



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, I ² C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	22
Program Memory Size	16KB (8K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	768 x 8
Voltage - Supply (Vcc/Vdd)	4.2V ~ 5.5V
Data Converters	A/D 5x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Through Hole
Package / Case	28-DIP (0.300", 7.62mm)
Supplier Device Package	28-SPDIP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f248-e-sp

28/40-Pin High-Performance, Enhanced Flash Microcontrollers with CAN

High-Performance RISC CPU:

- Linear program memory addressing up to 2 Mbytes
- Linear data memory addressing to 4 Kbytes
- Up to 10 MIPS operation
- DC – 40 MHz clock input
- 4 MHz-10 MHz oscillator/clock input with PLL active
- 16-bit wide instructions, 8-bit wide data path
- Priority levels for interrupts
- 8 x 8 Single-Cycle Hardware Multiplier

Peripheral Features:

- High current sink/source 25 mA/25 mA
- Three external interrupt pins
- Timer0 module: 8-bit/16-bit timer/counter with 8-bit programmable prescaler
- Timer1 module: 16-bit timer/counter
- Timer2 module: 8-bit timer/counter with 8-bit period register (time base for PWM)
- Timer3 module: 16-bit timer/counter
- Secondary oscillator clock option – Timer1/Timer3
- Capture/Compare/PWM (CCP) modules; CCP pins can be configured as:
 - Capture input: 16-bit, max resolution 6.25 ns
 - Compare: 16-bit, max resolution 100 ns (T_{CY})
 - PWM output: PWM resolution is 1 to 10-bit
Max. PWM freq. @ 8-bit resolution = 156 kHz
10-bit resolution = 39 kHz
- Enhanced CCP module which has all the features of the standard CCP module, but also has the following features for advanced motor control:
 - 1, 2 or 4 PWM outputs
 - Selectable PWM polarity
 - Programmable PWM dead time
- Master Synchronous Serial Port (MSSP) with two modes of operation:
 - 3-wire SPI™ (Supports all 4 SPI modes)
 - I²C™ Master and Slave mode
- Addressable USART module:
 - Supports interrupt-on-address bit

Advanced Analog Features:

- 10-bit, up to 8-channel Analog-to-Digital Converter module (A/D) with:
 - Conversion available during Sleep
 - Up to 8 channels available
- Analog Comparator module:
 - Programmable input and output multiplexing
- Comparator Voltage Reference module
- Programmable Low-Voltage Detection (LVD) module:
 - Supports interrupt-on-Low-Voltage Detection
- Programmable Brown-out Reset (BOR)

CAN bus Module Features:

- Complies with ISO CAN Conformance Test
- Message bit rates up to 1 Mbps
- Conforms to CAN 2.0B Active Spec with:
 - 29-bit Identifier Fields
 - 8-byte message length
 - 3 Transmit Message Buffers with prioritization
 - 2 Receive Message Buffers
 - 6 full, 29-bit Acceptance Filters
 - Prioritization of Acceptance Filters
 - Multiple Receive Buffers for High Priority Messages to prevent loss due to overflow
 - Advanced Error Management Features

Special Microcontroller Features:

- Power-on Reset (POR), Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)
- Watchdog Timer (WDT) with its own on-chip RC oscillator
- Programmable code protection
- Power-saving Sleep mode
- Selectable oscillator options, including:
 - 4x Phase Lock Loop (PLL) of primary oscillator
 - Secondary Oscillator (32 kHz) clock input
- In-Circuit Serial Programming™ (ICSP™) via two pins

Flash Technology:

- Low-power, high-speed Enhanced Flash technology
- Fully static design
- Wide operating voltage range (2.0V to 5.5V)
- Industrial and Extended temperature ranges

2.7 Effects of Sleep Mode on the On-Chip Oscillator

When the device executes a `SLEEP` instruction, the on-chip clocks and oscillator are turned off and the device is held at the beginning of an instruction cycle (Q1 state). With the oscillator off, the OSC1 and OSC2 signals will stop oscillating. Since all the transistor switching currents have been removed, Sleep mode achieves the lowest current consumption of the device (only leakage currents). Enabling any on-chip feature that will operate during Sleep will increase the current consumed during Sleep. The user can wake from Sleep through external Reset, Watchdog Timer Reset or through an interrupt.

2.8 Power-up Delays

Power-up delays are controlled by two timers so that no external Reset circuitry is required for most applications. The delays ensure that the device is kept in

Reset until the device power supply and clock are stable. For additional information on Reset operation, see **Section 3.0 “Reset”**.

The first timer is the Power-up Timer (PWRT), which optionally provides a fixed delay of T_{PWRT} (parameter #D033) on power-up only (POR and BOR). The second timer is the Oscillator Start-up Timer (OST), intended to keep the chip in Reset until the crystal oscillator is stable.

With the PLL enabled (HS4 Oscillator mode), the time-out sequence following a Power-on Reset is different from other oscillator modes. The time-out sequence is as follows: the PWRT time-out is invoked after a POR time delay has expired, then the Oscillator Start-up Timer (OST) is invoked. However, this is still not a sufficient amount of time to allow the PLL to lock at high frequencies. The PWRT timer is used to provide an additional fixed 2 ms (nominal) to allow the PLL ample time to lock to the incoming clock frequency.

TABLE 2-3: OSC1 AND OSC2 PIN STATES IN SLEEP MODE

OSC Mode	OSC1 Pin	OSC2 Pin
RC	Floating, external resistor should pull high	At logic low
RCIO	Floating, external resistor should pull high	Configured as PORTA, bit 6
ECIO	Floating	Configured as PORTA, bit 6
EC	Floating	At logic low
LP, XT and HS	Feedback inverter disabled at quiescent voltage level	Feedback inverter disabled at quiescent voltage level

Note: See Table 3-1 in **Section 3.0 “Reset”** for time-outs due to Sleep and $\overline{\text{MCLR}}$ Reset.

PIC18FXX8

REGISTER 8-6: PIR3: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 3

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	TXB0IF	RXB1IF	RXB0IF
bit 7						bit 0	

- bit 7 **IRXIF:** Invalid Message Received Interrupt Flag bit
1 = An invalid message has occurred on the CAN bus
0 = An invalid message has not occurred on the CAN bus
- bit 6 **WAKIF:** Bus Activity Wake-up Interrupt Flag bit
1 = Activity on the CAN bus has occurred
0 = Activity on the CAN bus has not occurred
- bit 5 **ERRIF:** CAN bus Error Interrupt Flag bit
1 = An error has occurred in the CAN module (multiple sources)
0 = An error has not occurred in the CAN module
- bit 4 **TXB2IF:** 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
- bit 3 **TXB1IF:** Transmit Buffer 1 Interrupt Flag bit
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:** Transmit Buffer 0 Interrupt Flag bit
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 **RXB1IF:** 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
- bit 0 **RXB0IF:** 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

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

PIC18FXX8

FIGURE 9-1: RA3:RA0 AND RA5 PINS BLOCK DIAGRAM

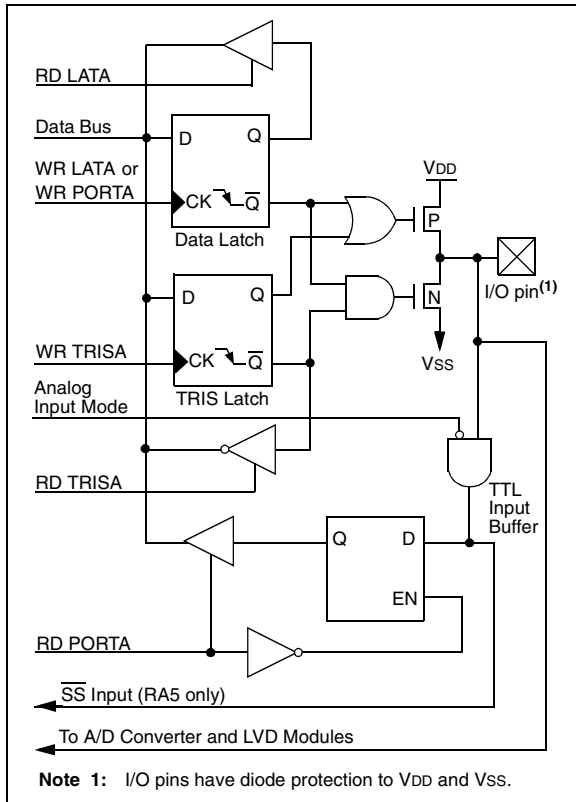


FIGURE 9-2: RA4/T0CKI PIN BLOCK DIAGRAM

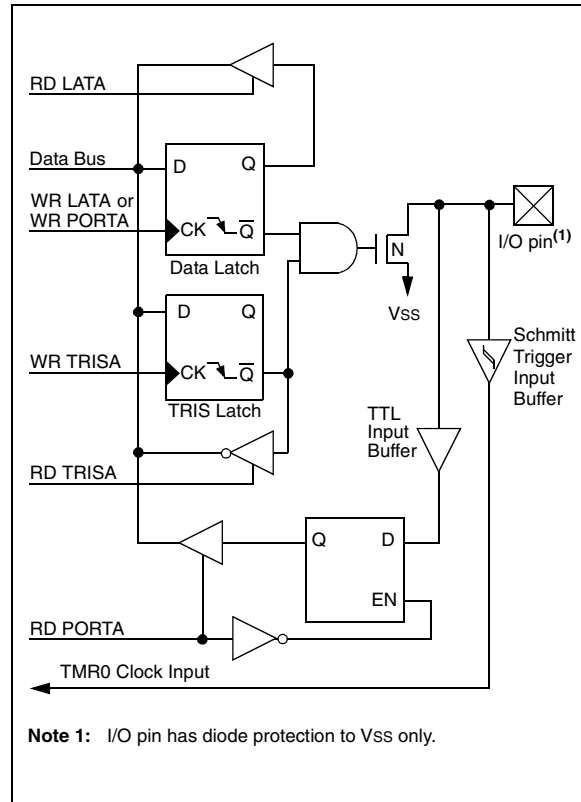
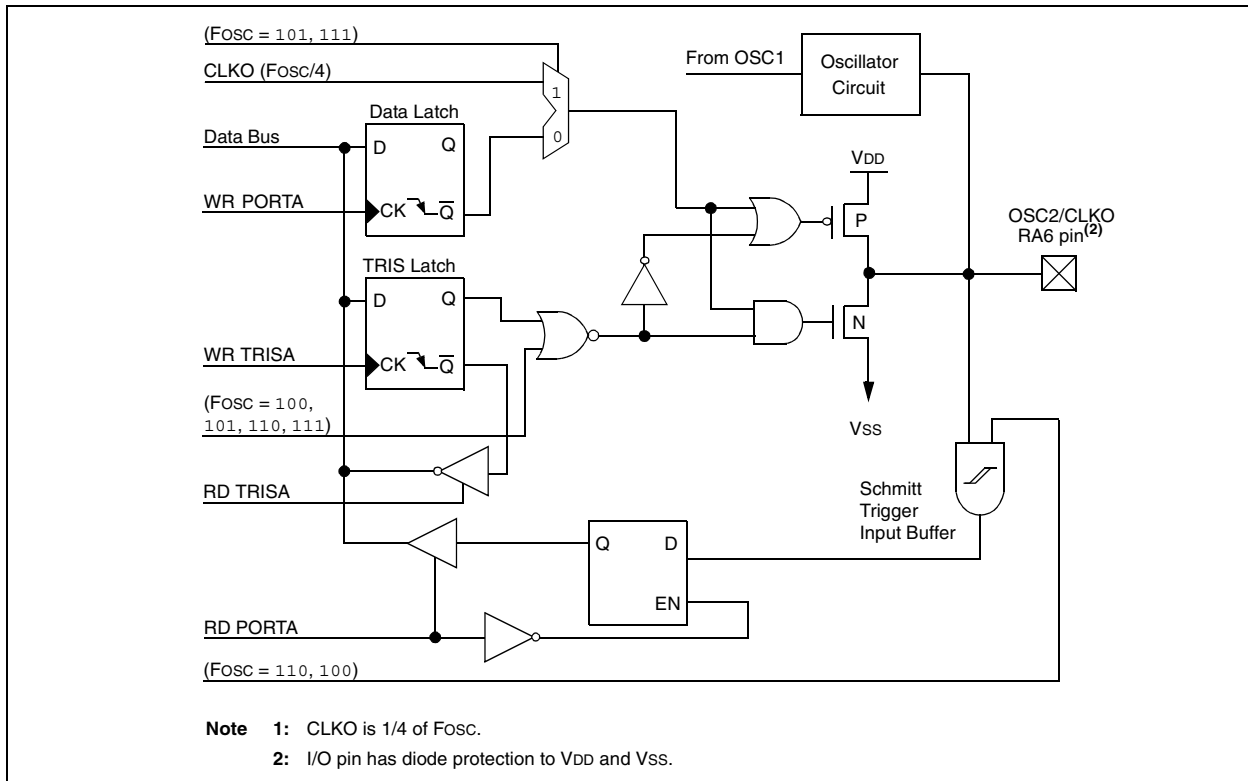


FIGURE 9-3: OSC2/CLKO/RA6 PIN BLOCK DIAGRAM



PIC18FXX8

15.3 Compare Mode

In Compare mode, the 16-bit CCPR1 and ECCPR1 register value is constantly compared against either the TMR1 register pair value or the TMR3 register pair value. When a match occurs, the CCP1 pin can have one of the following actions:

- Driven high
- Driven low
- Toggle output (high-to-low or low-to-high)
- Remains unchanged

The action on the pin is based on the value of control bits CCP1M3:CCP1M0. At the same time, interrupt flag bit CCP1IF is set.

15.3.1 CCP1 PIN CONFIGURATION

The user must configure the CCP1 pin as an output by clearing the appropriate TRISC bit.

Note: Clearing the CCP1CON register will force the CCP1 compare output latch to the default low level. This is not the data latch.

15.3.2 TIMER1/TIMER3 MODE SELECTION

Timer1 and/or Timer3 must be running in Timer mode, or Synchronized Counter mode, if the CCP module is using the compare feature. In Asynchronous Counter mode, the compare operation may not work.

15.3.3 SOFTWARE INTERRUPT MODE

When generate software interrupt is chosen, the CCP1 pin is not affected. Only a CCP interrupt is generated (if enabled).

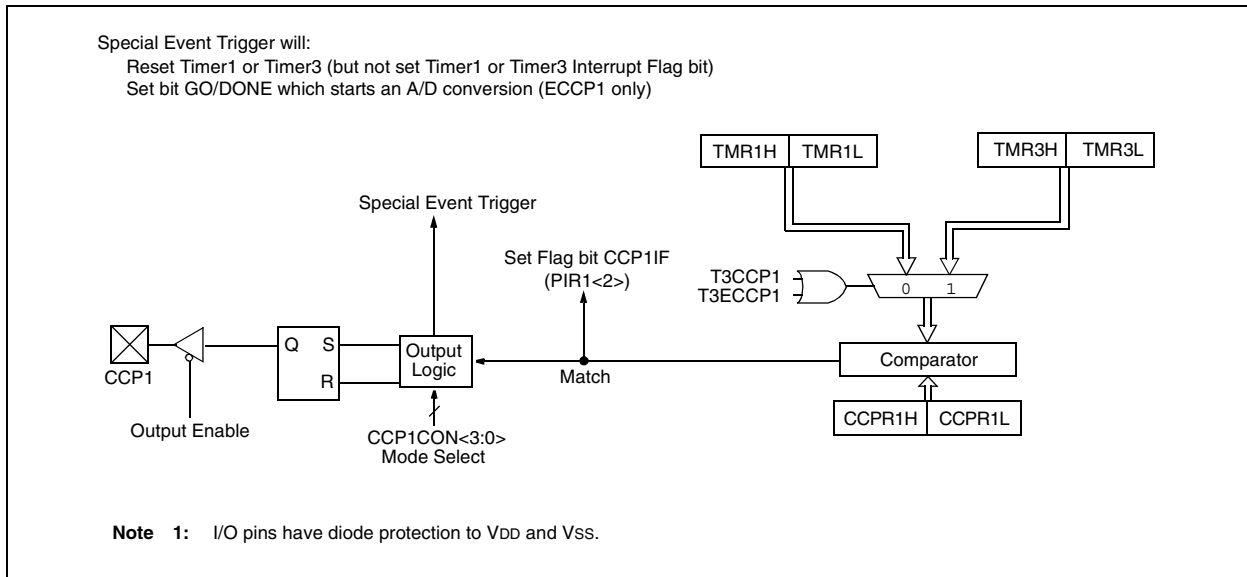
15.3.4 SPECIAL EVENT TRIGGER

In this mode, an internal hardware trigger is generated, which may be used to initiate an action.

The special event trigger output of CCP1 resets either the TMR1 or TMR3 register pair. Additionally, the ECCPR1 special event trigger will start an A/D conversion if the A/D module is enabled.

Note: The special event trigger from the ECCP1 module will not set the Timer1 or Timer3 interrupt flag bits.

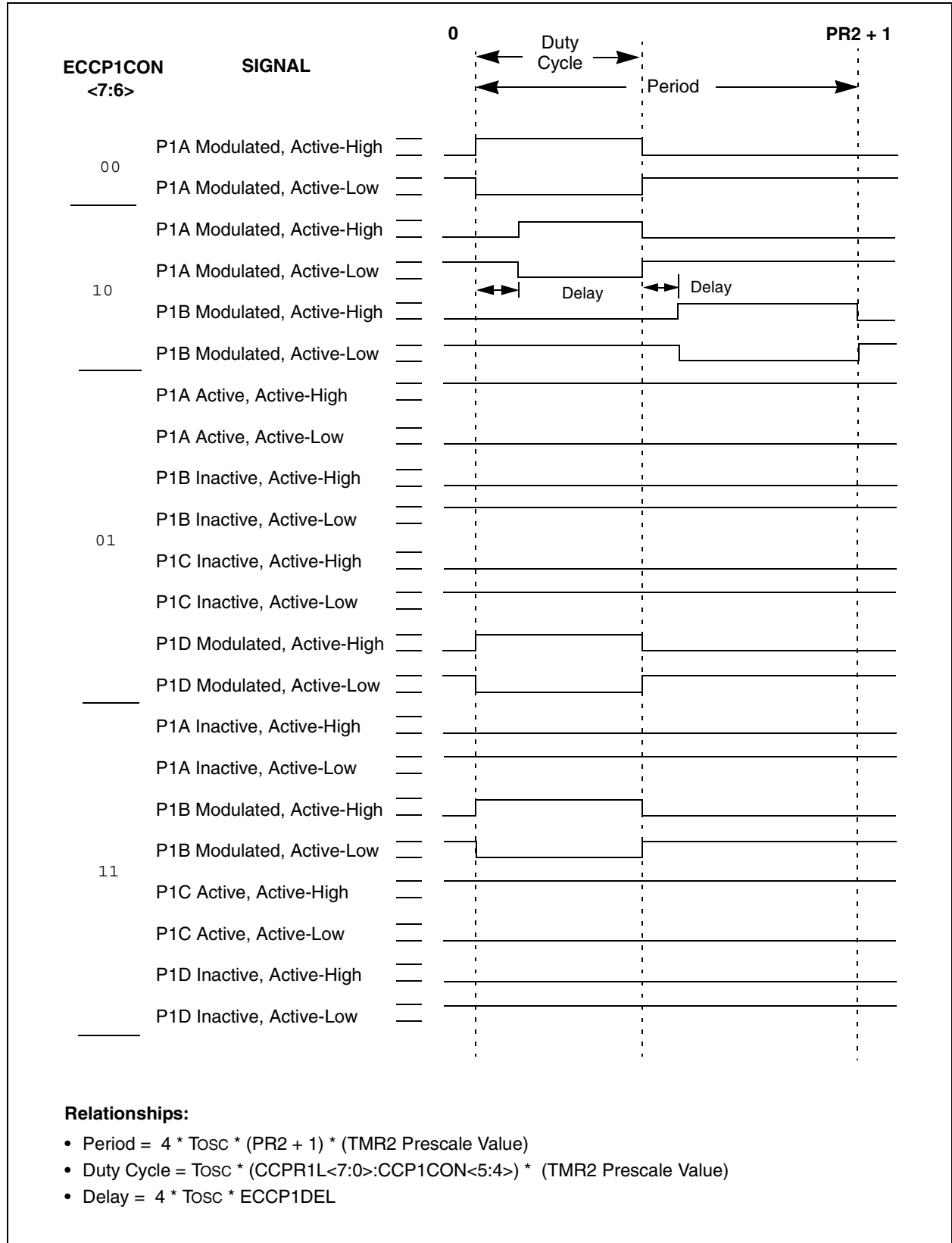
FIGURE 15-2: COMPARE MODE OPERATION BLOCK DIAGRAM



PIC18FXX8

NOTES:

FIGURE 16-2: PWM OUTPUT RELATIONSHIPS



PIC18FXX8

17.3.5 MASTER MODE

The master can initiate the data transfer at any time because it controls the SCK. The master determines when the slave (Processor 2, Figure 17-2) is to broadcast data by the software protocol.

In Master mode, the data is transmitted/received as soon as the SSPBUF register is written to. If the SPI is only going to receive, the SDO output could be disabled (programmed as an input). The SSPSR register will continue to shift in the signal present on the SDI pin at the programmed clock rate. As each byte is received, it will be loaded into the SSPBUF register as if a normal received byte (interrupts and status bits appropriately set). This could be useful in receiver applications as a "Line Activity Monitor" mode.

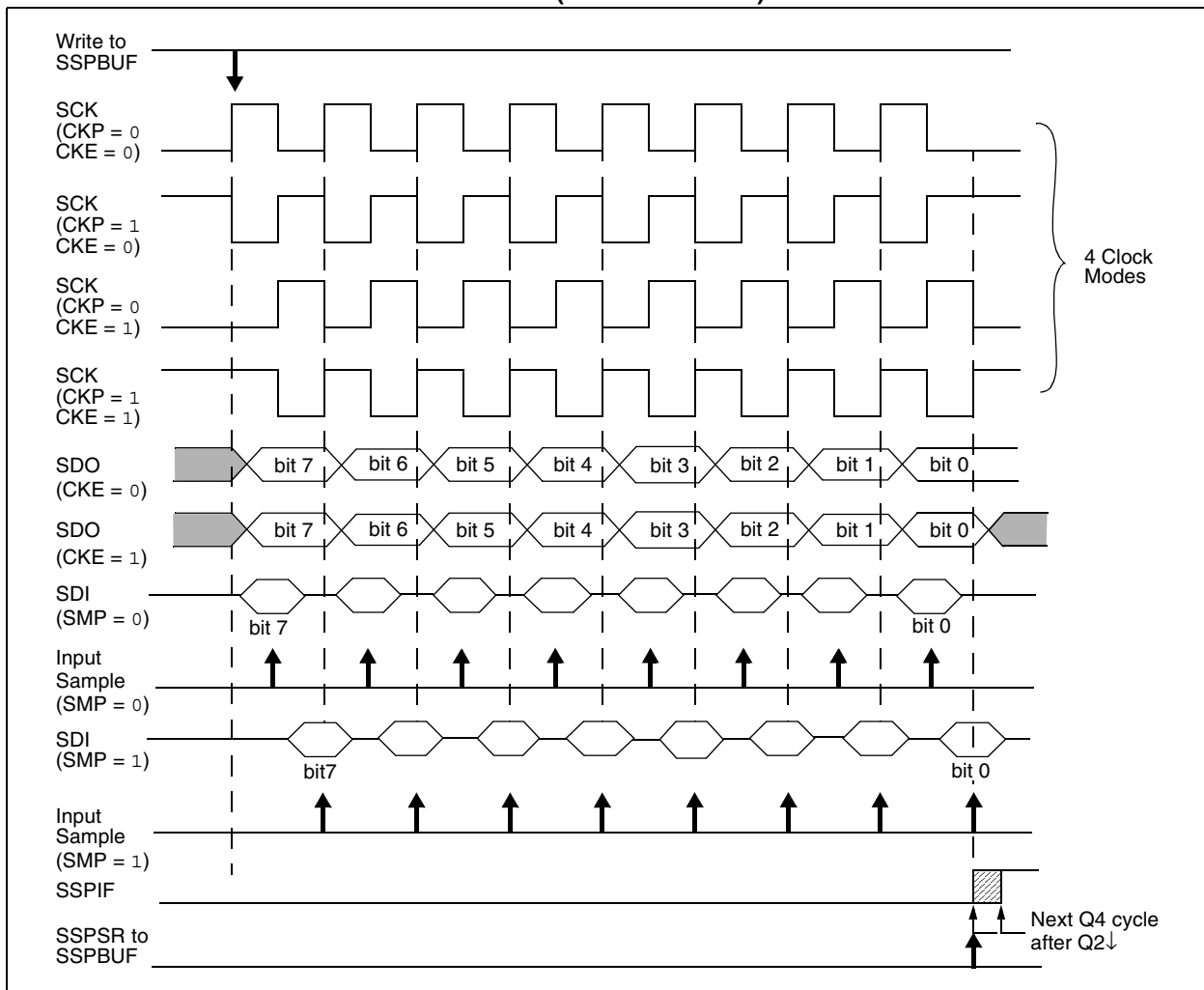
The clock polarity is selected by appropriately programming the CKP bit (SSPCON1<4>). This then, would give waveforms for SPI communication as shown in Figure 17-3, Figure 17-5 and Figure 17-6, where the MSB is transmitted first. In Master mode, the SPI clock rate (bit rate) is user programmable to be one of the following:

- $F_{OSC}/4$ (or T_{CY})
- $F_{OSC}/16$ (or $4 \cdot T_{CY}$)
- $F_{OSC}/64$ (or $16 \cdot T_{CY}$)
- $\text{Timer2 output}/2$

This allows a maximum data rate (at 40 MHz) of 10.00 Mbps.

Figure 17-3 shows the waveforms for Master mode. When the CKE bit is set, the SDO data is valid before there is a clock edge on SCK. The change of the input sample is shown based on the state of the SMP bit. The time when the SSPBUF is loaded with the received data is shown.

FIGURE 17-3: SPI™ MODE WAVEFORM (MASTER MODE)



PIC18FXX8

REGISTER 17-4: SSPCON1: MSSP CONTROL REGISTER 1 (I²C MODE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0

bit 7

bit 0

bit 7 **WCOL:** Write Collision Detect bit

In Master Transmit mode:

1 = A write to the SSPBUF register was attempted while the I²C conditions were not valid for a transmission to be started (must be cleared in software)

0 = No collision

In Slave Transmit mode:

1 = The SSPBUF register is written while it is still transmitting the previous word (must be cleared in software)

0 = No collision

In Receive mode (Master or Slave modes):

This is a “don’t care” bit.

bit 6 **SSPOV:** Receive Overflow Indicator bit

In Receive mode:

1 = A byte is received while the SSPBUF register is still holding the previous byte (must be cleared in software)

0 = No overflow

In Transmit mode:

This is a “don’t care” bit in Transmit mode.

bit 5 **SSPEN:** Synchronous Serial Port Enable bit

1 = Enables the serial port and configures the SDA and SCL pins as the serial port pins

0 = Disables serial port and configures these pins as I/O port pins

Note: When enabled, the SDA and SCL pins must be properly configured as input or output.

bit 4 **CKP:** SCK Release Control bit

In Slave mode:

1 = Release clock

0 = Holds clock low (clock stretch), used to ensure data setup time

In Master mode:

Unused in this mode.

bit 3-0 **SSPM3:SSPM0:** Synchronous Serial Port Mode Select bits

1111 = I²C Slave mode, 10-bit address with Start and Stop bit interrupts enabled

1110 = I²C Slave mode, 7-bit address with Start and Stop bit interrupts enabled

1011 = I²C Firmware Controlled Master mode (Slave Idle)

1000 = I²C Master mode, clock = FOSC/(4 * (SSPADD + 1))

0111 = I²C Slave mode, 10-bit address

0110 = I²C Slave mode, 7-bit address

Note: Bit combinations not specifically listed here are either reserved or implemented in SPI mode only.

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as ‘0’

-n = Value at POR

‘1’ = Bit is set

‘0’ = Bit is cleared

x = Bit is unknown

18.1 USART Baud Rate Generator (BRG)

The BRG supports both the Asynchronous and Synchronous modes of the USART. It is a dedicated 8-bit Baud Rate Generator. The SPBRG register controls the period of a free running, 8-bit timer. In Asynchronous mode, bit BRGH (TXSTA register) also controls the baud rate. In Synchronous mode, bit BRGH is ignored. Table 18-1 shows the formula for computation of the baud rate for different USART modes which only apply in Master mode (internal clock).

Given the desired baud rate and FOSC, the nearest integer value for the SPBRG register can be calculated using the formula in Table 18-1. From this, the error in baud rate can be determined.

Example 18-1 shows the calculation of the baud rate error for the following conditions:

FOSC = 16 MHz
Desired Baud Rate = 9600
BRGH = 0
SYNC = 0

It may be advantageous to use the high baud rate (BRGH = 1) even for slower baud clocks. This is because the $FOSC/(16(X + 1))$ equation can reduce the baud rate error in some cases.

Writing a new value to the SPBRG register causes the BRG timer to be reset (or cleared). This ensures the BRG does not wait for a timer overflow before outputting the new baud rate.

18.1.1 SAMPLING

The data on the RC7/RX/DT pin is sampled three times by a majority detect circuit to determine if a high or a low level is present at the RX pin.

EXAMPLE 18-1: CALCULATING BAUD RATE ERROR

Desired Baud Rate	=	$FOSC/(64 (X + 1))$
Solving for X:		
	X =	$((FOSC/Desired\ Baud\ Rate)/64) - 1$
	X =	$((16000000/9600)/64) - 1$
	X =	$[25.042] = 25$
Calculated Baud Rate	=	$16000000/(64 (25 + 1))$
	=	9615
Error	=	$\frac{(Calculated\ Baud\ Rate - Desired\ Baud\ Rate)}{Desired\ Baud\ Rate}$
	=	$(9615 - 9600)/9600$
	=	0.16%

TABLE 18-1: BAUD RATE FORMULA

SYNC	BRGH = 0 (Low Speed)	BRGH = 1 (High Speed)
0	(Asynchronous) Baud Rate = $FOSC/(64 (X + 1))$	Baud Rate = $FOSC/(16 (X + 1))$
1	(Synchronous) Baud Rate = $FOSC/(4 (X + 1))$	NA

Legend: X = value in SPBRG (0 to 255)

TABLE 18-2: REGISTERS ASSOCIATED WITH BAUD RATE GENERATOR

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000u
SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented, read as '0'. Shaded cells are not used by the BRG.

REGISTER 19-3: COMSTAT: COMMUNICATION STATUS REGISTER

R/C-0	R/C-0	R-0	R-0	R-0	R-0	R-0	R-0
RXB0OVFL	RXB1OVFL	TXBO	TXBP	RXBP	TXWARN	RXWARN	EWARN
bit 7							bit 0

- bit 7 **RXB0OVFL:** Receive Buffer 0 Overflow bit
 1 = Receive Buffer 0 overflowed
 0 = Receive Buffer 0 has not overflowed
- bit 6 **RXB1OVFL:** Receive Buffer 1 Overflow bit
 1 = Receive Buffer 1 overflowed
 0 = Receive Buffer 1 has not overflowed
- bit 5 **TXBO:** Transmitter Bus-Off bit
 1 = Transmit Error Counter > 255
 0 = Transmit Error Counter ≤ 255
- bit 4 **TXBP:** Transmitter Bus Passive bit
 1 = Transmission Error Counter > 127
 0 = Transmission Error Counter ≤ 127
- bit 3 **RXBP:** Receiver Bus Passive bit
 1 = Receive Error Counter > 127
 0 = Receive Error Counter ≤ 127
- bit 2 **TXWARN:** Transmitter Warning bit
 1 = 127 ≥ Transmit Error Counter > 95
 0 = Transmit Error Counter ≤ 95
- bit 1 **RXWARN:** Receiver Warning bit
 1 = 127 ≥ Receive Error Counter > 95
 0 = Receive Error Counter ≤ 95
- bit 0 **EWARN:** Error Warning bit
 This bit is a flag of the RXWARN and TXWARN bits.
 1 = The RXWARN or the TXWARN bits are set
 0 = Neither the RXWARN or the TXWARN bits are set

Legend:

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

19.8 Synchronization

To compensate for phase shifts between the oscillator frequencies of each of the nodes on the bus, each CAN controller must be able to synchronize to the relevant signal edge of the incoming signal. When an edge in the transmitted data is detected, the logic will compare the location of the edge to the expected time (Sync_Seg). The circuit will then adjust the values of Phase Segment 1 and Phase Segment 2, as necessary. There are two mechanisms used for synchronization.

19.8.1 HARD SYNCHRONIZATION

Hard synchronization is only done when there is a recessive to dominant edge during a bus Idle condition, indicating the start of a message. After hard synchronization, the bit time counters are restarted with Sync_Seg. Hard synchronization forces the edge which has occurred to lie within the synchronization segment of the restarted bit time. Due to the rules of synchronization, if a hard synchronization occurs, there will not be a resynchronization within that bit time.

19.8.2 RESYNCHRONIZATION

As a result of resynchronization, Phase Segment 1 may be lengthened or Phase Segment 2 may be shortened. The amount of lengthening or shortening of the phase buffer segments has an upper bound given by the Synchronization Jump Width (SJW). The value of the SJW will be added to Phase Segment 1 (see Figure 19-8) or subtracted from Phase Segment 2 (see Figure 19-9). The SJW is programmable between 1 T_Q and 4 T_Q.

Clocking information will only be derived from recessive to dominant transitions. The property, that only a fixed maximum number of successive bits have the same value, ensures resynchronization to the bit stream during a frame.

The phase error of an edge is given by the position of the edge relative to Sync_Seg, measured in T_Q. The phase error is defined in magnitude of T_Q as follows:

- $e = 0$ if the edge lies within Sync_Seg.
- $e > 0$ if the edge lies before the sample point.
- $e < 0$ if the edge lies after the sample point of the previous bit.

If the magnitude of the phase error is less than or equal to the programmed value of the synchronization jump width, the effect of a resynchronization is the same as that of a hard synchronization.

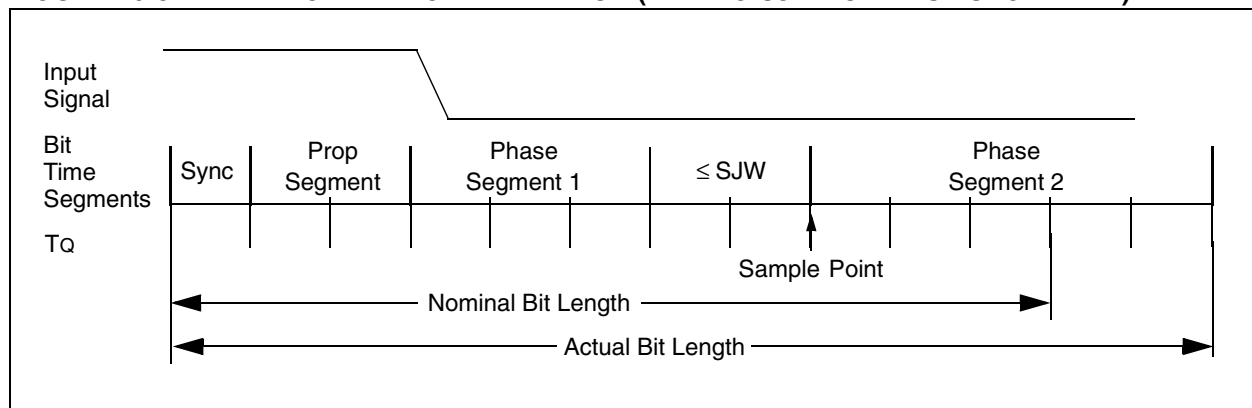
If the magnitude of the phase error is larger than the synchronization jump width and if the phase error is positive, then Phase Segment 1 is lengthened by an amount equal to the synchronization jump width.

If the magnitude of the phase error is larger than the resynchronization jump width and if the phase error is negative, then Phase Segment 2 is shortened by an amount equal to the synchronization jump width.

19.8.3 SYNCHRONIZATION RULES

- Only one synchronization within one bit time is allowed.
- An edge will be used for synchronization only if the value detected at the previous sample point (previously read bus value) differs from the bus value immediately after the edge.
- All other recessive to dominant edges, fulfilling rules 1 and 2, will be used for resynchronization with the exception that a node transmitting a dominant bit will not perform a resynchronization as a result of a recessive to dominant edge with a positive phase error.

FIGURE 19-8: LENGTHENING A BIT PERIOD (ADDING SJW TO PHASE SEGMENT 1)



To calculate the minimum acquisition time, Equation 20-1 may be used. This equation assumes that 1/2 LSB error is used (1024 steps for the A/D). The 1/2 LSB error is the maximum error allowed for the A/D to meet its specified resolution.

Example 20-1 shows the calculation of the minimum required acquisition time TACQ. This calculation is based on the following application system assumptions:

- CHOLD = 120 pF
- Rs = 2.5 kΩ
- Conversion Error ≤ 1/2 LSB
- VDD = 5V → Rss = 7 kΩ
- Temperature = 50°C (system max.)
- VHOLD = 0V @ time = 0

EQUATION 20-1: ACQUISITION TIME

$$\begin{aligned} \text{TACQ} &= \text{Amplifier Settling Time} + \text{Holding Capacitor Charging Time} + \text{Temperature Coefficient} \\ &= \text{TAMP} + \text{TC} + \text{Tcoff} \end{aligned}$$

EQUATION 20-2: A/D MINIMUM CHARGING TIME

$$\begin{aligned} \text{VHOLD} &= (\text{VREF} - (\text{VREF}/2048)) \cdot (1 - e^{-(\text{TC}/\text{CHOLD}(\text{RIC} + \text{RSS} + \text{RS})))} \\ \text{or} \\ \text{TC} &= -(120 \text{ pF})(1 \text{ k}\Omega + \text{RSS} + \text{RS}) \ln(1/2047) \end{aligned}$$

EXAMPLE 20-1: CALCULATING THE MINIMUM REQUIRED ACQUISITION TIME

$$\begin{aligned} \text{TACQ} &= \text{TAMP} + \text{TC} + \text{Tcoff} \\ \text{Temperature coefficient is only required for temperatures} &> 25^\circ\text{C}. \\ \text{TACQ} &= 2 \mu\text{s} + \text{TC} + [(\text{Temp} - 25^\circ\text{C})(0.05 \mu\text{s}/^\circ\text{C})] \\ \text{TC} &= -\text{CHOLD} (\text{RIC} + \text{RSS} + \text{RS}) \ln(1/2047) \\ &= -120 \text{ pF} (1 \text{ k}\Omega + 7 \text{ k}\Omega + 2.5 \text{ k}\Omega) \ln(0.0004885) \\ &= -120 \text{ pF} (10.5 \text{ k}\Omega) \ln(0.0004885) \\ &= -1.26 \mu\text{s} (-7.6241) \\ &= 9.61 \mu\text{s} \\ \text{TACQ} &= 2 \mu\text{s} + 9.61 \mu\text{s} + [(50^\circ\text{C} - 25^\circ\text{C})(0.05 \mu\text{s}/^\circ\text{C})] \\ &= 11.61 \mu\text{s} + 1.25 \mu\text{s} \\ &= 12.86 \mu\text{s} \end{aligned}$$

Note: When using external voltage references with the A/D converter, the source impedance of the external voltage references must be less than 20Ω to obtain the specified A/D resolution. Higher reference source impedances will increase both offset and gain errors. Resistive voltage dividers will not provide a sufficiently low source impedance.

To maintain the best possible performance in A/D conversions, external VREF inputs should be buffered with an operational amplifier or other low output impedance circuit.

PIC18FXX8

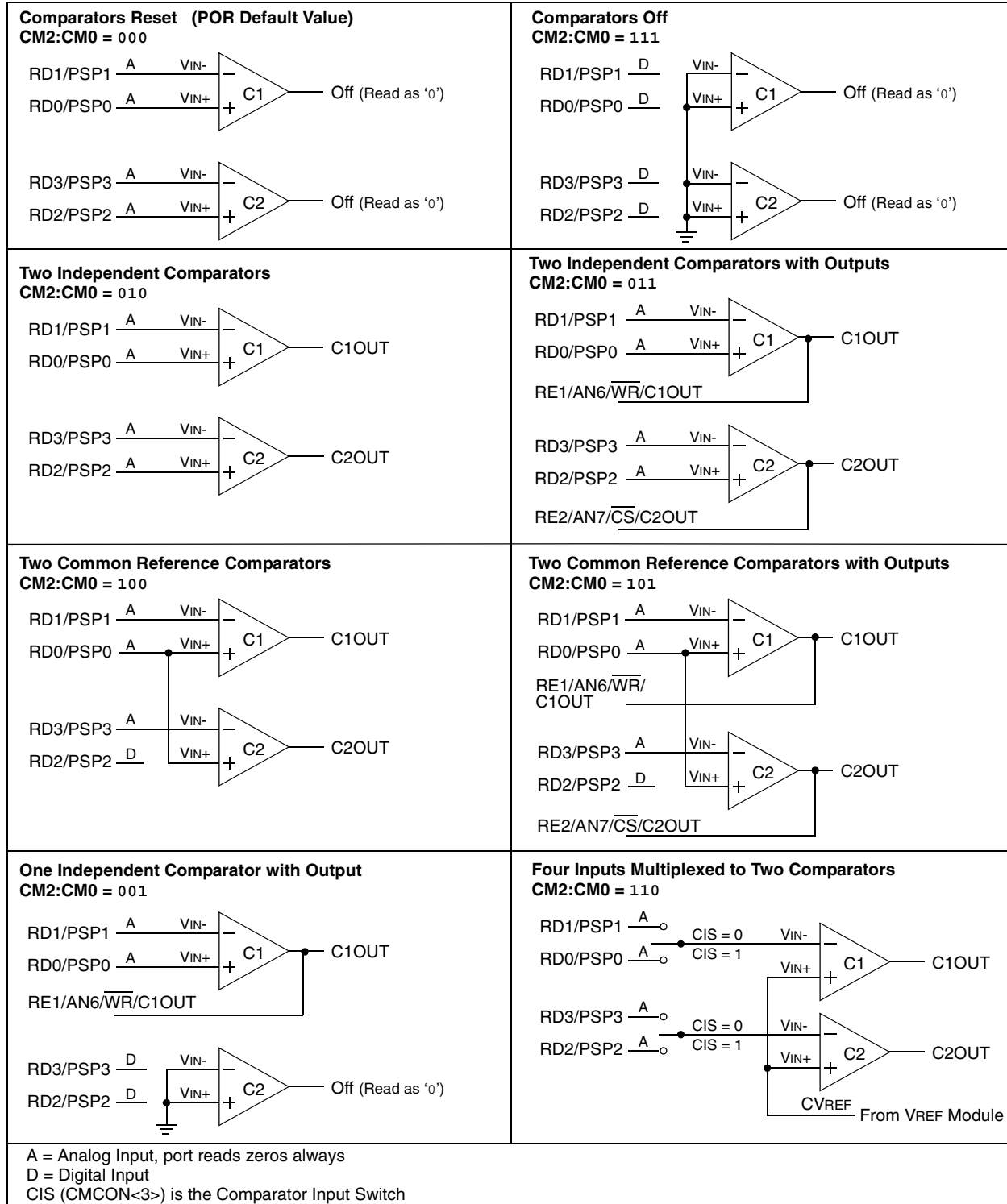
21.1 Comparator Configuration

There are eight modes of operation for the comparators. The CMCON register is used to select these modes. Figure 21-1 shows the eight possible modes. The TRISD register controls the data direction of the comparator pins for each mode. If the Comparator

mode is changed, the comparator output level may not be valid for the specified mode change delay shown in Section 27.0 “Electrical Characteristics”.

Note: Comparator interrupts should be disabled during a Comparator mode change; otherwise, a false interrupt may occur.

FIGURE 21-1: COMPARATOR I/O OPERATING MODES



21.7 Comparator Operation During Sleep

When a comparator is active and the device is placed in Sleep mode, the comparator remains active and the interrupt is functional if enabled. This interrupt will wake-up the device from Sleep mode when enabled. While the comparator is powered up, higher Sleep currents than shown in the power-down current specification will occur. Each operational comparator will consume additional current, as shown in the comparator specifications. To minimize power consumption while in Sleep mode, turn off the comparators, $CM<2:0> = 111$, before entering Sleep. If the device wakes up from Sleep, the contents of the CMCON register are not affected.

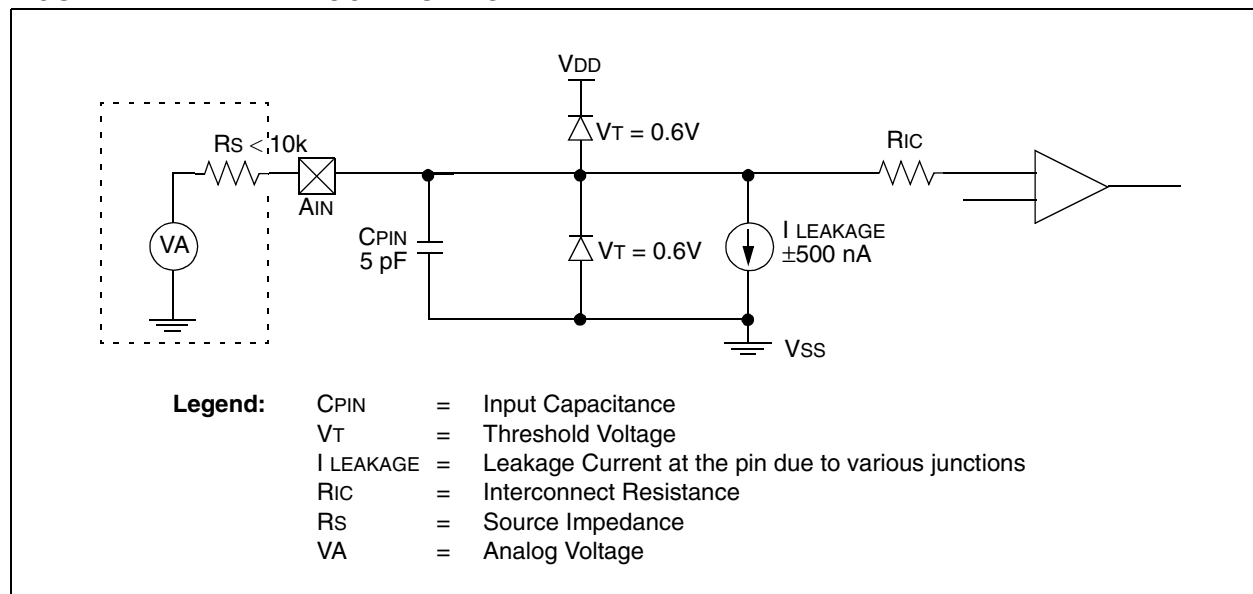
21.8 Effects of a Reset

A device Reset forces the CMCON register to its Reset state, causing the comparator module to be in the Comparator Reset mode, $CM<2:0> = 000$. This ensures that all potential inputs are analog inputs. Device current is minimized when analog inputs are present at Reset time. The comparators will be powered down during the Reset interval.

21.9 Analog Input Connection Considerations

A simplified circuit for an analog input is shown in Figure 21-4. Since the analog pins are connected to a digital output, they have reverse biased diodes to VDD and VSS. The analog input, therefore, must be between VSS and VDD. If the input voltage deviates from this range by more than 0.6V in either direction, one of the diodes is forward biased and a latch-up condition may occur. A maximum source impedance of 10 k Ω is recommended for the analog sources. Any external component connected to an analog input pin, such as a capacitor or a Zener diode, should have very little leakage current.

FIGURE 21-4: ANALOG INPUT MODEL



PIC18FXX8

REGISTER 24-5: CONFIG5L: CONFIGURATION REGISTER 5 LOW (BYTE ADDRESS 300008h)

U-0	U-0	U-0	U-0	R/C-1	R/C-1	R/C-1	R/C-1
—	—	—	—	CP3 ⁽¹⁾	CP2 ⁽¹⁾	CP1	CP0

bit 7 bit 0

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

bit 3 **CP3:** Code Protection bit⁽¹⁾

1 = Block 3 (006000-007FFFh) not code-protected

0 = Block 3 (006000-007FFFh) code-protected

bit 2 **CP2:** Code Protection bit⁽¹⁾

1 = Block 2 (004000-005FFFh) not code-protected

0 = Block 2 (004000-005FFFh) code-protected

bit 1 **CP1:** Code Protection bit

1 = Block 1 (002000-003FFFh) not code-protected

0 = Block 1 (002000-003FFFh) code-protected

bit 0 **CP0:** Code Protection bit

1 = Block 0 (000200-001FFFh) not code-protected

0 = Block 0 (000200-001FFFh) code-protected

Note 1: Unimplemented in PIC18FX48 devices; maintain this bit set.

Legend:

R = Readable bit

C = Clearable bit

U = Unimplemented bit, read as '0'

-n = Value when device is unprogrammed

u = Unchanged from programmed state

REGISTER 24-6: CONFIG5H: CONFIGURATION REGISTER 5 HIGH (BYTE ADDRESS 300009h)

R/C-1	R/C-1	U-0	U-0	U-0	U-0	U-0	U-0
CPD	CPB	—	—	—	—	—	—

bit 7 bit 0

bit 7 **CPD:** Data EEPROM Code Protection bit

1 = Data EEPROM not code-protected

0 = Data EEPROM code-protected

bit 6 **CPB:** Boot Block Code Protection bit

1 = Boot Block (000000-0001FFFh) not code-protected

0 = Boot Block (000000-0001FFFh) code-protected

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

Legend:

R = Readable bit

C = Clearable bit

U = Unimplemented bit, read as '0'

-n = Value when device is unprogrammed

u = Unchanged from programmed state

24.4.2 DATA EEPROM CODE PROTECTION

The entire data EEPROM is protected from external reads and writes by two bits: CPD and WRTD. CPD inhibits external reads and writes of data EEPROM. WRTD inhibits external writes to data EEPROM. The CPU can continue to read and write data EEPROM regardless of the protection bit settings.

24.4.3 CONFIGURATION REGISTER PROTECTION

The Configuration registers can be write-protected. The WRTC bit controls protection of the Configuration registers. In user mode, the WRTC bit is readable only. WRTC can only be written via ICSP or an external programmer.

24.5 ID Locations

Eight memory locations (200000h-200007h) are designated as ID locations where the user can store checksum or other code identification numbers. These locations are accessible during normal execution through the TBLRD and TBLWT instructions or during program/verify. The ID locations can be read when the device is code-protected.

24.6 In-Circuit Serial Programming

PIC18FXXX microcontrollers can be serially programmed while in the end application circuit. This is simply done with two lines for clock and data and three other lines for power, ground and the programming voltage. This allows customers to manufacture boards with unprogrammed devices and then program the microcontroller just before shipping the product. This also allows the most recent firmware or a custom firmware to be programmed.

24.7 In-Circuit Debugger

When the DEBUG bit in Configuration register, CONFIG4L, is programmed to a '0', the In-Circuit Debugger functionality is enabled. This function allows simple debugging functions when used with MPLAB® IDE. When the microcontroller has this feature enabled, some of the resources are not available for general use. Resources used include 2 I/O pins, stack locations, program memory and data memory. For more information on the resources required, see the User's Guide for the In-Circuit Debugger you are using.

To use the In-Circuit Debugger function of the microcontroller, the design must implement In-Circuit Serial Programming connections to MCLR/VPP, VDD, GND, RB7 and RB6. This will interface to the In-Circuit Debugger module available from Microchip or one of the third party development tool companies. The Microchip In-Circuit Debugger (ICD) used with the PIC18FXXX microcontrollers is the MPLAB® ICD 2.

24.8 Low-Voltage ICSP Programming

The LVP bit in Configuration register, CONFIG4L, enables Low-Voltage ICSP Programming. This mode allows the microcontroller to be programmed via ICSP using a VDD source in the operating voltage range. This only means that VPP does not have to be brought to VIH but can instead be left at the normal operating voltage. In this mode, the RB5/PGM pin is dedicated to the programming function and ceases to be a general purpose I/O pin. During programming, VDD is applied to the MCLR/VPP pin. To enter Programming mode, VDD must be applied to the RB5/PGM pin, provided the LVP bit is set. The LVP bit defaults to a ('1') from the factory.

Note 1: The High-Voltage Programming mode is always available, regardless of the state of the LVP bit, by applying VIH to the MCLR pin.

2: While in Low-Voltage ICSP mode, the RB5 pin can no longer be used as a general purpose I/O pin.

3: When using Low-Voltage ICSP Programming (LVP) and the pull-ups on PORTB are enabled, bit 5 in the TRISB register must be cleared to disable the pull-up on RB5 and ensure the proper operation of the device.

If Low-Voltage Programming mode is not used, the LVP bit can be programmed to a '0' and RB5/PGM becomes a digital I/O pin. However, the LVP bit may only be programmed when programming is entered with VIH on MCLR/VPP. The LVP bit can only be changed when using high voltage on MCLR.

It should be noted that once the LVP bit is programmed to '0', only the High-Voltage Programming mode is available and only High-Voltage Programming mode can be used to program the device.

When using Low-Voltage ICSP Programming, the part must be supplied 4.5V to 5.5V if a bulk erase will be executed. This includes reprogramming of the code-protect bits from an ON state to an OFF state. For all other cases of Low-Voltage ICSP Programming, the part may be programmed at the normal operating voltage. This means unique user IDs or user code can be reprogrammed or added.

BTG Bit Toggle f

Syntax:	[<i>label</i>] BTG f,b[,a]			
Operands:	$0 \leq f \leq 255$ $0 \leq b \leq 7$ $a \in [0,1]$			
Operation:	$\overline{(f \ll b)} \rightarrow f \ll b$			
Status Affected:	None			
Encoding:	0111	bbba	ffff	ffff
Description:	Bit 'b' in data memory location 'f' is inverted. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).			
Words:	1			
Cycles:	1			
Q Cycle Activity:				
	Q1	Q2	Q3	Q4
	Decode	Read register 'f'	Process Data	Write register 'f'

Example: BTG PORTC, 4

Before Instruction:
PORTC = 0111 0101 [0x75]

After Instruction:
PORTC = 0110 0101 [0x65]

BOV Branch if Overflow

Syntax:	[<i>label</i>] BOV n			
Operands:	$-128 \leq n \leq 127$			
Operation:	if Overflow bit is '1' $(PC) + 2 + 2n \rightarrow PC$			
Status Affected:	None			
Encoding:	1110	0100	nnnn	nnnn
Description:	If the Overflow bit is '1', 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 two-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 BOV JUMP

Before Instruction
PC = address (HERE)

After Instruction
If Overflow = 1;
PC = address (JUMP)
If Overflow = 0;
PC = address (HERE + 2)

PIC18FXX8

IORLW Inclusive OR Literal with W

Syntax: [*label*] IORLW *k*

Operands: $0 \leq k \leq 255$

Operation: (W) .OR. *k* \rightarrow W

Status Affected: N, Z

Encoding:

0000	1001	kkkk	kkkk
------	------	------	------

Description: The contents of W are ORed with 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: IORLW 0x35

Before Instruction
W = 0x9A

After Instruction
W = 0xBF

IORWF Inclusive OR W with f

Syntax: [*label*] IORWF *f* [,d [,a]]

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

Operation: (W) .OR. (*f*) \rightarrow dest

Status Affected: N, Z

Encoding:

0001	00da	ffff	ffff
------	------	------	------

Description: Inclusive OR W with register 'f'. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default). If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: IORWF RESULT, W

Before Instruction
RESULT = 0x13
W = 0x91

After Instruction
RESULT = 0x13
W = 0x93

26.0 DEVELOPMENT SUPPORT

The PICmicro® microcontrollers are supported with a full range of hardware and software development tools:

- Integrated Development Environment
 - MPLAB® IDE Software
- Assemblers/Compilers/Linkers
 - MPASM™ Assembler
 - MPLAB C17 and MPLAB C18 C Compilers
 - MPLINK™ Object Linker/
MPLIB™ Object Librarian
 - MPLAB C30 C Compiler
 - MPLAB ASM30 Assembler/Linker/Library
- Simulators
 - MPLAB SIM Software Simulator
 - MPLAB dsPIC30 Software Simulator
- Emulators
 - MPLAB ICE 2000 In-Circuit Emulator
 - MPLAB ICE 4000 In-Circuit Emulator
- In-Circuit Debugger
 - MPLAB ICD 2
- Device Programmers
 - PRO MATE® II Universal Device Programmer
 - PICSTART® Plus Development Programmer
 - MPLAB PM3 Device Programmer
- Low-Cost Demonstration Boards
 - PICDEM™ 1 Demonstration Board
 - PICDEM.net™ Demonstration Board
 - PICDEM 2 Plus Demonstration Board
 - PICDEM 3 Demonstration Board
 - PICDEM 4 Demonstration Board
 - PICDEM 17 Demonstration Board
 - PICDEM 18R Demonstration Board
 - PICDEM LIN Demonstration Board
 - PICDEM USB Demonstration Board
- Evaluation Kits
 - KEELOQ® Evaluation and Programming Tools
 - PICDEM MSC
 - microID® Developer Kits
 - CAN
 - PowerSmart® Developer Kits
 - Analog

26.1 MPLAB Integrated Development Environment Software

The MPLAB IDE software brings an ease of software development previously unseen in the 8/16-bit microcontroller market. The MPLAB IDE is a Windows® based application that contains:

- An interface to debugging tools
 - simulator
 - programmer (sold separately)
 - emulator (sold separately)
 - in-circuit debugger (sold separately)
- A full-featured editor with color coded context
- A multiple project manager
- Customizable data windows with direct edit of contents
- High-level source code debugging
- Mouse over variable inspection
- Extensive on-line help

The MPLAB IDE allows you to:

- Edit your source files (either assembly or C)
- One touch assemble (or compile) and download to PICmicro emulator and simulator tools (automatically updates all project information)
- Debug using:
 - source files (assembly or C)
 - mixed assembly and C
 - machine code

MPLAB IDE supports multiple debugging tools in a single development paradigm, from the cost effective simulators, through low-cost in-circuit debuggers, to full-featured emulators. This eliminates the learning curve when upgrading to tools with increasing flexibility and power.

26.2 MPASM Assembler

The MPASM assembler is a full-featured, universal macro assembler for all PICmicro 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