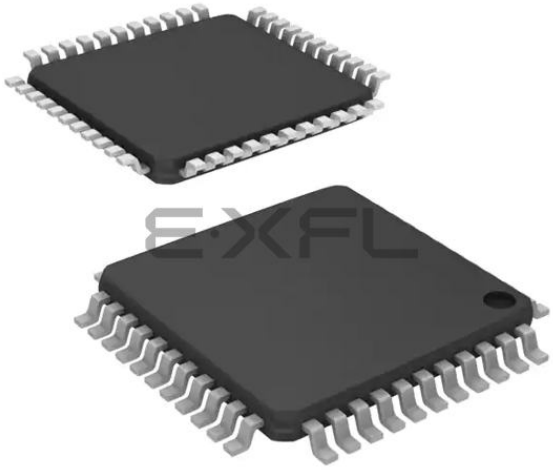


Welcome to E-XFL.COM

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	I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, HLVD, POR, PWM, WDT
Number of I/O	36
Program Memory Size	64KB (32K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	4K x 8
Voltage - Supply (Vcc/Vdd)	2.3V ~ 5.5V
Data Converters	A/D 35x12b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°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/pic18f46k42-e-pt

TABLE 4-9: SPECIAL FUNCTION REGISTER MAP FOR PIC18(L)F26/27/45/46/47/55/56/57K42 DEVICES BANK 58

3AFFh	—	3ADFh	SPI1SDIPPS	3ABFh	PPSLOCK	3A9Fh	—	3A7Fh	—	3A5Fh	—	3A3Fh	—	3A1Fh	RD7PPS(2)
3AFEh	—	3ADEh	SPI1SCKPPS	3ABEh	—(4)	3A9Eh	—	3A7Eh	—	3A5Eh	—	3A3Eh	—	3A1Eh	RD6PPS(2)
3AFDh	—	3ADDh	ADACTPPS	3ABDh	—	3A9Dh	—	3A7Dh	—	3A5Dh	—	3A3Dh	—	3A1Dh	RD5PPS(2)
3AFCh	—	3ADCh	CLCIN3PPS	3ABCh	—	3A9Ch	—	3A7Ch	—	3A5Ch	—	3A3Ch	—	3A1Ch	RD4PPS(2)
3AFBh	—	3ADBh	CLCIN2PPS	3ABBh	—	3A9Bh	—	3A7Bh	RD1I2C(2)	3A5Bh	RB2I2C	3A3Bh	—	3A1Bh	RD3PPS(2)
3AFAh	—	3ADAh	CLCIN1PPS	3ABAh	—	3A9Ah	—	3A7Ah	RD0I2C(2)	3A5Ah	RB1I2C	3A3Ah	—	3A1Ah	RD2PPS(2)
3AF9h	—	3AD9h	CLCIN0PPS	3AB9h	—	3A99h	—(4)	3A79h	—(4)	3A59h	—(4)	3A39h	—	3A19h	RD1PPS(2)
3AF8h	—	3AD8h	MD1SRCPPS	3AB8h	—	3A98h	—(4)	3A78h	—(4)	3A58h	—(4)	3A38h	—	3A18h	RD0PPS(2)
3AF7h	—	3AD7h	MD1CARHPPS	3AB7h	—	3A97h	—	3A77h	—	3A57h	IOCBF	3A37h	—	3A17h	RC7PPS
3AF6h	—	3AD6h	MD1CARLPPS	3AB6h	—	3A96h	—	3A76h	—	3A56h	IOCBN	3A36h	—	3A16h	RC6PPS
3AF5h	—	3AD5h	CWG3INPPS	3AB5h	—	3A95h	—	3A75h	—	3A55h	IOCBP	3A35h	—	3A15h	RC5PPS
3AF4h	—	3AD4h	CWG2INPPS	3AB4h	—	3A94h	INLVLF(3)	3A74h	INLVLD(2)	3A54h	INLVLB	3A34h	—	3A14h	RC4PPS
3AF3h	—	3AD3h	CWG1INPPS	3AB3h	—	3A93h	SLRCONF(3)	3A73h	SLRCOND(2)	3A53h	SLRCONB	3A33h	—	3A13h	RC3PPS
3AF2h	—	3AD2h	SMT1SIGPPS	3AB2h	—	3A92h	ODCONF(3)	3A72h	ODCOND(2)	3A52h	ODCONB	3A32h	—	3A12h	RC2PPS
3AF1h	—	3AD1h	SMT1WINPPS	3AB1h	—	3A91h	WPUF(3)	3A71h	WPUD(2)	3A51h	WPUB	3A31h	—	3A11h	RC1PPS
3AF0h	—	3AD0h	CCP4PPS	3AB0h	—	3A90h	ANSELF(3)	3A70h	ANSELD(2)	3A50h	ANSELB	3A30h	—	3A10h	RC0PPS
3AEFh	—	3ACFh	CCP3PPS	3AAFh	—	3A8Fh	—	3A6Fh	—	3A4Fh	—	3A2Fh	RF7PPS(3)	3A0Fh	RB7PPS
3AEEh	—	3ACEh	CCP2PPS	3AAEh	—	3A8Eh	—	3A6Eh	—	3A4Eh	—	3A2Eh	RF6PPS(3)	3A0Eh	RB6PPS
3AEDh	—	3ACDh	CCP1PPS	3AADh	—	3A8Dh	—	3A6Dh	—	3A4Dh	—	3A2Dh	RF5PPS(3)	3A0Dh	RB5PPS
3AECh	—	3ACCh	T6INPPS	3AACh	—	3A8Ch	—	3A6Ch	—	3A4Ch	—	3A2Ch	RF4PPS(3)	3A0Ch	RB4PPS
3AEBh	—	3ACBh	T4INPPS	3AABh	—	3A8Bh	—	3A6Bh	RC4I2C	3A4Bh	—	3A2Bh	RF3PPS(3)	3A0Bh	RB3PPS
3AEAh	—	3ACAh	T2INPPS	3AAAh	—	3A8Ah	—	3A6Ah	RC3I2C	3A4Ah	—	3A2Ah	RF2PPS(3)	3A0Ah	RB2PPS
3AE9h	U2CTSPPS	3AC9h	T5GPPS	3AA9h	—	3A89h	—(4)	3A69h	—(4)	3A49h	—(4)	3A29h	RF1PPS(3)	3A09h	RB1PPS
3AE8h	U2RXPPS	3AC8h	T5CKIPPS	3AA8h	—	3A88h	—(4)	3A68h	—(4)	3A48h	—(4)	3A28h	RF0PPS(3)	3A08h	RB0PPS
3AE7h	—	3AC7h	T3GPPS	3AA7h	—	3A87h	IOCEF	3A67h	IOCCF	3A47h	IOCAF	3A27h	—	3A07h	RA7PPS
3AE6h	U1CTSPPS	3AC6h	T3CKIPPS	3AA6h	—	3A86h	IOCEN	3A66h	IOCCN	3A46h	IOCAN	3A26h	—	3A06h	RA6PPS
3AE5h	U1RXPPS	3AC5h	T1GPPS	3AA5h	—	3A85h	IOCEP	3A65h	IOCCP	3A45h	IOCAP	3A25h	—	3A05h	RA5PPS
3AE4h	I2C2SDAPPS	3AC4h	T1CKIPPS	3AA4h	—	3A84h	INLVLE	3A64h	INLVLC	3A44h	INLVLA	3A24h	—	3A04h	RA4PPS
3AE3h	I2C2SCLPPS	3AC3h	T0CKIPPS	3AA3h	—	3A83h	SLRCONE(2)	3A63h	SLRCONC	3A43h	SLRCONA	3A23h	—	3A03h	RA3PPS
3AE2h	I2C1SDAPPS	3AC2h	INT2PPS	3AA2h	—	3A82h	ODCONE(2)	3A62h	ODCONC	3A42h	ODCONA	3A22h	RE2PPS(2)	3A02h	RA2PPS
3AE1h	I2C1SCLPPS	3AC1h	INT1PPS	3AA1h	—	3A81h	WPUE	3A61h	WPUC	3A41h	WPUA	3A21h	RE1PPS(2)	3A01h	RA1PPS
3AE0h	SPI1SSPPS	3AC0h	INT0PPS	3AA0h	—	3A80h	ANSELE(2)	3A60h	ANSELC	3A40h	ANSELA	3A20h	RE0PPS(2)	3A00h	RA0PPS

Legend: Unimplemented data memory locations and registers, read as '0'.

Note 1: Unimplemented in LF devices.

2: Unimplemented in PIC18(L)F26/27K42.

3: Unimplemented in PIC18(L)F26/27/45/46/47K42.

4: Reserved, maintain as '0'.

7.2 Clock Source Types

Clock sources can be classified as external or internal.

External clock sources rely on external circuitry for the clock source to function. Examples are: oscillator modules (ECH, ECM, ECL mode), quartz crystal resonators or ceramic resonators (LP, XT and HS modes).

Internal clock sources are contained within the oscillator module. The internal oscillator block has two internal oscillators that are used to generate internal system clock sources. The High-Frequency Internal Oscillator (HFINTOSC) can produce 1, 2, 4, 8, 12, 16, 32, 48 and 64 MHz clock. The frequency can be controlled through the OSCFRQ register (Register 7-5). The Low-Frequency Internal Oscillator (LFINTOSC) generates a fixed 31 kHz frequency.

A 4x PLL is provided that can be used with an external clock. When used with the HFINTOSC the 4x PLL has input frequency limitations.See **Section 7.2.1.4 "4x PLL"** for more details.

The system clock can be selected between external or internal clock sources via the NOSC bits in the OSCCON1 register. See **Section 7.3 "Clock Switching"** for additional information. The system clock can be made available on the OSC2/CLKOUT pin for any of the modes that do not use the OSC2 pin. The clock out functionality is governed by the $\overline{\text{CLKOUTEN}}$ bit in the CONFIG1H register (Register 5-2). If enabled, the clock out signal is always at a frequency of FOSC/4.

7.2.1 EXTERNAL CLOCK SOURCES

An external clock source can be used as the device system clock by performing one of the following actions:

- Program the RSTOSC<2:0> and FEXTOSC<2:0> bits in the Configuration Words to select an external clock source that will be used as the default system clock upon a device Reset.
- Write the NOSC<2:0> and NDIV<3:0> bits in the OSCCON1 register to switch the system clock source.

See **Section 7.3 "Clock Switching"** for more information.

7.2.1.1 EC Mode

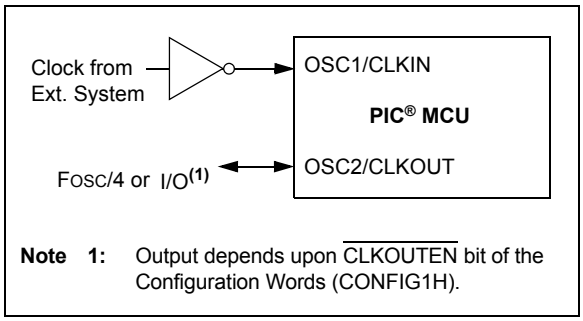
The External Clock (EC) mode allows an externally generated logic level signal to be the system clock source. When operating in this mode, an external clock source is connected to the OSC1 input. OSC2/CLKOUT is available for general purpose I/O or CLKOUT. Figure 7-2 shows the pin connections for EC mode.

EC mode has three power modes to select from through Configuration Words:

- ECH – High power
- ECM – Medium power
- ECL – Low power

Refer to Table 44-9 for External Clock/Oscillator Timing Requirements. The Oscillator Start-up Timer (OST) is disabled when EC mode is selected. Therefore, there is no delay in operation after a Power-on Reset (POR) or wake-up from Sleep. Because the PIC® MCU design is fully static, stopping the external clock input will have the effect of halting the device while leaving all data intact. Upon restarting the external clock, the device will resume operation as if no time had elapsed.

FIGURE 7-2: EXTERNAL CLOCK (EC) MODE OPERATION



Note 1: Output depends upon $\overline{\text{CLKOUTEN}}$ bit of the Configuration Words (CONFIG1H).

7.2.1.2 LP, XT, HS Modes

The LP, XT and HS modes support the use of quartz crystal resonators or ceramic resonators connected to OSC1 and OSC2 (Figure 7-3). The three modes select a low, medium or high gain setting of the internal inverter-amplifier to support various resonator types and speed.

LP Oscillator mode selects the lowest gain setting of the internal inverter-amplifier. LP mode current consumption is the least of the three modes. This mode is designed to drive only 32.768 kHz tuning-fork type crystals (watch crystals).

XT Oscillator mode selects the intermediate gain setting of the internal inverter-amplifier. XT mode current consumption is the medium of the three modes. This mode is best suited to drive resonators with a medium drive level specification (above 100 kHz - 8 MHz).

HS Oscillator mode selects the highest gain setting of the internal inverter-amplifier. HS mode current consumption is the highest of the three modes. This mode is best suited for resonators that require a high drive setting (above 8 MHz).

Figure 7-3 and Figure 7-4 show typical circuits for quartz crystal and ceramic resonators, respectively.

REGISTER 9-16: PIE2: PERIPHERAL INTERRUPT ENABLE REGISTER 2

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
I2C1RXIE	SPI1IE	SPI1TXIE	SPI1RXIE	DMA1AIE	DMA1ORIE	DMA1DCNTIE	DMA1SCNTIE
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	

bit 7 **I2C1RXIE:** I^2C1 Receive Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 6 **SPI1IE:** SPI1 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 5 **SPI1TXIE:** SPI1 Transmit Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 4 **SPI1RXIE:** SPI1 Receive Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 3 **DMA1AIE:** DMA1 Abort Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 2 **DMA1ORIE:** DMA1 Overrun Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 1 **DMA1DCNTIE:** DMA1 Destination Count Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 0 **DMA1SCNTIE:** DMA1 Source Count Interrupt Enable bit
1 = Enabled
0 = Disabled

13.4 Register Definitions: Nonvolatile Memory

REGISTER 13-1: NVMCON1: NONVOLATILE MEMORY CONTROL 1 REGISTER

R/W-0/0	R/W-0/0	U-0	R/S/HC-0/0	R/W/HS-x/q	R/W-0/0	R/S/HC-0/0	R/S/HC-0/0
REG<1:0>		—	FREE	WRERR	WREN	WR	RD
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	HC = Bit is cleared by hardware
x = Bit is unknown	-n = Value at POR	S = Bit can be set by software, but not cleared
'0' = Bit is cleared	'1' = Bit is set	U = Unimplemented bit, read as '0'

bit 7-6 **REG<1:0>:** NVM Region Selection bit
10 =Access PFM Locations
x1 = Access User IDs, Configuration Bits, DIA, DCI, Rev ID and Device ID
00 = Access Data EEPROM Memory Locations

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

bit 4 **FREE:** Program Flash Memory Erase Enable bit(1)
1 = Performs an erase operation on the next WR command
0 = The next WR command performs a write operation

bit 3 **WRERR:** Write-Reset Error Flag bit(2,3,4)
1 = A write operation was interrupted by a Reset (hardware set),
or WR was written to 0b1 when an invalid address is accessed (Table 9-1, Table 13-1)
or WR was written to 0b1 when REG<1:0> and address do not point to the same region
or WR was written to 0b1 when a write-protected address is accessed (Table 9-2).
0 = All write operations have completed normally

bit 2 **WREN:** Program/Erase Enable bit
1 = Allows program/erase and refresh cycles
0 = Inhibits programming/erasing and user refresh of NVM

bit 1 **WR:** Write Control bit(5,6,7)
When REG points to a Data EEPROM Memory location:
1 = Initiates an erase/program cycle at the corresponding Data EEPROM Memory location
When REG points to a PFM location:
1 = Initiates the PFM write operation with data from the holding registers
0 = NVM program/erase operation is complete and inactive

bit 0 **RD:** Read Control bit(8)
1 = Initiates a read at address pointed by REG and NVMADR, and loads data into NVMDAT
0 = NVM read operation is complete and inactive

Note 1: This can only be used with PFM.
2: This bit is set when WR = 1 and clears when the internal programming timer expires or the write is completed successfully.
3: Bit must be cleared by the user; hardware will not clear this bit.
4: Bit may be written to '1' by the user in order to implement test sequences.
5: This bit can only be set by following the unlock sequence of **Section 13.1.4 "NVM Unlock Sequence"**.
6: Operations are self-timed and the WR bit is cleared by hardware when complete.
7: Once a write operation is initiated, setting this bit to zero will have no effect.
8: The bit can only be set in software. The bit is cleared by hardware when the operation is complete.

14.5 CRC Check Value

The CRC check value will be located in the CRCACC registers after the CRC calculation has finished. The check value will depend on two mode settings of the CRCCON0 register: ACCM and SHIFTM. When the ACCM bit is set, the CRC module augments the data with a number of zeros equal to the length of the polynomial to align the final check value. When the ACCM bit is not set, the CRC will stop at the end of the data. A number of zeros equal to the length of the polynomial can then be entered into CRCDAT to find the same check value as augmented mode. Alternatively, the expected check value can be entered at this point to make the final result equal '0'.

When the CRC check value is computed with the SHIFTM bit set, selecting LSb first, and the ACCM bit is also set, then the final value in the CRCACC registers will be reversed such that the LSb will be in the MSb position and vice versa. This is the expected check value in bit reversed form. If you are creating a check value to be appended to a data stream, then a bit reversal must be performed on the final value to achieve the correct checksum. You can use the CRC to do this reversal by the following method:

- Save the CRCACC value in user RAM space
- Clear the CRCACC registers
- Clear the CRCXOR registers
- Write the saved CRCACC value to the CRCDAT input.

The properly oriented check value will be in the CRCACC registers as the result.

14.6 CRC Interrupt

The CRC will generate an interrupt when the BUSY bit transitions from 1 to 0. The CRCIF Interrupt Flag is set every time the BUSY bit transitions, regardless of whether or not the CRC interrupt is enabled. The CRCIF bit can only be cleared in software.

14.7 Configuring the CRC

The following steps illustrate how to properly configure the CRC.

1. Determine if the automatic program memory scan will be used with the scanner or manual calculation through the SFR interface and perform the actions specified in **Section 14.4 "CRC Data Sources"**, depending on which decision was made.
2. If desired, seed a starting CRC value into the CRCACCH/L registers.
3. Program the CRCXORH/L registers with the desired generator polynomial.
4. Program the DLEN<3:0> bits of the CRCCON1 register with the length of the data word - 1 (refer to Example 14-1). This determines how many times the shifter will shift into the accumulator for each data word.
5. Program the PLEN<3:0> bits of the CRCCON1 register with the length of the polynomial -2 (refer to Example 14-1).
6. Determine whether shifting in trailing zeros is desired and set the ACCM bit of the CRCCON0 register appropriately.
7. Likewise, determine whether the MSb or LSb should be shifted first and write the SHIFTM bit of the CRCCON0 register appropriately.
8. Write the GO bit of the CRCCON0 register to begin the shifting process.

9a. If manual SFR entry is used, monitor the FULL bit of the CRCCON0 register. When FULL = 0, another word of data can be written to the CRCDATH/L registers, keeping in mind that CRCDATH should be written first if the data has more than eight bits, as the shifter will begin upon the CRCDATL register being written.

9b. If the scanner is used, the scanner will automatically load words into the CRCDATH/L registers as needed, as long as the GO bit is set.

10a. If manual entry is used, monitor the CRCIF (and BUSY bit to determine when the completed CRC calculation can be read from CRCACCH/L registers.

10b. If using the memory scanner, monitor the SCANIF (or the GO bit) for the scanner to finish pushing information into the CRCDAT registers. After the scanner is completed, monitor the BUSY bit to determine that the CRC has been completed and the check value can be read from the CRCACC registers. If both the interrupt flags are set and the BUSY and GO bits are cleared, the completed CRC calculation can be read from the CRCACCH/L registers.

16.0 I/O PORTS

The PIC18(L)F26/27/45/46/47/55/56/57K42 devices have six I/O ports, allocated as shown in Table 16-1.

TABLE 16-1: PORT ALLOCATION TABLE FOR PIC18(L)F26/27/45/46/47/55/56/57K42 DEVICES

Device	PORTA	PORTB	PORTC	PORTD	PORTE	PORTF
PIC18(L)F26K42	•	•	•		•(1)	
PIC18(L)F27K42	•	•	•		•(1)	
PIC18(L)F45K42	•	•	•	•	•(2)	
PIC18(L)F46K42	•	•	•	•	•(2)	
PIC18(L)F47K42	•	•	•	•	•(2)	
PIC18(L)F55K42	•	•	•	•	•(2)	•
PIC18(L)F56K42	•	•	•	•	•(2)	•
PIC18(L)F57K42	•	•	•	•	•(2)	•

Note 1: Pin RE3 only.
2: Pins RE0, RE1, RE2 and RE3 only.

Each port has ten registers to control the operation. These registers are:

- PORTx registers (reads the levels on the pins of the device)
- LATx registers (output latch)
- TRISx registers (data direction)
- ANSELx registers (analog select)
- WPUx registers (weak pull-up)
- INLVLx (input level control)
- SLRCONx registers (slew rate control)
- ODCONx registers (open-drain control)

Most port pins share functions with device peripherals, both analog and digital. In general, when a peripheral is enabled on a port pin, that pin cannot be used as a general purpose output; however, the pin can still be read.

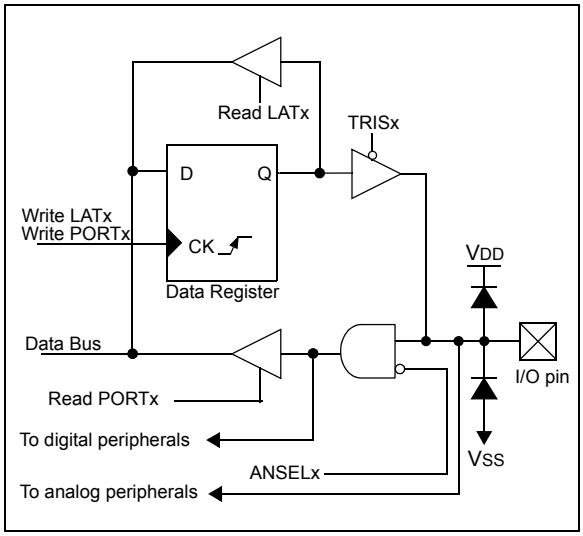
The Data Latch (LATx registers) is useful for read-modify-write operations on the value that the I/O pins are driving.

A write operation to the LATx register has the same effect as a write to the corresponding PORTx register. A read of the LATx register reads of the values held in the I/O PORT latches, while a read of the PORTx register reads the actual I/O pin value.

Ports that support analog inputs have an associated ANSELx register. When an ANSELx bit is set, the digital input buffer associated with that bit is disabled.

Disabling the input buffer prevents analog signal levels on the pin between a logic high and low from causing excessive current in the logic input circuitry. A simplified model of a generic I/O port, without the interfaces to other peripherals, is shown in Figure 16-1.

FIGURE 16-1: GENERIC I/O PORT OPERATION



16.1 I/O Priorities

Each pin defaults to the PORT data latch after Reset. Other functions are selected with the peripheral pin select logic. See **Section 17.0 "Peripheral Pin Select (PPS) Module"** for more information.

Analog input functions, such as ADC and comparator inputs, are not shown in the peripheral pin select lists. These inputs are active when the I/O pin is set for Analog mode using the ANSELx register. Digital output functions may continue to control the pin when it is in Analog mode.

Analog outputs, when enabled, take priority over digital outputs and force the digital output driver into a high-impedance state.

The pin function priorities are as follows:

1. Configuration bits
2. Analog outputs (disable the input buffers)
3. Analog inputs
4. Port inputs and outputs from PPS

16.2 PORTx Registers

In this section, the generic names such as PORTx, LATx, TRISx, etc. can be associated with PORTA, PORTB, and PORTC. The functionality of PORTE is different compared to other ports and is explained in a separate section.

TABLE 16-11: SUMMARY OF REGISTERS ASSOCIATED WITH I/O (CONTINUED)

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
INLVLC	INLVLC7	INLVLC6	INLVLC5	INLVLC4(5)	INLVLC3(5)	INLVLC2	INLVLC1	INLVLC0	270
INLVLD(6)	INLVLD7	INLVLD6	INLVLD5	INLVLD4	INLVLD3	INLVLD2	INLVLD1(5)	INLVLD0(5)	270
INLVLF(7)	INLVLF7	INLVLF6	INLVLF5	INLVLF4	INLVLF3	INLVLF2	INLVLF1	INLVLF0	270
INLVLE	—	—	—	—	INLVLE3	—	—	—	270
RB1I2C	—	SLEW	PU<1:0>		—	—	TH<1:0>		271
RB2I2C	—	SLEW	PU<1:0>		—	—	TH<1:0>		271
RC3I2C	—	SLEW	PU<1:0>		—	—	TH<1:0>		271
RC4I2C	—	SLEW	PU<1:0>		—	—	TH<1:0>		271
RD0I2C(6)	—	SLEW	PU<1:0>		—	—	TH<1:0>		271
RD1I2C(6)	—	SLEW	PU<1:0>		—	—	TH<1:0>		271

Legend: — = unimplemented location, read as '0'. Shaded cells are not used by I/O Ports.

Note 1: Bits RB6 and RB7 read '1' while in Debug mode.
2: Bit PORTE3 is read-only, and will read '1' when MCLRE = 1 (Master Clear enabled).
3: Bits RB6 and RB7 read '1' while in Debug mode.
4: If MCLRE = 1, the weak pull-up in RE3 is always enabled; bit WPUE3 is not affected.
5: Any peripheral using the I^2C pins read the I^2C ST inputs when enabled via RxyI2C.
6: Unimplemented in PIC18(L)F26/27K42.
7: Unimplemented in PIC18(L)F26/27/45/46/47K42 parts.

17.0 PERIPHERAL PIN SELECT (PPS) MODULE

The Peripheral Pin Select (PPS) module connects peripheral inputs and outputs to the device I/O pins. Only digital signals are included in the selections. All analog inputs and outputs remain fixed to their assigned pins. Input and output selections are independent as shown in the simplified block diagram Figure 17-1.

The peripheral input is selected with the peripheral xxxPPS register (Register 17-1), and the peripheral output is selected with the PORT RxyPPS register (Register 17-2). For example, to select PORTC<7> as the UART1 RX input, set U1RXPPS to 0b1 0111, and to select PORTC<6> as the UART1 TX output set RC6PPS to 0b01 0011.

17.1 PPS Inputs

Each peripheral has a PPS register with which the inputs to the peripheral are selected. Inputs include the device pins.

Multiple peripherals can operate from the same source simultaneously. Port reads always return the pin level regardless of peripheral PPS selection. If a pin also has analog functions associated, the ANSEL bit for that pin must be cleared to enable the digital input buffer.

Although every peripheral has its own PPS input selection register, the selections are identical for every peripheral as shown in Register 17-1.

Note:	The notation “xxx” in the register name is a place holder for the peripheral identifier. For example, INT0PPS.

17.2 PPS Outputs

Each I/O pin has a PPS register with which the pin output source is selected. With few exceptions, the port TRIS control associated with that pin retains control over the pin output driver. Peripherals that control the pin output driver as part of the peripheral operation will override the TRIS control as needed. These peripherals include:

- UART

Although every pin has its own PPS peripheral selection register, the selections are identical for every pin as shown in Register 17-2.

Note:	The notation “Rxy” is a place holder for the pin identifier. For example, RA0PPS.

18.6 Register Definitions: Interrupt-on-Change Control

REGISTER 18-1: IOCxP: INTERRUPT-ON-CHANGE POSITIVE EDGE REGISTER EXAMPLE

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
IOCxP7	IOCxP6	IOCxP5	IOCxP4	IOCxP3	IOCxP2	IOCxP1	IOCxP0
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-0 **IOCxP<7:0>:** Interrupt-on-Change Positive Edge Enable bits
1 = Interrupt-on-Change enabled on the IOCx pin for a positive-going edge. Associated Status bit and interrupt flag will be set upon detecting an edge.
0 = Interrupt-on-Change disabled for the associated pin.

REGISTER 18-2: IOCxN: INTERRUPT-ON-CHANGE NEGATIVE EDGE REGISTER EXAMPLE

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
IOCxN7	IOCxN6	IOCxN5	IOCxN4	IOCxN3	IOCxN2	IOCxN1	IOCxN0
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-0 **IOCxN<7:0>:** Interrupt-on-Change Negative Edge Enable bits
1 = Interrupt-on-Change enabled on the IOCx pin for a negative-going edge. Associated Status bit and interrupt flag will be set upon detecting an edge.
0 = Interrupt-on-Change disabled for the associated pin

REGISTER 18-3: IOCxF: INTERRUPT-ON-CHANGE FLAG REGISTER EXAMPLE

R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0
IOCxF7	IOCxF6	IOCxF5	IOCxF4	IOCxF3	IOCxF2	IOCxF1	IOCxF0
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	HS - Bit is set in hardware

bit 7-0 **IOCxF<7:0>:** Interrupt-on-Change Flag bits
1 = A enabled change was detected on the associated pin. Set when IOCP[n] = 1 and a positive edge was detected on the IOCn pin, or when IOCN[n] = 1 and a negative edge was detected on the IOCn pin
0 = No change was detected, or the user cleared the detected change

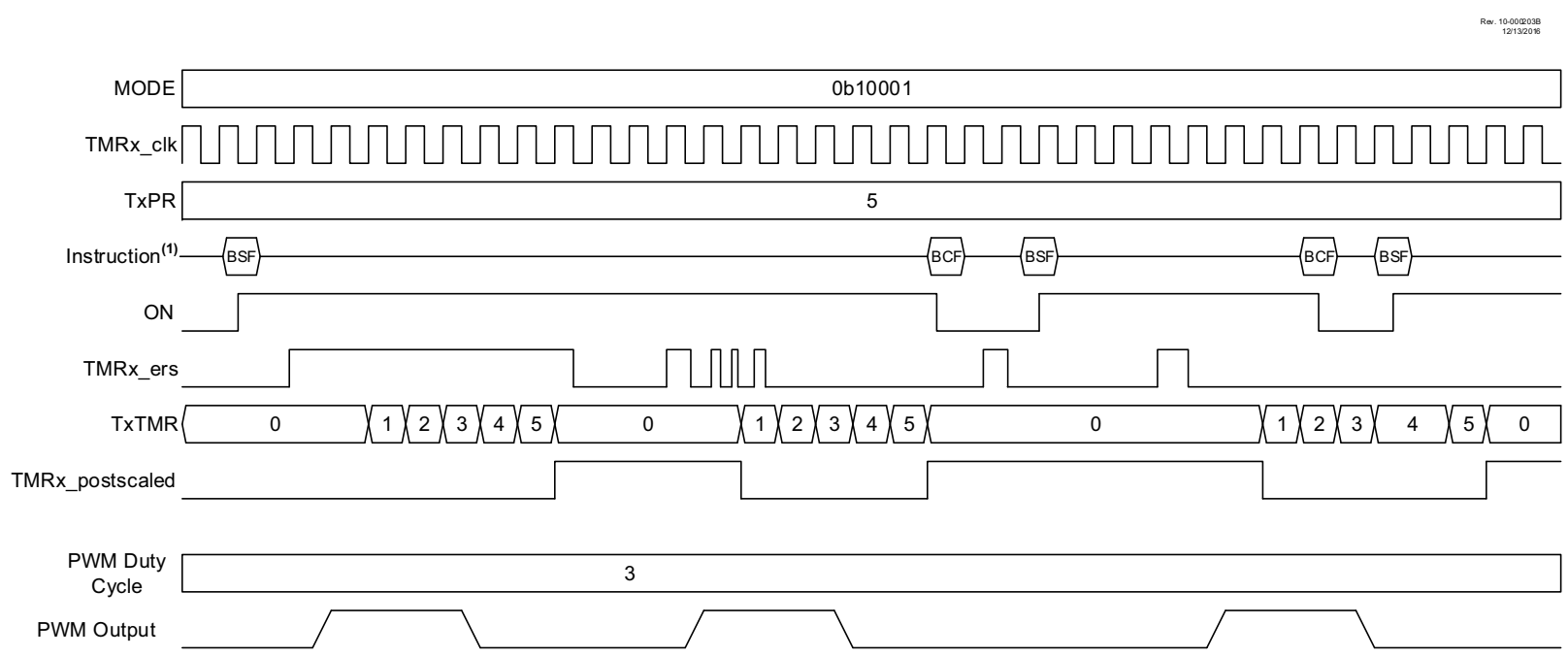
22.5.9 EDGE-TRIGGERED MONOSTABLE MODES

The Edge-Triggered Monostable modes start the timer on an edge from the external Reset signal input, after the ON bit is set, and stop incrementing the timer when the timer matches the T2PR period value. The following edges will start the timer:

- Rising edge (MODE<4:0> = 10001)
- Falling edge (MODE<4:0> = 10010)
- Rising or Falling edge (MODE<4:0> = 10011)

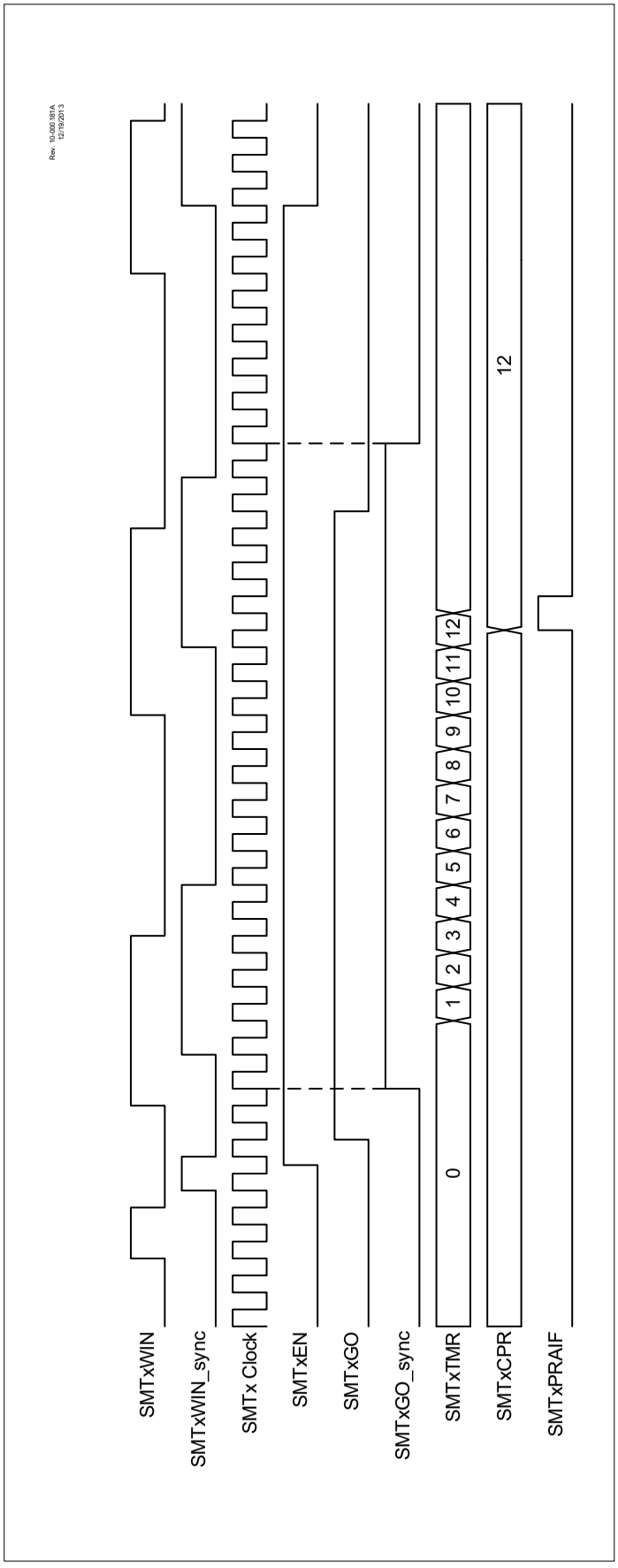
When an Edge-Triggered Monostable mode is used in conjunction with the CCP PWM operation the PWM drive goes active with the external Reset signal edge that starts the timer, but will not go active when the timer matches the T2PR value. While the timer is incrementing, additional edges on the external Reset signal will not affect the CCP PWM.

FIGURE 22-12: RISING EDGE-TRIGGERED MONOSTABLE MODE TIMING DIAGRAM (MODE = 10001)



Note 1: BSF and BCF represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

FIGURE 25-11: WINDOWED MEASURE MODE SINGLE ACQUISITION TIMING DIAGRAM

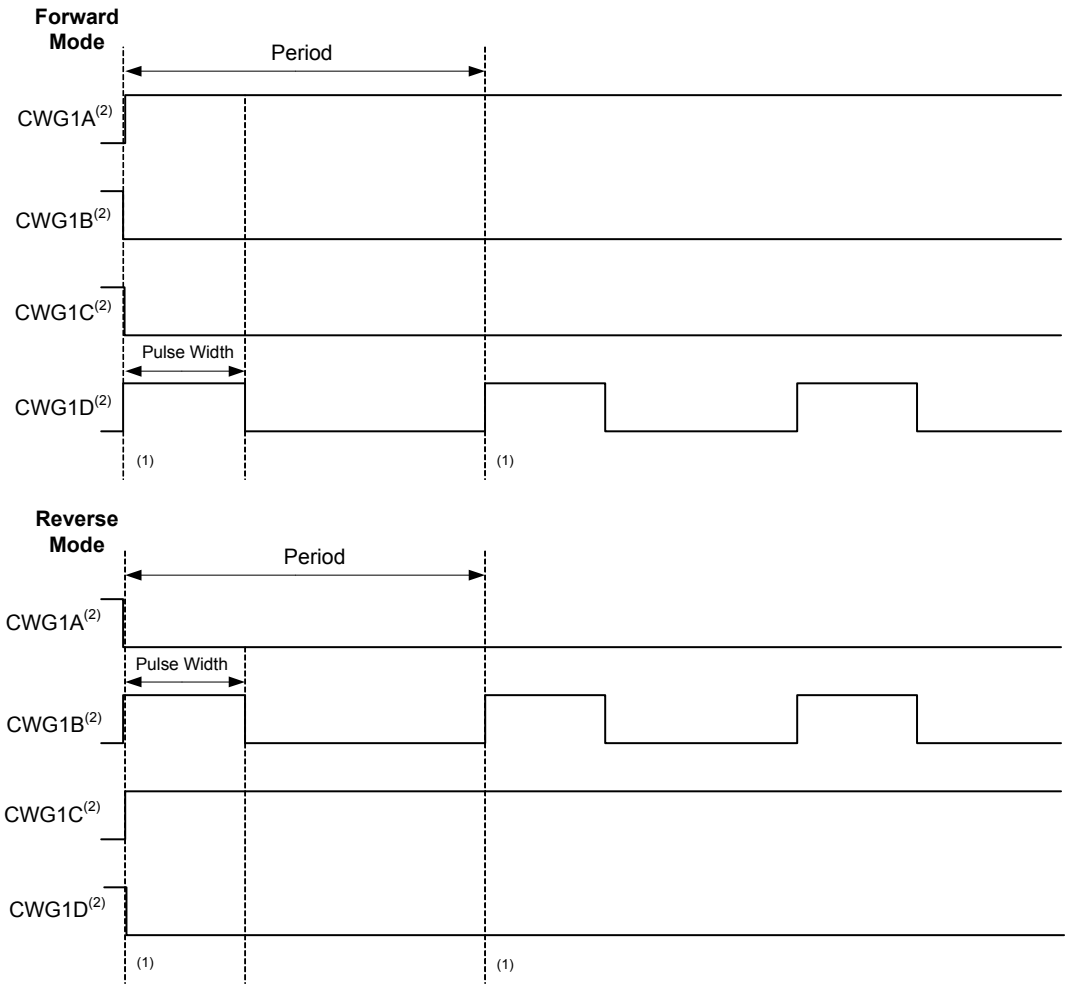


In Forward Full-Bridge mode (MODE<2:0> = 010), CWGxA is driven to its active state, CWGxB and CWGxC are driven to their inactive state, and CWGxD is modulated by the input signal, as shown in Figure 26-7.

In Reverse Full-Bridge mode (MODE<2:0> = 011), CWGxC is driven to its active state, CWGxA and CWGxD are driven to their inactive states, and CWGxB is modulated by the input signal, as shown in Figure 26-7.

In Full-Bridge mode, the dead-band period is used when there is a switch from forward to reverse or vice-versa. This dead-band control is described in **Section 26.6 "Dead-Band Control"**, with additional details in **Section 26.7 "Rising Edge and Reverse Dead Band"** and **Section 26.8 "Falling Edge and Forward Dead Band"**. Steering modes are not used with either of the Full-Bridge modes. The mode selection may be toggled between forward and reverse toggling the MODE<0> bit of the CWGxCON0 while keeping MODE<2:1> static, without disabling the CWG module.

FIGURE 26-7: EXAMPLE OF FULL-BRIDGE OUTPUT



Note 1: A rising CWG data input creates a rising event on the modulated output.
2: Output signals shown as active-high; all POLy bits are clear.

REGISTER 27-11: CLCDATA: CLC DATA OUTPUT

U-0	U-0	U-0	U-0	R-0	R-0	R-0	R-0
—	—	—	—	CLC4OUT	CLC3OUT	CLC2OUT	CLC1OUT
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	

bit 7-4 **Unimplemented:** Read as '0'

bit 3 **CLC4OUT:** Mirror copy of OUT bit of CLC4CON register

bit 2 **CLC3OUT:** Mirror copy of OUT bit of CLC3CON register

bit 1 **CLC2OUT:** Mirror copy of OUT bit of CLC2CON register

bit 0 **CLC1OUT:** Mirror copy of OUT bit of CLC1CON register

TABLE 27-3: SUMMARY OF REGISTERS ASSOCIATED WITH CLCx

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
CLCxCON	EN	—	OUT	INTP	INTN	MODE<2:0>			440
CLCxPOL	POL	—	—	—	G4POL	G3POL	G2POL	G1POL	441
CLCxSEL0	—	—	D1S<5:0>						442
CLCxSEL1	—	—	D2S<5:0>						442
CLCxSEL2	—	—	D3S<5:0>						442
CLCxSEL3	—	—	D4S<5:0>						442
CLCxGLS0	G1D4T	G1D4N	G1D3T	G1D3N	G1D2T	G1D2N	G1D1T	G1D1N	443
CLCxGLS1	G2D4T	G2D4N	G2D3T	G2D3N	G2D2T	G2D2N	G2D1T	G2D1N	444
CLCxGLS2	G3D4T	G3D4N	G3D3T	G3D3N	G3D2T	G3D2N	G3D1T	G3D1N	445
CLCxGLS3	G4D4T	G4D4N	G4D3T	G4D3N	G4D2T	G4D2N	G4D1T	G4D1N	446
CLCDATA	—	—	—	—	CLC4OUT	CLC3OUT	CLC2OUT	CLC1OUT	447

Legend: — = unimplemented, read as '0'. Shaded cells are unused by the CLCx modules.

29.10 Register Definitions: ZCD Control

REGISTER 29-1: ZCDCON: ZERO-CROSS DETECT CONTROL REGISTER

R/W-0/0	U-0	R-x	R/W-0/0	U-0	U-0	R/W-0/0	R/W-0/0
SEN	—	OUT	POL	—	—	INTP	INTN
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 **SEN:** Zero-Cross Detect Software Enable bit
This bit is ignored when ZCDSEN configuration bit is set.
1= Zero-cross detect is enabled.
0= Zero-cross detect is disabled. ZCD pin operates according to PPS and TRIS controls.

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

bit 5 **OUT:** Zero-Cross Detect Data Output bit
ZCDPOL bit = 0:
1 = ZCD pin is sinking current
0 = ZCD pin is sourcing current
ZCDPOL bit = 1:
1 = ZCD pin is sourcing current
0 = ZCD pin is sinking current

bit 4 **POL:** Zero-Cross Detect Polarity bit
1 = ZCD logic output is inverted
0 = ZCD logic output is not inverted

bit 3-2 **Unimplemented:** Read as '0'

bit 1 **INTP:** Zero-Cross Detect Positive-Going Edge Interrupt Enable bit
1 = ZCDIF bit is set on low-to-high ZCD_output transition
0 = ZCDIF bit is unaffected by low-to-high ZCD_output transition

bit 0 **INTN:** Zero-Cross Detect Negative-Going Edge Interrupt Enable bit
1 = ZCDIF bit is set on high-to-low ZCD_output transition
0 = ZCDIF bit is unaffected by high-to-low ZCD_output transition

TABLE 29-1: SUMMARY OF REGISTERS ASSOCIATED WITH THE ZCD MODULE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
ZCDCON	SEN	—	OUT	POL	—	—	INTP	INTN	462

Legend: — = unimplemented, read as '0'. Shaded cells are unused by the ZCD module.

REGISTER 30-3: MD1CARH: MODULATION HIGH CARRIER CONTROL REGISTER

U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	CH<4:0>(1)				
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	

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

bit 4-0 **CH<4:0>:** Modulator Carrier High Selection bits(1)
See Table 30-1 for signal list

Note 1: Unused selections provide an input value.

REGISTER 30-4: MD1CARL: MODULATION LOW CARRIER CONTROL REGISTER

U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	CL<4:0>(1)				
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	

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

bit 4-0 **CL<4:0>:** Modulator Carrier Low Input Selection bits(1)
See Table 30-1 for signal list

Note 1: Unused selections provide a zero as the input value.

32.7 SPI Operation in Sleep Mode

SPI master mode will operate in Sleep, provided the clock source selected by SPIxCLK is active in Sleep mode. FIFOs will operate as they would when the part is awake. When TXR = 1, the TXFIFO will need to contain data in order for transfers to take place in Sleep. All interrupts will still set the interrupt flags in Sleep but only enabled interrupts will wake the device from Sleep.

SPI Slave mode will operate in Sleep, because the clock is provided by an external master device. FIFOs will still operate and interrupts will set interrupt flags, and enabled interrupts will wake the device from Sleep.

32.8 SPI Interrupts

There are three top level SPI interrupts in the PIRx register:

- SPI Transmit
- SPI Receive
- SPI Module status

The status interrupts are enabled at the module level in the SPIxINTE register. Only enabled status interrupts will cause the single top level SPIxIF flag to be set.

32.8.1 SPI RECEIVER DATA INTERRUPT

The SPI Receiver Data Interrupt is set when RXFIFO contains data, and is cleared when the RXFIFO is empty. The interrupt flag SPI1RXIF is located in PIRx and the interrupt enable SPI1RXIE is located in PIEx. This interrupt flag is read-only.

32.8.2 SPI TRANSMITTER DATA INTERRUPT

The SPI Transmitter Data Interrupt is set when TXFIFO is not full, and is cleared when the TXFIFO is full. The interrupt flag SPI1TXIF is located in PIRx and the interrupt enable SPI1TXIE is located in PIEx. The interrupt flag is read-only.

32.8.3 SPI MODULE STATUS INTERRUPTS

The SPIxIF flag in the respective PIR register is set when any of the individual status flags in SPIxINTF and their respective SPIxINTE bits are set. In order for the setting of any specific interrupt flag to interrupt normal program flow both the SPIxIE bit as well as the specific bit in SPIxINTE associated with that interrupt must be set.

The Status Interrupts are:

- Shift Register Empty Interrupt
- Transfer Counter is Zero Interrupt
- Start of Slave Select Interrupt
- End of Slave Select Interrupt
- Receiver Overflow Interrupt
- Transmitter Underflow Interrupt

REGISTER 37-2: DAC1CON1: DAC DATA REGISTER

U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	DATA<4:0>				
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	

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

bit 4-0 **DATA<4:0>:** Data Input Register for DAC bits

TABLE 37-1: SUMMARY OF REGISTERS ASSOCIATED WITH THE DAC MODULE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
DAC1CON0	EN	—	OE1	OE2	PSS<1:0>		—	NSS	640
DAC1CON1	—	—	—	DATA<4:0>					641

Legend: — = Unimplemented location, read as '0'. Shaded cells are not used with the DAC module.

BNC **Branch if Not Carry**

Syntax: BNC n

Operands: -128 ≤ n ≤ 127

Operation: if CARRY bit is '0'
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0011	nnnn	nnnn

Description: If the CARRY bit is '0', then the program will branch.
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a 2-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

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

If No Jump:

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

Example: `HERE BNC Jump`

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

BNN **Branch if Not Negative**

Syntax: BNN n

Operands: -128 ≤ n ≤ 127

Operation: if NEGATIVE bit is '0'
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0111	nnnn	nnnn

Description: If the NEGATIVE bit is '0', then the program will branch.
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a 2-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

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

If No Jump:

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

Example: `HERE BNN Jump`

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

CALLW **Subroutine Call Using WREG**

Syntax: CALLW

Operands: None

Operation: (PC + 2) → TOS,
(W) → PCL,
(PCLATH) → PCH,
(PCLATU) → PCU

Status Affected: None

Encoding:

0000	0000	0001	0100

Description: First, the return address (PC + 2) is pushed onto the return stack. Next, the contents of W are written to PCL; the existing value is discarded. Then, the contents of PCLATH and PCLATU are latched into PCH and PCU, respectively. The second cycle is executed as a NOP instruction while the new next instruction is fetched. Unlike CALL, there is no option to update W, Status or BSR.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read WREG	PUSH PC to stack	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    CALLW
```
Before Instruction
PC = address (HERE)
PCLATH = 10h
PCLATU = 00h
W = 06h
After Instruction
PC = 001006h
TOS = address (HERE + 2)
PCLATH = 10h
PCLATU = 00h
W = 06h

CLRF **Clear f**

Syntax: CLRF f {,a}

Operands: 0 ≤ f ≤ 255
a ∈ [0,1]

Operation: 000h → f
1 → Z

Status Affected: Z

Encoding:

0110	101a	ffff	ffff

Description: Clears the contents of the specified register.
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 41.2.3 "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:
```
CLRF    FLAG_REG, 1
```
Before Instruction
FLAG_REG = 5Ah
After Instruction
FLAG_REG = 00h

CPFSLT **Compare f with W, skip if f < W**

Syntax: CPFSLT f {,a}

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

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

Status Affected: None

Encoding:

0110	000a	ffff	ffff

Description: Compares the contents of data memory location 'f' to the contents of W by performing an unsigned subtraction. If the contents of 'f' are less than the contents of W, then the fetched instruction is discarded and a NOP is executed instead, making this a 2-cycle instruction. If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    CPFSLT REG, 1
NLESS   :
LESS    :
```

Before Instruction
PC = Address (HERE)
W = ?
After Instruction
If REG < W;
PC = Address (LESS)
If REG ≥ W;
PC = Address (NLESS)

DAW **Decimal Adjust W Register**

Syntax: DAW

Operands: None

Operation: If [W<3:0> > 9] or [DC = 1] then
(W<3:0>) + 6 → W<3:0>;
else
(W<3:0>) → W<3:0>;

If [W<7:4> + DC > 9] or [C = 1] then
(W<7:4>) + 6 + DC → W<7:4> ;
else
(W<7:4>) + DC → W<7:4>

Status Affected: C

Encoding:

0000	0000	0000	0111

Description: DAW adjusts the 8-bit value in W, resulting from the earlier addition of two variables (each in packed BCD format) and produces a correct packed BCD result.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register W	Process Data	Write W

Example1:
```
DAW
```
Before Instruction
W = A5h
C = 0
DC = 0
After Instruction
W = 05h
C = 1
DC = 0

Example 2:
Before Instruction
W = CEh
C = 0
DC = 0
After Instruction
W = 34h
C = 1
DC = 0