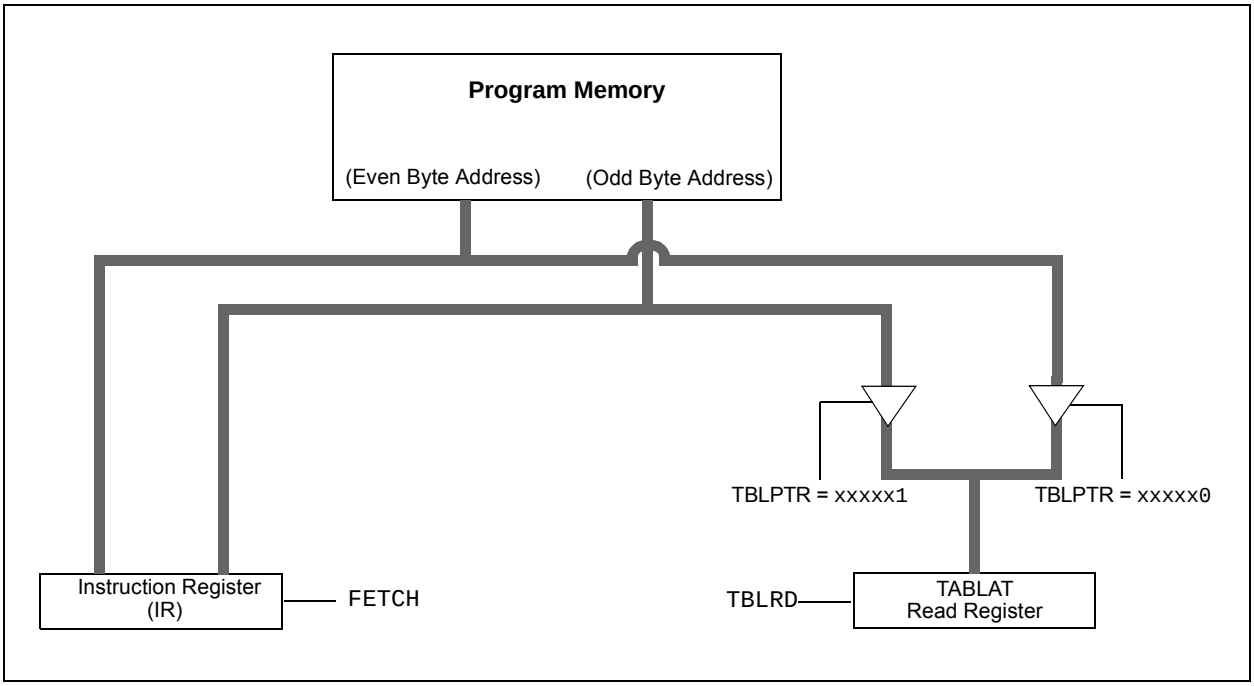
7.3 Reading the Flash Program Memory

The TBLRD instruction is used to retrieve data from program memory and places it into data RAM. Table reads from program memory are performed, one byte at a time.

TBLPTR points to a byte address in program space. Executing TBLRD places the byte pointed to into TABLAT. In addition, the TBLPTR can be modified automatically for the next table read operation.

The internal program memory is typically organized by words. The Least Significant bit of the address selects between the high and low bytes of the word. Figure 7-4 shows the interface between the internal program memory and the TABLAT.

FIGURE 7-4: READS FROM FLASH PROGRAM MEMORY



EXAMPLE 7-1: READING A FLASH PROGRAM MEMORY WORD

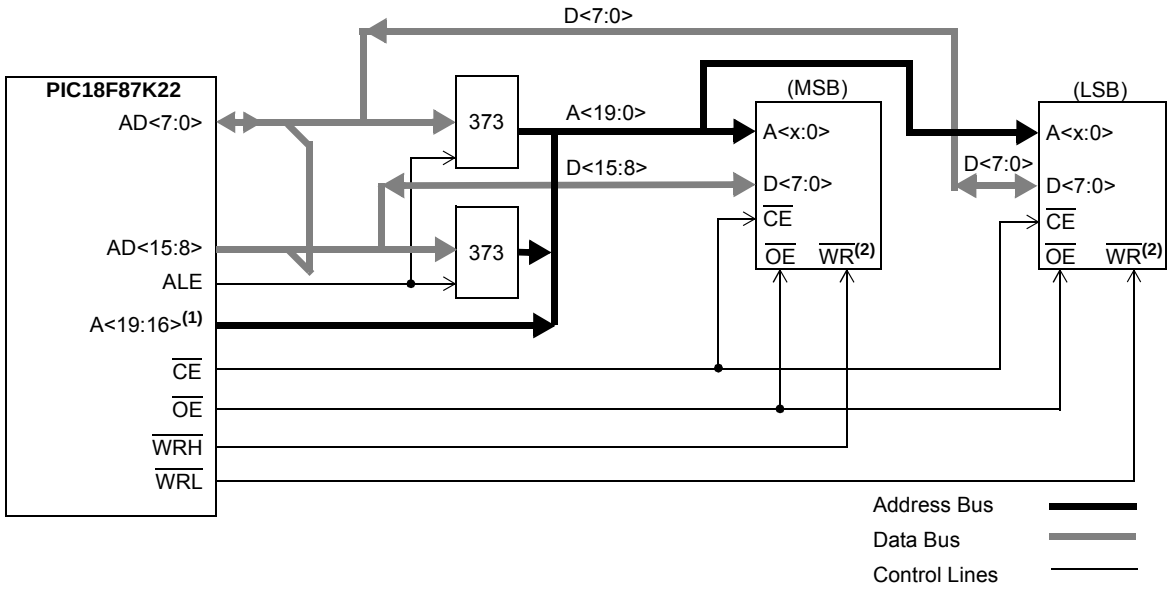
```
            BCF     EECON1, CFGS          ; point to Flash program memory
            BSF     EECON1, EEPGD         ; access Flash program memory
            MOVLW   CODE_ADDR_UPPER       ; Load TBLPTR with the base
            MOVWF   TBLPTRU               ; address of the word
            MOVLW   CODE_ADDR_HIGH
            MOVWF   TBLPTRH
            MOVLW   CODE_ADDR_LOW
            MOVWF   TBLPTRL
READ_WORD
            TBLRD*+                       ; read into TABLAT and increment
            MOVF    TABLAT, W             ; get data
            MOVWF   WORD_EVEN
            TBLRD*+                       ; read into TABLAT and increment
            MOVF    TABLAT, W             ; get data
            MOVF    WORD_ODD
```

8.6.1 16-BIT BYTE WRITE MODE

Figure 8-1 shows an example of 16-Bit Byte Write mode for PIC18F87K22 family devices. This mode is used for two separate 8-bit memories connected for 16-bit operation. This generally includes basic EPROM and Flash devices. It allows table writes to byte-wide external memories.

During a TBLWT instruction cycle, the TABLAT data is presented on the upper and lower bytes of the AD<15:0> bus. The appropriate $\overline{WRH}$ or $\overline{WRL}$ control line is strobed on the LSb of the TBLPTR.

FIGURE 8-1: 16-BIT BYTE WRITE MODE EXAMPLE



Note 1: Upper order address lines are used only for 20-bit address widths.

2: This signal only applies to table writes. See **Section 7.1 "Table Reads and Table Writes"**.

TABLE 12-5: PORTC FUNCTIONS (CONTINUED)

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RC3/SCK1/ SCL1	RC3	0	O	DIG	LATC<3> data output.
		1	I	ST	PORTC<3> data input.
	SCK1	0	O	DIG	SPI clock output (MSSP module); takes priority over port data.
		1	I	ST	SPI clock input (MSSP module).
	SCL1	0	O	DIG	I^2C clock output (MSSP module); takes priority over port data.
		1	I	I^2C	I^2C clock input (MSSP module); input type depends on module setting.
RC4/SDI1/ SDA1	RC4	0	O	DIG	LATC<4> data output.
		1	I	ST	PORTC<4> data input.
	SDI1		I	ST	SPI data input (MSSP module).
	SDA1	1	O	DIG	I^2C data output (MSSP module); takes priority over port data.
		1	I	I^2C	I^2C data input (MSSP module); input type depends on module setting.
RC5/SDO1	RC5	0	O	DIG	LATC<5> data output.
		1	I	ST	PORTC<5> data input.
	SDO1	0	O	DIG	SPI data output (MSSP module).
RC6/TX1/CK1	RC6	0	O	DIG	LATC<6> data output.
		1	I	ST	PORTC<6> data input.
	TX1	1	O	DIG	Synchronous serial data output (EUSART module); takes priority over port data.
	CK1	1	O	DIG	Synchronous serial data input (EUSART module); user must configure as an input.
		1	I	ST	Synchronous serial clock input (EUSART module).
RC7/RX1/DT1	RC7	0	O	DIG	LATC<7> data output.
		1	I	ST	PORTC<7> data input.
	RX1	1	I	ST	Asynchronous serial receive data input (EUSART module).
	DT1	1	O	DIG	Synchronous serial data output (EUSART module); takes priority over port data.
		1	I	ST	Synchronous serial data input (EUSART module); user must configure as an input.

Legend: O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Trigger Buffer Input, TTL = TTL Buffer Input, I^2C = I^2C™/SMBus Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

Note 1: Default assignment for ECCP2 when the CCP2MX Configuration bit is set.

TABLE 12-6: SUMMARY OF REGISTERS ASSOCIATED WITH PORTC

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0
LATC	LATC7	LATBC6	LATC5	LATCB4	LATC3	LATC2	LATC1	LATC0
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0
ODCON1	SSP1OD	CCP2OD	CCP1OD	—	—	—	—	SSP2OD
ODCON3	U2OD	U1OD	—	—	—	—	—	CTMUDS

Legend: Shaded cells are not used by PORTC.

12.11 Parallel Slave Port

PORTD can function as an 8-bit-wide Parallel Slave Port (PSP), or microprocessor port, when control bit, PSPMODE (PSPCON<4>), is set. The port is asynchronously readable and writable by the external world through the $\overline{RD}$ control input pin (RE0/P2D/$\overline{RD}$/AD8) and $\overline{WR}$ control input pin (RE1/P2C/$\overline{WR}$/AD9).

Note:	The Parallel Slave Port is available only in Microcontroller mode.

The PSP can directly interface to an 8-bit microprocessor data bus. The external microprocessor can read or write the PORTD latch as an eight-bit latch.

Setting bit, PSPMODE, enables port pin, RE0/P2D/$\overline{RD}$/AD8, to be the $\overline{RD}$ input, RE1/P2C/$\overline{WR}$/AD9 to be the $\overline{WR}$ input and RE2/P2B/CCP10/$\overline{CS}$/AD10 to be the $\overline{CS}$ (Chip Select) input. For this functionality, the corresponding data direction bits of the TRISE register (TRISE<2:0>) must be configured as inputs (= 111).

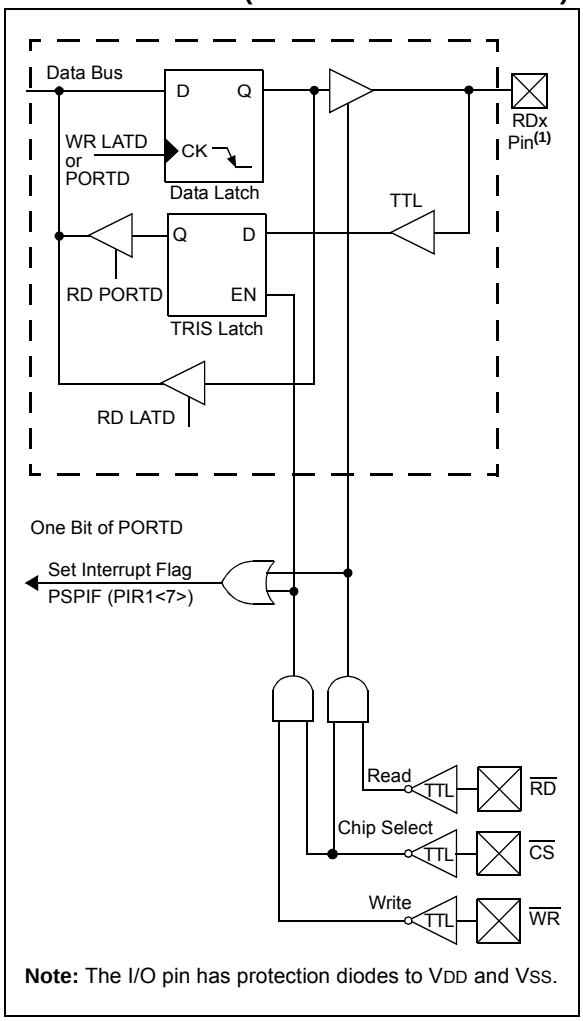
A write to the PSP occurs when both the $\overline{CS}$ and $\overline{WR}$ lines are first detected low and ends when either are detected high. The PSPIF and IBF flag bits (PIR1<7> and PSPCON<7>, respectively) are set when the write ends.

A read from the PSP occurs when both the $\overline{CS}$ and $\overline{RD}$ lines are first detected low. The data in PORTD is read out and the OBF bit (PSPCON<6>) is set. If the user writes new data to PORTD to set OBF, the data is immediately read out, but the OBF bit is not set.

When either the $\overline{CS}$ or $\overline{RD}$ line is detected high, the PORTD pins return to the input state and the PSPIF bit is set. User applications should wait for PSPIF to be set before servicing the PSP. When this happens, the IBF and OBF bits can be polled and the appropriate action taken.

The timing for the control signals in Write and Read modes is shown in Figure 12-4 and Figure 12-5, respectively.

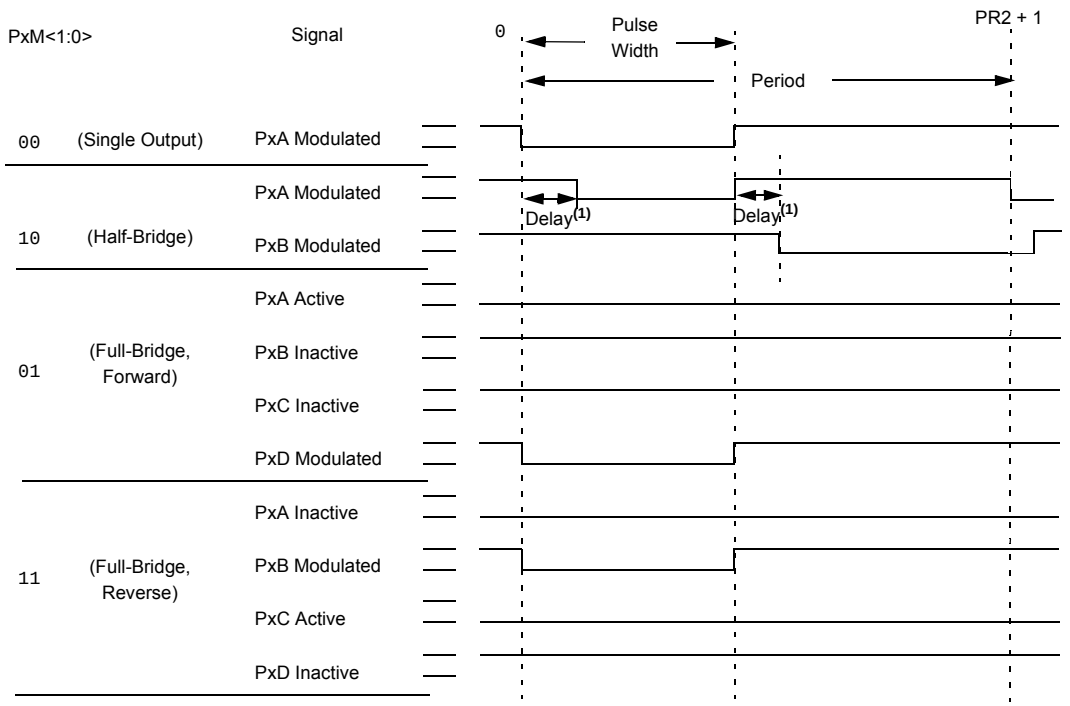
FIGURE 12-3: PORTD AND PORTE BLOCK DIAGRAM (PARALLEL SLAVE PORT)



Note: The I/O pin has protection diodes to VDD and VSS.

NOTES:

FIGURE 20-5: EXAMPLE ENHANCED PWM OUTPUT RELATIONSHIPS (ACTIVE-LOW STATE)

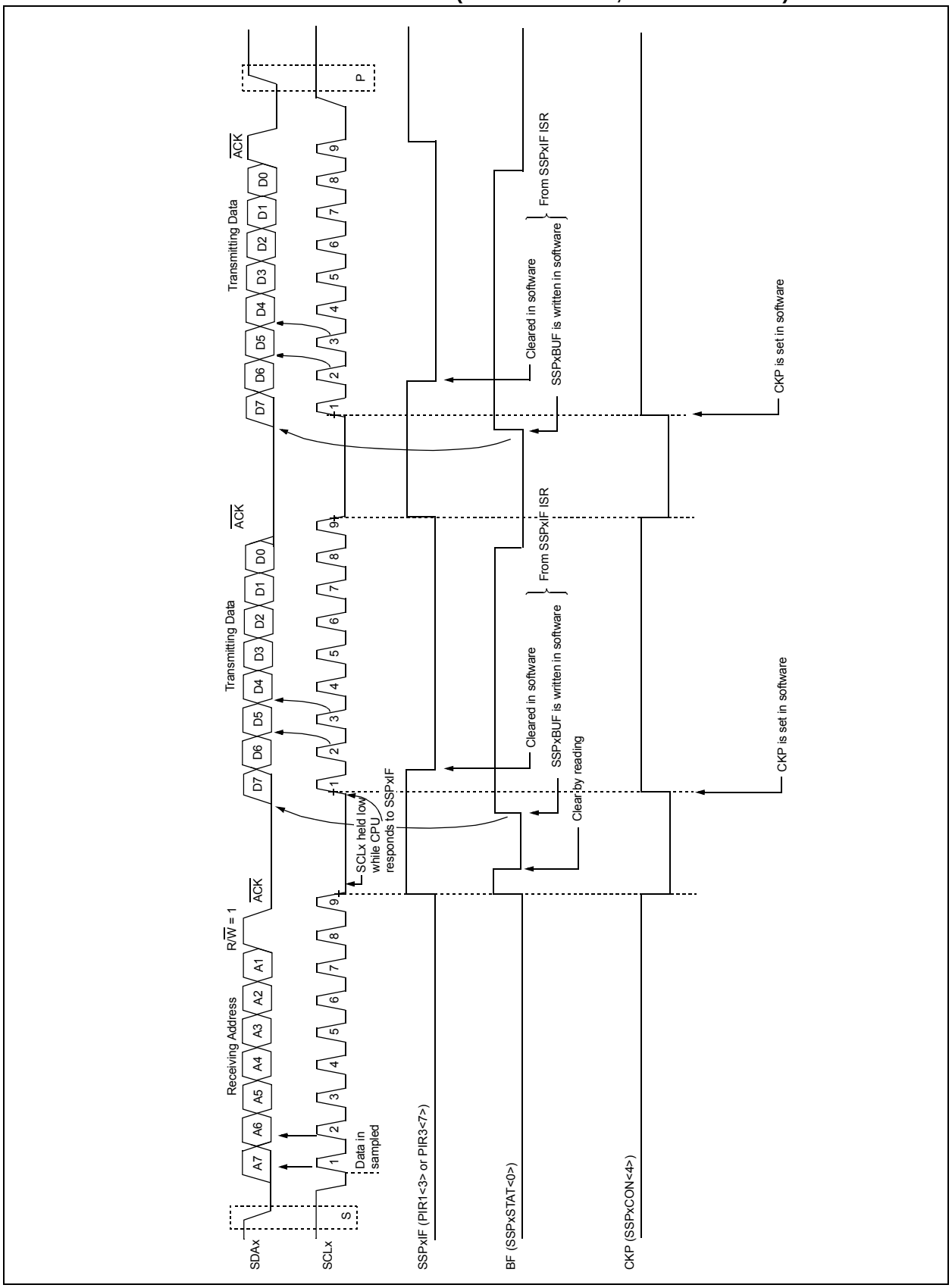


Relationships:

- Period = 4 * TOSC * (PR2 + 1) * (TMR2 Prescale Value)
- Pulse Width = TOSC * (CCPRxL<7:0>:CCPxCON<5:4>) * (TMR2 Prescale Value)
- Delay = 4 * TOSC * (ECCPxDEL<6:0>)

Note 1: Dead-band delay is programmed using the ECCP1DEL register (see **Section 20.4.6 "Programmable Dead-Band Delay Mode"**).

FIGURE 21-10: I²C™ SLAVE MODE TIMING (TRANSMISSION, 7-BIT ADDRESS)



21.4.9 I²C™ MASTER MODE REPEATED START CONDITION TIMING

A Repeated Start condition occurs when the RSEN bit (SSPxCON2<1>) is programmed high and the I²C logic module is in the Idle state. When the RSEN bit is set, the SCLx pin is asserted low. When the SCLx pin is sampled low, the Baud Rate Generator is loaded with the contents of SSPxADD<5:0> and begins counting. The SDAx pin is released (brought high) for one Baud Rate Generator count (TBRG). When the Baud Rate Generator times out, and if SDAx is sampled high, the SCLx pin will be deasserted (brought high). When SCLx is sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD<6:0> and begins counting. SDAx and SCLx must be sampled high for one TBRG. This action is then followed by assertion of the SDAx pin (SDAx = 0) for one TBRG while SCLx is high. Following this, the RSEN bit (SSPxCON2<1>) will be automatically cleared and the Baud Rate Generator will not be reloaded, leaving the SDAx pin held low. As soon as a Start condition is detected on the SDAx and SCLx pins, the S bit (SSPxSTAT<3>) will be set. The SSPxIF bit will not be set until the Baud Rate Generator has timed out.

Note 1: If RSEN is programmed while any other event is in progress, it will not take effect.

2: A bus collision during the Repeated Start condition occurs if:

- SDAx is sampled low when SCLx goes from low-to-high.
- SCLx goes low before SDAx is asserted low. This may indicate that another master is attempting to transmit a data '1'.

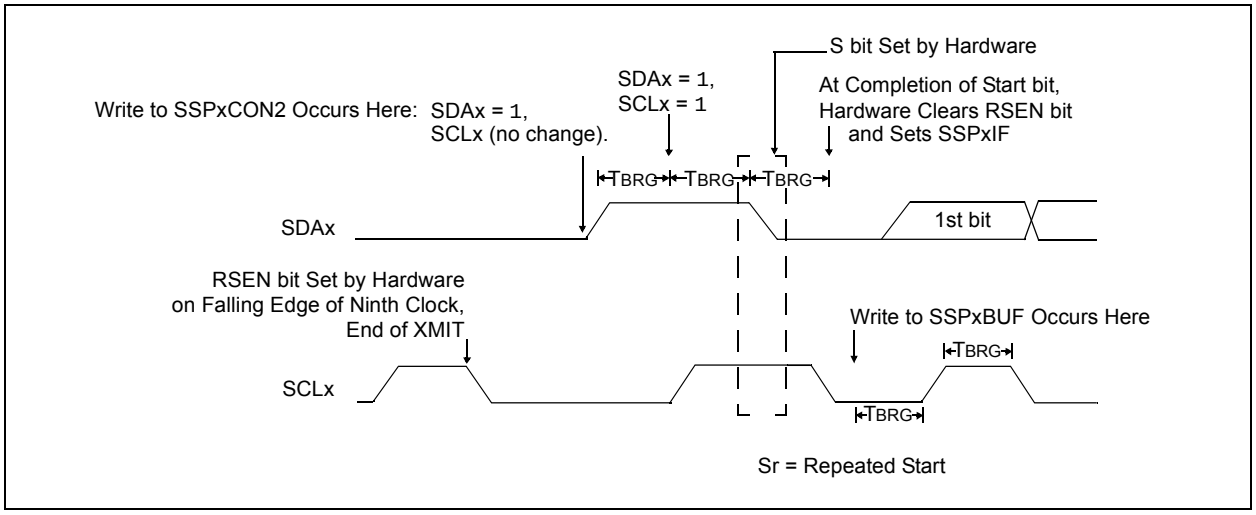
Immediately following the SSPxIF bit getting set, the user may write the SSPxBUF with the 7-bit address in 7-bit mode or the default first address in 10-bit mode. After the first eight bits are transmitted and an ACK is received, the user may then transmit an additional eight bits of address (10-bit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

21.4.9.1 WCOL Status Flag

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

Note: Because queueing of events is not allowed, writing of the lower 5 bits of SSPxCON2 is disabled until the Repeated Start condition is complete.

FIGURE 21-22: REPEATED START CONDITION WAVEFORM



REGISTER 22-3: BAUDCONx: BAUD RATE CONTROL REGISTER

R/W-0	R-1	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN
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 **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 **RXDTP:** Data/Receive Polarity Select bit
Asynchronous mode:
1 = Receive data (RXx) is inverted (active-low)
0 = Receive data (RXx) is not inverted (active-high)
Synchronous mode:
1 = Data (DTx) is inverted (active-low)
0 = Data (DTx) is not inverted (active-high)

bit 4 **TXCKP:** Synchronous Clock Polarity Select bit
Asynchronous mode:
1 = Idle state for transmit (TXx) is a low level
0 = Idle state for transmit (TXx) is a high level
Synchronous mode:
1 = Idle state for clock (CKx) is a high level
0 = Idle state for clock (CKx) is a low level

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

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

bit 1 **WUE:** Wake-up Enable bit
Asynchronous mode:
1 = EUSART will continue to sample the RXx pin – interrupt generated on falling edge; bit cleared in hardware on following rising edge
0 = RXx 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 is disabled or has completed
Synchronous mode:
Unused in this mode.

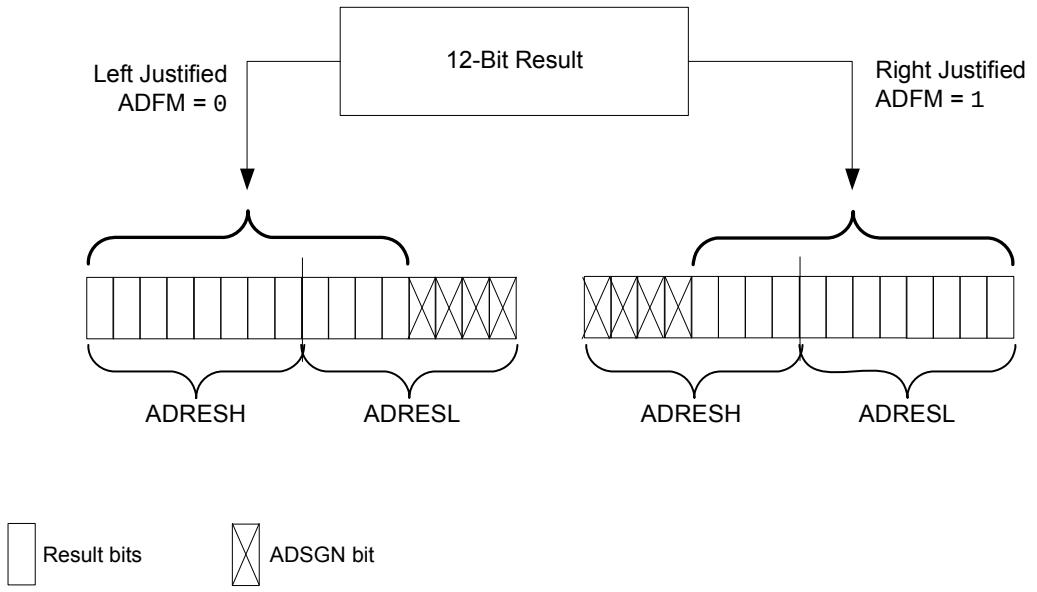
23.2.2 A/D RESULT REGISTERS

The ADRESH:ADRESL register pair is where the 12-bit A/D result and extended sign bits (ADSGN) are loaded at the completion of a conversion. This register pair is 16 bits wide. The A/D module gives the flexibility of left or right justifying the 12-bit result in the 16-Bit Result register. The A/D Format Select bit (ADFM) controls this justification.

Figure 23-3 shows the operation of the A/D result justification and location of the extended sign bits (ADSGN). The extended sign bits allow for easier 16-bit math to be performed on the result. The results are represented as a two's compliment binary value. This means that when sign bits and magnitude bits are considered together in right justification, the ADRESH and ADRESL registers can be read as a single signed integer value.

When the A/D Converter is disabled, these 8-bit registers can be used as two general purpose registers.

FIGURE 23-3: A/D RESULT JUSTIFICATION



Two's Complement Example Results Number Line

Left Justified		Right Justified	
Hex	**Decimal**	**Hex**	**Decimal**
0xFFF0	4095	0x0FFF	4095
0xFFE0	4094	0x0FFE	4094
...	...	...	...
0x0020	2	0x0002	2
0x0010	1	0x0001	1
0x0000	0	0x0000	0
0xFFFF	-1	0xFFFF	-1
0xFFEF	-2	0xFFFE	-2
...	...	...	...
0x001F	-4095	0xF001	-4095
0x000F	-4096	0xF000	-4096

BRA **Unconditional Branch**

Syntax: BRA n

Operands: $-1024 \le n \le 1023$

Operation: $(PC) + 2 + 2n \rightarrow PC$

Status Affected: None

Encoding:

1101	0nnn	nnnn	nnnn

Description: Add the 2's complement number '2n' to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is a two-cycle instruction.

Words: 1

Cycles: 2

Q Cycle Activity:

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

Example: HERE BRA Jump

Before Instruction
PC = address (HERE)
After Instruction
PC = address (Jump)

BSF **Bit Set f**

Syntax: BSF f, b {,a}

Operands: $0 \le f \le 255$
$0 \le b \le 7$
$a \in [0,1]$

Operation: $1 \rightarrow f<b>$

Status Affected: None

Encoding:

1000	bbba	ffff	ffff

Description: Bit 'b' in register 'f' is set.

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 $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: BSF FLAG_REG, 7, 1

Before Instruction
FLAG_REG = 0Ah
After Instruction
FLAG_REG = 8Ah

SUBLW **Subtract W from Literal**

Syntax: SUBLW k

Operands: $0 \le k \le 255$

Operation: $k - (W) \rightarrow W$

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

Encoding:

0000	1000	kkkk	kkkk

Description: W is subtracted from the eight-bit literal 'k'. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example 1: SUBLW 02h

Before Instruction
W = 01h
C = ?
After Instruction
W = 01h
C = 1 ; result is positive
Z = 0
N = 0

Example 2: SUBLW 02h

Before Instruction
W = 02h
C = ?
After Instruction
W = 00h
C = 1 ; result is zero
Z = 1
N = 0

Example 3: SUBLW 02h

Before Instruction
W = 03h
C = ?
After Instruction
W = FFh ; (2's complement)
C = 0 ; result is negative
Z = 0
N = 1

SUBWF **Subtract W from f**

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

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

Operation: $(f) - (W) \rightarrow$ dest

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

Encoding:

0101	11da	ffff	ffff

Description: Subtract W from register 'f' (2's complement method). If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back 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 $f \le 95$ (5Fh). See **Section 29.2.3 "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: SUBWF REG, 1, 0

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

Example 2: SUBWF REG, 0, 0

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

Example 3: SUBWF REG, 1, 0

Before Instruction
REG = 1
W = 2
C = ?
After Instruction
REG = FFh ;(2's complement)
W = 2
C = 0 ; result is negative
Z = 0
N = 1

30.2 MPLAB C Compilers for Various Device Families

The MPLAB C Compiler code development systems are complete ANSI C compilers for Microchip's PIC18, PIC24 and PIC32 families of microcontrollers and the dsPIC30 and dsPIC33 families of digital signal controllers. These compilers provide powerful integration capabilities, superior code optimization and ease of use.

For easy source level debugging, the compilers provide symbol information that is optimized to the MPLAB IDE debugger.

30.3 HI-TECH C for Various Device Families

The HI-TECH C Compiler code development systems are complete ANSI C compilers for Microchip's PIC family of microcontrollers and the dsPIC family of digital signal controllers. These compilers provide powerful integration capabilities, omniscient code generation and ease of use.

For easy source level debugging, the compilers provide symbol information that is optimized to the MPLAB IDE debugger.

The compilers include a macro assembler, linker, preprocessor, and one-step driver, and can run on multiple platforms.

30.4 MPASM Assembler

The MPASM Assembler is a full-featured, universal macro assembler for PIC10/12/16/18 MCUs.

The MPASM Assembler generates relocatable object files for the MPLINK Object Linker, Intel® standard HEX files, MAP files to detail memory usage and symbol reference, absolute LST files that contain source lines and generated machine code and COFF files for debugging.

The MPASM Assembler features include:

- Integration into MPLAB IDE projects
- User-defined macros to streamline assembly code
- Conditional assembly for multi-purpose source files
- Directives that allow complete control over the assembly process

30.5 MPLINK Object Linker/ MPLIB Object Librarian

The MPLINK Object Linker combines relocatable objects created by the MPASM Assembler and the MPLAB C18 C Compiler. It can link relocatable objects from precompiled libraries, using directives from a linker script.

The MPLIB Object Librarian manages the creation and modification of library files of precompiled code. When a routine from a library is called from a source file, only the modules that contain that routine will be linked in with the application. This allows large libraries to be used efficiently in many different applications.

The object linker/library features include:

- Efficient linking of single libraries instead of many smaller files
- Enhanced code maintainability by grouping related modules together
- Flexible creation of libraries with easy module listing, replacement, deletion and extraction

30.6 MPLAB Assembler, Linker and Librarian for Various Device Families

MPLAB Assembler produces relocatable machine code from symbolic assembly language for PIC24, PIC32 and dsPIC devices. MPLAB C Compiler uses the assembler to produce its object file. The assembler generates relocatable object files that can then be archived or linked with other relocatable object files and archives to create an executable file. Notable features of the assembler include:

- Support for the entire device instruction set
- Support for fixed-point and floating-point data
- Command line interface
- Rich directive set
- Flexible macro language
- MPLAB IDE compatibility

31.3 DC Characteristics: PIC18F87K22 Family (Industrial/Extended) (Continued)

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**
	VOL	**Output Low Voltage**				
D080		I/O Ports:				
		PORTA, PORTB, PORTC	—	0.6	V	IOL = 8.5 mA, VDD = 4.5V, -40°C to +125°C
		PORTD, PORTE, PORTF, PORTG, PORTH, PORTJ	—	0.6	V	IOL = 3.5 mA, VDD = 4.5V, -40°C to +125°C
D083		OSC2/CLKO (EC modes)	—	0.6	V	IOL = 1.6 mA. VDD = 5.5V, -40°C to +125°C
	VOH	**Output High Voltage(1)**				
D090		I/O Ports:			V	
		PORTA, PORTB, PORTC	VDD – 0.7	—	V	IOH = -3 mA, VDD = 4.5V, -40°C to +125°C
		PORTD, PORTE, PORTF, PORTG, PORTH, PORTJ	VDD – 0.7	—	V	IOH = -2 mA, VDD = 4.5V, -40°C to +125°C
D092		OSC2/CLKO (INTOSC, EC modes)	VDD – 0.7	—	V	IOH = -1 mA, VDD = 5.5V, -40°C to +125°C
		Capacitive Loading Specs on Output Pins				
D100	COSC2	OSC2 Pin	—	20	pF	In HS mode when external clock is used to drive OSC1
D101	CIO	All I/O Pins and OSC2	—	50	pF	To meet the AC Timing Specifications
D102	CB	SCLx, SDAx	—	400	pF	I^2C™ Specification

Note 1: Negative current is defined as current sourced by the pin.

31.4 DC Characteristics: CTMU Current Source Specifications

DC CHARACTERISTICS			Standard Operating Conditions: 1.8V to 5.5V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended				
Param No.	**Sym**	**Characteristic**	**Min**	**Typ(1)**	**Max**	**Units**	**Conditions**
	IOUT1	CTMU Current Source, Base Range	—	550	—	nA	CTMUICON<1:0> = 01
	IOUT2	CTMU Current Source, 10x Range	—	5.5	—	μA	CTMUICON<1:0> = 10
	IOUT3	CTMU Current Source, 100x Range	—	55	—	μA	CTMUICON<1:0> = 11

Note 1: Nominal value at center point of current trim range (CTMUICON<7:2> = 000000).