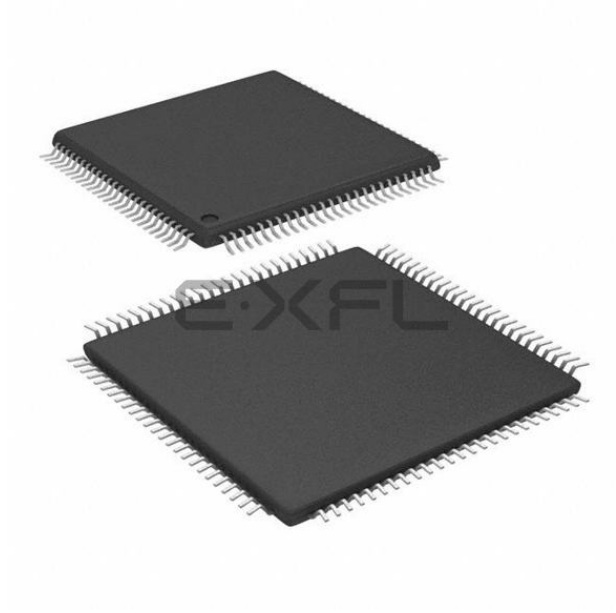


Welcome to E-XFL.COM

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, IrDA, LINbus, SPI, UART/USART, USB
Peripherals	Brown-out Detect/Reset, HLVD, LCD, POR, PWM, WDT
Number of I/O	86
Program Memory Size	64KB (32K x 16)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	4K x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 3.6V
Data Converters	A/D 24x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	100-TQFP
Supplier Device Package	100-TQFP (12x12)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f96j94-i-pt

1.0 DEVICE OVERVIEW

This document contains device-specific information for the following devices:

- PIC18F97J94
- PIC18F87J94
- PIC18F67J94
- PIC18F96J94
- PIC18F86J94
- PIC18F66J94
- PIC18F95J94
- PIC18F85J94
- PIC18F65J94

This family introduces a new line of low-voltage LCD microcontrollers with Universal Serial Bus (USB). It combines all the main traditional advantage of all PIC18 microcontrollers, namely, high computational performance and a rich feature set at an extremely competitive price point. These features make the PIC18F9XJ94 family a logical choice for many high-performance applications, where cost is a primary consideration.

1.1 Core Features

1.1.1 TECHNOLOGY

All of the devices in the PIC18F9XJ94 family incorporate a range of features that can significantly reduce power consumption during operation. Key items include:

- **Alternate Run Modes:** By clocking the controller from the Timer1 source or the Internal RC oscillator, power consumption during code execution can be reduced.
- **Multiple Idle Modes:** The controller can also run with its CPU core disabled but the peripherals still active. In these states, power consumption can be reduced even further.
- **On-the-Fly Mode Switching:** The power-managed modes are invoked by user code during operation, allowing the user to incorporate power-saving ideas into their application's software design.
- **XLP:** An extra low-power Sleep, BOR, RTCC and Watchdog Timer.

1.1.2 OSCILLATOR OPTIONS AND FEATURES

All of the devices in the PIC18F9XJ94 family offer different oscillator options, allowing users a range of choices in developing application hardware. These include:

- Two Crystal modes (HS, MS)
- One External Clock mode (EC)
- A Phase Lock Loop (PLL) frequency multiplier, which allows clock speeds of up to 64 MHz.
- A fast Internal Oscillator (FRC) block that provides an 8 MHz clock (±0.15% accuracy) with Active Clock Tuning (ACT) from USB or SOSC source.
  - Offers multiple divider options from 8 MHz to 500 kHz
  - Frees the two oscillator pins for use as additional general purpose I/O
- A separate Low-Power Internal RC Oscillator (LPRC) (31 kHz nominal) for low-power, timing-insensitive applications.

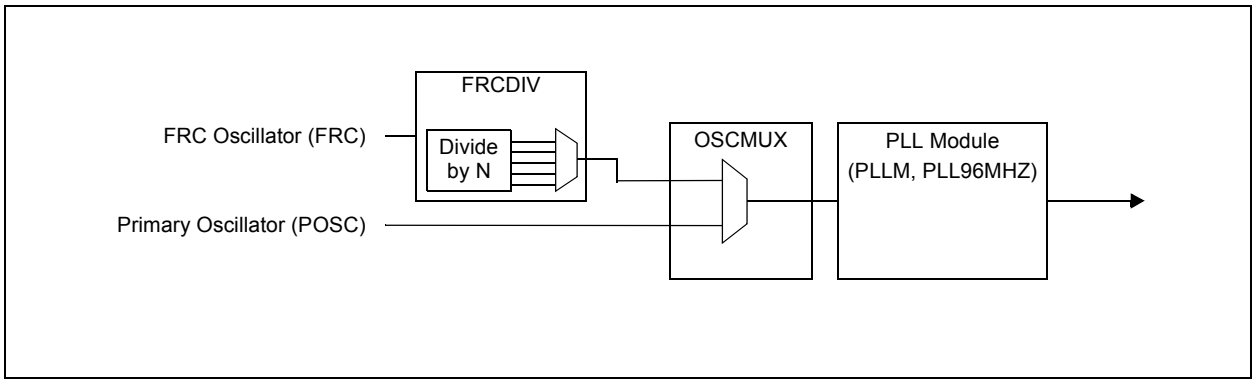
The internal oscillator block provides a stable reference source that gives the family additional features for robust operation:

- **Fail-Safe Clock Monitor (FSCM):** This option constantly monitors the main clock source against a reference signal provided by the internal oscillator. If a clock failure occurs, the controller is switched to the internal oscillator, allowing for continued low-speed operation or a safe application shutdown.
- **Two-Speed Start-up (IESO):** This option allows the internal oscillator to serve as the clock source from Power-on Reset, or wake-up from Sleep mode, until the primary clock source is available.

3.8 Phase Lock Loop (PLL) Branch

The PLL module contains two separate PLL submodules: PLLM and PLL96MHZ. The PLLM submodule is configurable as a 4x, 6x or 8x PLL. The PLL96MHZ submodule runs at 96 MHz and requires an input clock between 4 MHz and 48 MHz (a multiple of 4 MHz). These are selected through the PLLDIV<3:0> bits.

FIGURE 3-7: BASIC OSCILLATOR BLOCK DIAGRAM



4.4.10 CONTROL BIT SUMMARY FOR SLEEP MODES

Table 4-5 shows the settings for the bits relevant to Deep Sleep modes.

TABLE 4-5: BIT SETTINGS FOR ALL DEEP SLEEP MODES

Instruction-Based Mode	DSEN (DSCONH<7>)	Retention Regulator			DSWDTEN (CONFIG8H<0>)
		$\overline{\text{RETEN}}$ (CONFIG7L<0>)	SRETEN (RCON4<4>)	State	
Retention Deep Sleep	1	0	1	Enabled	0
Deep Sleep	1	1	x	Disabled	x

4.4.11 WAKE-UP DELAYS

The Reset delays associated with wake-up from Deep Sleep and Retention Deep Sleep modes, in different oscillator modes, are provided in Table 4-6 and Table 4-7, respectively.

Note: The PMSLP bit (RCON4<0>) allows the voltage regulator to be maintained during Sleep modes.

TABLE 4-6: DELAY TIMES FOR EXITING FROM DEEP SLEEP MODE

Clock Source		Exit Delay	Oscillator Delay	Notes
EC		TDSWU	—	
ECPLL		TDSWU	TLOCK	**1, 3**
MS, HS		TDSWU	TOST	**1, 2**
MSPLL, HSPLL		TDSWU	TOST + TLOCK	**1, 2, 3**
SOSC	(Off during Sleep)	TDSWU	TOST	**1, 2**
	(On during Sleep)	TDSWU	—	**1**
FRC, FRCDIV		TDSWU	TFRC	**1, 4**
LPRC	(Off during Sleep)	TDSWU	TLPRC	**1, 4**
	(On during Sleep)	TDSWU	—	**1**
FRCPLL		TDSWU	TFRC + TLOCK	**1, 3, 4**

Note 1: TDSWU = Deep Sleep wake-up delay.

2: TOST = Oscillator Start-up Timer; a delay of 1024 oscillator periods before the oscillator clock is released to the system.

3: TLOCK = PLL lock time.

4: TFRC and TLPRC are RC Oscillator start-up times.

TABLE 5-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices			Power-on Reset, Brown-out Reset	MCLR Resets, WDT Reset, RESET Instruction, Stack Resets	Wake-up via WDT or Interrupt
LCDDATA6	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA5	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA4	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA3	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA2	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA1	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA0	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
ADCON2H	64-pin	80-pin	100-pin	0000 00--	0000 00--	uuuu uu--
ADCON2L	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCON3H	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCON3L	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCON5H	64-pin	80-pin	100-pin	000- --00	000- --00	uuu- --uu
ADCON5L	64-pin	80-pin	100-pin	---- 0000	---- 0000	---- uuuu
ADCHS0H	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCHS0L	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCSS1H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCSS1L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCSS0H	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCSS0L	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCHIT1H	64-pin	80-pin	100-pin	---- --00	---- --00	---- --uu
ADCHIT1L	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCHIT0H	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCHIT0L	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCTMUEN1H	64-pin	80-pin	100-pin	-000 0000	-000 0000	uuuu uuuu
ADCTMUEN1L	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCTMUEN0H	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCTMUEN0L	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCBUF25H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF25L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF24H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF24L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF23H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF23L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu

Legend: u = unchanged; x = unknown; - = unimplemented bit, read as '0'; q = value depends on condition. Shaded cells indicate that conditions do not apply for the designated device.

Note 1: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

4: See Table 5-2 for Reset value for specific condition.

5: Bits 7,6 are unimplemented on 64 and 80-pin devices.

6: If the VBAT is always powered, the DSGPx register values will remain unchanged after the first POR.

TABLE 6-2: REGISTER FILE SUMMARY

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
FFFh	TOSU	—	—	—	Top-of-Stack Upper Byte (TOS<20:16>)				
FFEh	TOSH	Top-of-Stack High Byte (TOS<15:8>)							
FFDh	TOSL	Top-of-Stack Low Byte (TOS<7:0>)							
FFCh	STKPTR	STKFUL	STKUNF	—	STKPTR				
FFBh	PCLATU	—	—	—	Holding Register for PC<20:16>				
FFAh	PCLATH	Holding Register for PC<15:8>							
FF9h	PCL	PC Low Byte (PC<7:0>)							
FF8h	TBLPTRU	—	—	ACSS	Program Memory Table Pointer Upper Byte (TBLPTR<20:16>)				
FF7h	TBLPTRH	Program Memory Table Pointer High Byte (TBLPTR<15:8>)							
FF6h	TBLPTRL	Program Memory Table Pointer Low Byte (TBLPTR<7:0>)							
FF5h	TABLAT	Program Memory Table Latch							
FF4h	PRODH	Product Register High Byte							
FF3h	PRODL	Product Register Low Byte							
FF2h	INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	IOCIE	TMR0IF	INT0IF	IOCIF
FF1h	INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	IOCIP
FF0h	INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF
FEFh	INDF0	Uses contents of FSR0 to address data memory – value of FSR0 not changed (not a physical register)							
FEEh	POSTINC0	Uses contents of FSR0 to address data memory – value of FSR0 post-incremented (not a physical register)							
FEDh	POSTDEC0	Uses contents of FSR0 to address data memory – value of FSR0 post-decremented (not a physical register)							
FECh	PREINC0	Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented (not a physical register)							
FEBh	PLUSW0	Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented (not a physical register) – value of FSR0 offset by W							
FEAh	FSR0H	—	—	—	—	Indirect Data Memory Address Pointer 0 High			
FE9h	FSR0L	Indirect Data Memory Address Pointer 0 Low Byte							
FE8h	WREG	Working Register							
FE7h	INDF1	Uses contents of FSR1 to address data memory – value of FSR1 not changed (not a physical register)							
FE6h	POSTINC1	Uses contents of FSR1 to address data memory – value of FSR1 post-incremented (not a physical register)							
FE5h	POSTDEC1	Uses contents of FSR1 to address data memory – value of FSR1 post-decremented (not a physical register)							
FE4h	PREINC1	Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented (not a physical register)							
FE3h	PLUSW1	Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented (not a physical register) – value of FSR1 offset by W							
FE2h	FSR1H	—	—	—	—	Indirect Data Memory Address Pointer 1 High			
FE1h	FSR1L	Indirect Data Memory Address Pointer 1 Low Byte							
FE0h	BSR	—	—	—	—	Bank Select Register			
FDFh	INDF2	Uses contents of FSR2 to address data memory – value of FSR2 not changed (not a physical register)							
FDEh	POSTINC2	Uses contents of FSR2 to address data memory – value of FSR2 post-incremented (not a physical register)							
FDDh	POSTDEC2	Uses contents of FSR2 to address data memory – value of FSR2 post-decremented (not a physical register)							
FDCh	PREINC2	Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented (not a physical register)							
FDBh	PLUSW2	Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented (not a physical register) – value of FSR2 offset by W							
FDAh	FSR2H	—	—	—	—	Indirect Data Memory Address Pointer 2 High			
FD9h	FSR2L	Indirect Data Memory Address Pointer 2 Low Byte							
FD8h	STATUS	—	—	—	N	OV	Z	DC	C
FD7h	TMR0H	Timer0 Register High Byte							
FD6h	TMR0L	Timer0 Register Low Byte							
FD5h	T0CON	TMR0ON	T08BIT	T0CS1	T0CS0	PSA	T0PS2	T0PS1	T0PS0
FD4h	Unimplemented	—	—	—	—	—	—	—	—
FD3h	OSCCON	IDLEN	COSC2	COSC1	COSC0	—	NOSC2	NOSC1	NOSC0
FD2h	IPR5	—	ACTORSIP	ACTLOCKIP	TMR8IP	—	TMR6IP	TMR5IP	TMR4IP
FD1h	IOCF	IOCF7	IOCF6	IOCF5	IOCF4	IOCF3	IOCF2	IOCF1	IOCF0
FD0h	RCON	IPEN	—	$\overline{\text{CM}}$	$\overline{\text{RI}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$

Legend: — = unimplemented, read as '0'.

11.1.1 OUTPUT PIN DRIVE

When used as digital I/O, the output pin drive strengths vary, according to the pins' grouping, to meet the needs for a variety of applications. In general, there are two classes of output pins in terms of drive capability:

- Outputs designed to drive higher current loads, such as LEDs:
  - PORTB
  - PORTC
- Outputs with lower drive levels, but capable of driving normal digital circuit loads with a high input impedance. Able to drive LEDs, but only those with smaller current requirements:
  - PORTA
  - PORTD
  - PORTE
  - PORTF
  - PORTG
  - PORTH(1)
  - PORTJ(1)
  - PORTK(2)
  - PORTL(2)

Note 1: These ports are not available on 64-pin devices.

2: These ports are not available on 64-pin or 80-pin devices.

11.1.2 PULL-UP CONFIGURATION

Nine of the I/O ports (all ports except PORTA and PORTC) implement configurable weak pull-ups on all pins. These are internal pull-ups that allow floating digital input signals to be pulled to a consistent level without the use of external resistors.

Pull-ups for PORTB are enabled by clearing the $\overline{\text{RBPU}}$ bit (INTCON2<7>). PORTB pull-ups are individually selectable through the WPUB register.

Pull-ups for PORTD, PORTE, PORTF, PORTG, PORTH, PORTJ, PORTK and PORTL are enabled through their corresponding enable bits in the PADCFG register, but are not pin-selectable.

11.10 PORTJ, LATJ and TRISJ Registers

Note: PORTJ is available only on 80-pin and 100-pin devices.

PORTJ is an 8-bit wide, bidirectional port. The corresponding Data Direction and Output Latch registers are TRISJ and LATJ.

All pins on PORTJ are implemented with Schmitt Trigger input buffers. Each pin is individually configurable as an input or output.

Note: These pins are configured as digital inputs on any device Reset.

When the external memory interface is enabled, all of the PORTJ pins function as control outputs for the interface. This occurs automatically when the interface is enabled by clearing the EBDIS control bit (MEMCON<7>). The TRISJ bits are also overridden.

Each of the PORTJ pins has a weak internal pull-up. The pull-ups are provided to keep the inputs at a known state for the external memory interface while powering up. A single control bit can turn off all the pull-ups. This is performed by clearing bit, RJPU (PADCFG<2>). The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on any device Reset.

EXAMPLE 11-9: INITIALIZING PORTJ

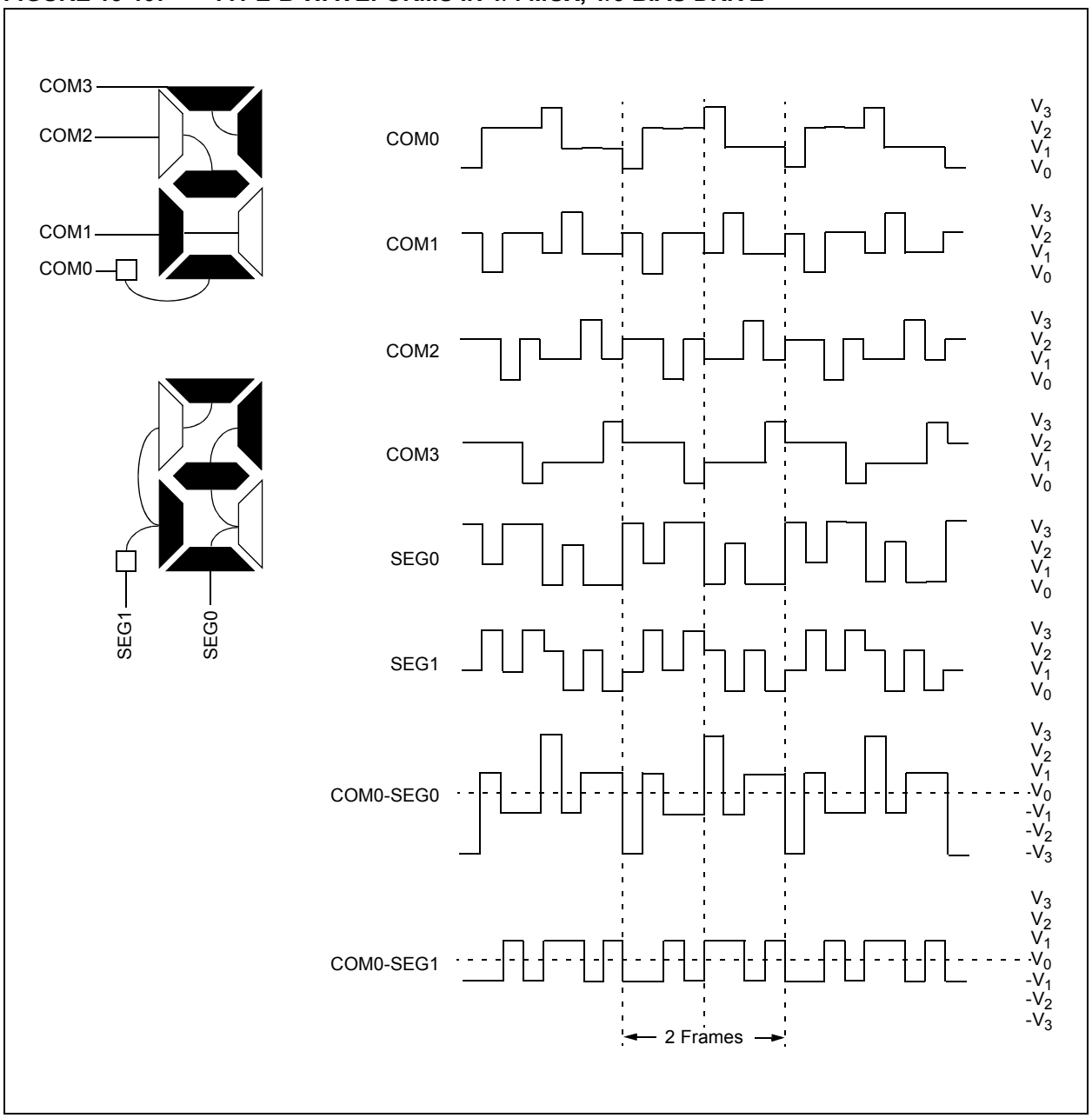
```
CLRF    PORTJ   ; Initialize PORTJ by
                ; clearing output latches
CLRF    LATJ    ; Alternate method
                ; to clear output latches
MOVLW   0CFh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISJ   ; Set RJ3:RJ0 as inputs
                ; RJ5:RJ4 as output
                ; RJ7:RJ6 as inputs
```

TABLE 11-9: PORTJ FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RJ0/SEG32/ ALE	RJ0	0	O	DIG	LATJ<0> data output.
		1	I	ST	PORTJ<0> data input.
	SEG32	0	O	ANA	LCD Segment 32 output; disables all other pin functions.
	ALE	x	O	DIG	External Memory Bus Address Latch Enable (ALE) signal.
RJ1/SEG33/$\overline{OE}$	RJ1	0	O	DIG	LATJ<1> data output.
		1	I	ST	PORTJ<1> data input.
	SEG33	0	O	ANA	LCD Segment 33 output; disables all other pin functions.
	$\overline{OE}$	x	O	DIG	External Memory Bus Address Latch Enable ($\overline{OE}$) signal.
RJ2/SEG34/ $\overline{WRL}$	RJ2	0	O	DIG	LATJ<2> data output.
		1	I	ST	PORTJ<2> data input.
	SEG34	0	O	ANA	LCD Segment 34 output; disables all other pin functions.
	$\overline{WRL}$	x	O	DIG	External Memory Bus Write Low ($\overline{WRL}$) signal.
RJ3/SEG35/ $\overline{WRH}$	RJ3	0	O	DIG	LATJ<3> data output.
		1	I	ST	PORTJ<3> data input.
	SEG35	0	O	ANA	LCD Segment 35 output; disables all other pin functions.
	$\overline{WRH}$	x	O	DIG	External Memory Bus Write High ($\overline{WRH}$) signal.
RJ4/SEG39/ BA0	RJ4	0	O	DIG	LATJ<4> data output.
		1	I	ST	PORTJ<4> data input.
	SEG39	0	O	ANA	LCD Segment 39 output; disables all other pin functions.
	BA0	x	O	DIG	External Memory Bus Byte Access 0 (BA0) signal.
RJ5/SEG38/$\overline{CE}$	RJ5	0	O	DIG	LATJ<5> data output.
		1	I	ST	PORTJ<5> data input.
	SEG38	0	O	ANA	LCD Segment 38 output; disables all other pin functions.
	$\overline{CE}$	x	O	DIG	External Memory Bus Chip Enable ($\overline{CE}$) signal.

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

FIGURE 13-19: TYPE-B WAVEFORMS IN 1/4 MUX, 1/3 BIAS DRIVE



14.0 TIMER0 MODULE

The Timer0 module incorporates the following features:

- Software-selectable operation as a timer or counter in both 8-bit or 16-bit modes
- Readable and writable registers
- Dedicated 8-bit, software programmable prescaler
- Selectable clock source (internal or external)
- Edge select for external clock
- Interrupt-on-overflow

The T0CON register (Register 14-1) controls all aspects of the module's operation, including the prescale selection. It is both readable and writable.

Figure 14-1 provides a simplified block diagram of the Timer0 module in 8-bit mode. Figure 14-2 provides a simplified block diagram of the Timer0 module in 16-bit mode.

REGISTER 14-1: T0CON: TIMER0 CONTROL REGISTER

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
TMR0ON	T08BIT	T0CS1	T0CS0	PSA	T0PS2	T0PS1	T0PS0
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 **TMR0ON:** Timer0 On/Off Control bit
1 = Enables Timer0
0 = Stops Timer0

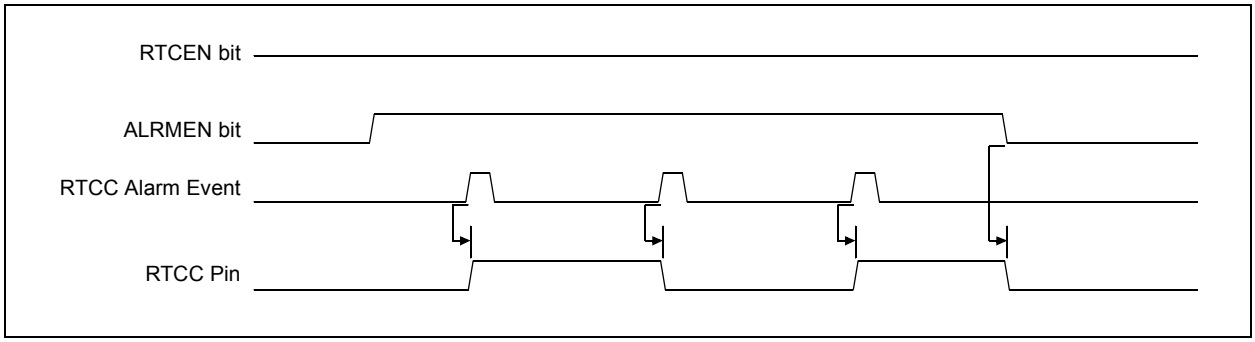
bit 6 **T08BIT**: Timer0 8-Bit/16-Bit Control bit
1 = Timer0 is configured as an 8-bit timer/counter
0 = Timer0 is configured as a 16-bit timer/counter

bit 5-4 **T0CS<1:0>**: Timer0 Clock Source Select bit
11 = Increment on high-to-low transition on T0CKI pin
10 = Increment on low-to-high transition on T0CKI pin
01 = Internal clock (FOSC/4)
00 = INTOSC

bit 3 **PSA**: Timer0 Prescaler Assignment bit
1 = Timer0 prescaler is not assigned; Timer0 clock input bypasses prescaler
0 = Timer0 prescaler is assigned; Timer0 clock input comes from prescaler output

bit 2-0 **T0PS<2:0>**: Timer0 Prescaler Select bits
111 = 1:256 Prescale value
110 = 1:128 Prescale value
101 = 1:64 Prescale value
100 = 1:32 Prescale value
011 = 1:16 Prescale value
010 = 1:8 Prescale value
001 = 1:4 Prescale value
000 = 1:2 Prescale value

FIGURE 17-6: TIMER PULSE GENERATION



17.4 Sleep Mode

The timer and alarm continue to operate while in Sleep mode. The operation of the alarm is not affected by Sleep, as an alarm event can always wake-up the CPU.

The Idle mode does not affect the operation of the timer or alarm.

17.5 Reset

17.5.1 DEVICE RESET

When a device Reset occurs, the ALRMRPT register is forced to its Reset state, causing the alarm to be disabled (if enabled prior to the Reset). If the RTCC was enabled, it will continue to operate when a basic device Reset occurs.

17.5.2 POWER-ON RESET (POR)

The RTCCON1 and ALRMRPT registers are reset only on a POR. Once the device exits the POR state, the clock registers should be reloaded with the desired values.

The timer prescaler values can be reset only by writing to the SECONDS register. No device Reset can affect the prescalers.

19.4.2 PWM DUTY CYCLE

The PWM duty cycle is specified, to use CCP4 as an example, by writing to the CCPR4L register and to the CCP4CON<5:4> bits. Up to 10-bit resolution is available. The CCPR4L contains the eight MSbs and the CCP4CON<5:4> contains the two LSbs. This 10-bit value is represented by CCPR4L:CCP4CON<5:4>. The following equation is used to calculate the PWM duty cycle in time:

EQUATION 19-2: PWM DUTY CYCLE (IN TIME)

PWM Duty Cycle = (CCPR4L:CCP4CON<5:4>) • TOSC • (TMR2 Prescale Value)

CCPR4L and CCP4CON<5:4> can be written to at any time, but the duty cycle value is not latched into CCPR4H until after a match between PR2 and TMR2 occurs (that is, the period is complete). In PWM mode, CCPR4H is a read-only register.

The CCPR4H register and a two-bit internal latch are used to double-buffer the PWM duty cycle. This double-buffering is essential for glitchless PWM operation.

When the CCPR4H and two-bit latch match TMR2, concatenated with an internal two-bit Q clock or two bits of the TMR2 prescaler, the CCP4 pin is cleared.

The maximum PWM resolution (bits) for a given PWM frequency is given by the equation:

EQUATION 19-3: PWM RESOLUTION

$$\text{PWM Resolution (max)} = \frac{\log\left(\frac{FOSC}{FPWM}\right)}{\log(2)} \text{bits}$$

Note: If the PWM duty cycle value is longer than the PWM period, the CCP4 pin will not be cleared.

TABLE 19-4: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 40 MHz

PWM Frequency	2.44 kHz	9.77 kHz	39.06 kHz	156.25 kHz	312.50 kHz	416.67 kHz
Timer Prescaler (1, 4, 16)	16	4	1	1	1	1
PR2 Value	FFh	FFh	FFh	3Fh	1Fh	17h
Maximum Resolution (bits)	10	10	10	8	7	6.58

19.4.3 SETUP FOR PWM OPERATION

To configure the CCP module for PWM operation using CCP4 as an example:

1. Set the PWM period by writing to the PR2 register.
2. Set the PWM duty cycle by writing to the CCPR4L register and CCP4CON<5:4> bits.
3. Make the CCP4 pin an output by clearing the appropriate TRIS bit.
4. Set the TMR2 prescale value, then enable Timer2 by writing to T2CON.
5. Configure the CCP4 module for PWM operation.

REGISTER 20-6: SSPxSTAT: MSSPx STATUS REGISTER (I^2C MODE)

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE	D/$\overline{A}$	P(1)	S(1)	R/$\overline{W}$(2,3)	UA	BF
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 **SMP:** Slew Rate Control bit

In Master or Slave mode:
1 = Slew rate control is disabled for Standard Speed mode (100 kHz and 1 MHz)
0 = Slew rate control is enabled for High-Speed mode (400 kHz)

bit 6 **CKE:** SMBus Select bit

In Master or Slave mode:
1 = Enables SMBus-specific inputs
0 = Disables SMBus-specific inputs

bit 5 **D/$\overline{A}$:** Data/$\overline{\text{Address}}$ bit

In Master mode:
Reserved.

In Slave mode:
1 = Indicates that the last byte received or transmitted was data
0 = Indicates that the last byte received or transmitted was address

bit 4 **P:** Stop bit(1)

1 = Indicates that a Stop bit has been detected last
0 = Stop bit was not detected last

bit 3 **S:** Start bit(1)

1 = Indicates that a Start bit has been detected last
0 = Start bit was not detected last

bit 2 **R/$\overline{W}$:** Read/$\overline{\text{Write}}$ Information bit(2,3)

In Slave mode:
1 = Read
0 = Write

In Master mode:
1 = Transmit is in progress
0 = Transmit is not in progress

bit 1 **UA:** Update Address bit (10-Bit Slave mode only)

1 = Indicates that the user needs to update the address in the SSPxADD register
0 = Address does not need to be updated

bit 0 **BF:** Buffer Full Status bit

In Transmit mode:
1 = SSPxBUF is full
0 = SSPxBUF is empty

In Receive mode:
1 = SSPxBUF is full (does not include the $\overline{\text{ACK}}$ and Stop bits)
0 = SSPxBUF is empty (does not include the $\overline{\text{ACK}}$ and Stop bits)

Note 1: This bit is cleared on Reset and when SSPEN is cleared.

2: This bit holds the R/$\overline{W}$ bit information following the last address match. This bit is only valid from the address match to the next Start bit, Stop bit or not $\overline{\text{ACK}}$ bit.

3: ORing this bit with SEN, RSEN, PEN, RCEN or ACKEN will indicate if the MSSPx is in Active mode.

20.5.9 MASTER MODE

Master mode is enabled by setting and clearing the appropriate SSPMx bits in SSPxCON1, and by setting the SSPEN bit. In Master mode, the SCLx and SDAx lines are manipulated by the MSSPx hardware if the TRIS bits are set.

The Master mode of operation is supported by interrupt generation on the detection of the Start and Stop conditions. The Stop (P) and Start (S) bits are cleared from a Reset or when the MSSPx module is disabled. Control of the I^2C bus may be taken when the P bit is set, or the bus is Idle, with both the S and P bits clear.

In Firmware Controlled Master mode, user code conducts all I^2C bus operations based on Start and Stop bit conditions.

Once Master mode is enabled, the user has six options.

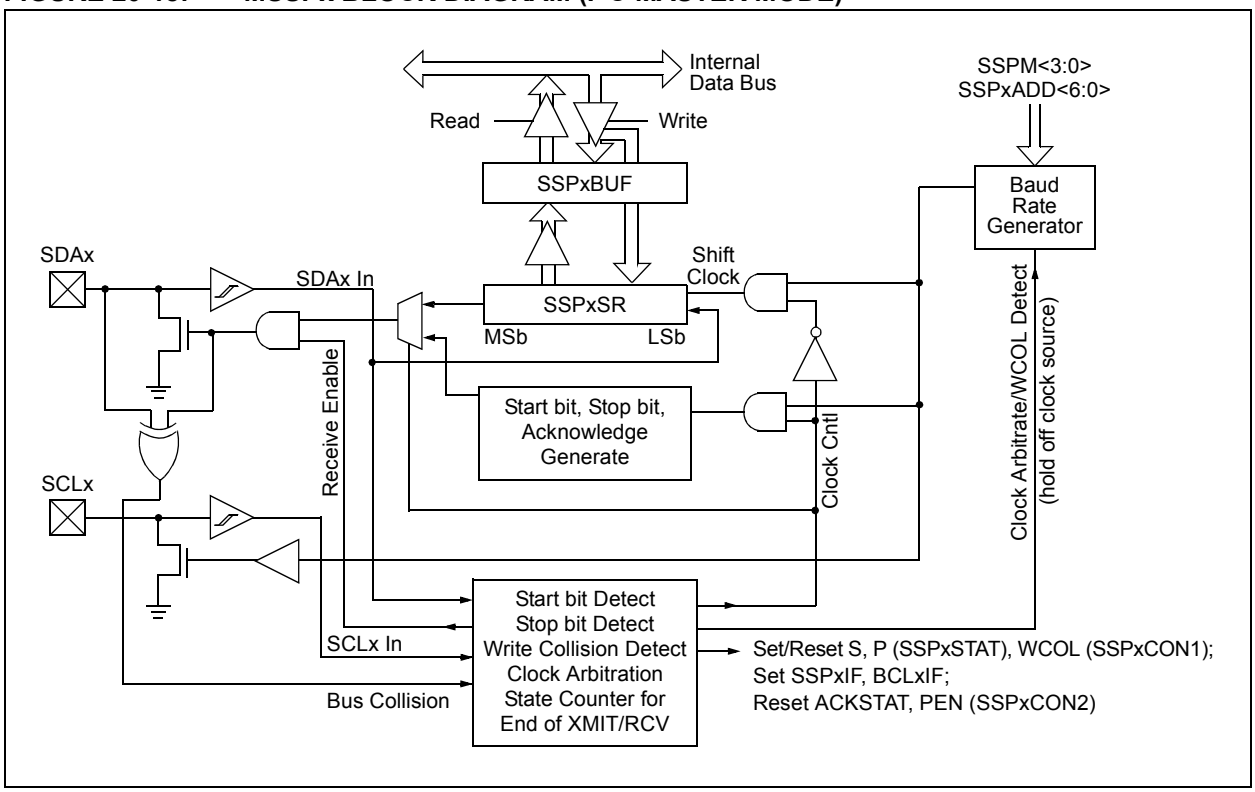
1. Assert a Start condition on SDAx and SCLx.
2. Assert a Repeated Start condition on SDAx and SCLx.
3. Write to the SSPxBUF register initiating transmission of data/address.
4. Configure the I^2C port to receive data.
5. Generate an Acknowledge condition at the end of a received byte of data.
6. Generate a Stop condition on SDAx and SCLx.

Note: The MSSPx module, when configured in I^2C Master mode, does not allow queueing of events. For instance, the user is not allowed to initiate a Start condition and immediately write the SSPxBUF register to initiate transmission before the Start condition is complete. In this case, the SSPxBUF will not be written to and the WCOL bit will be set, indicating that a write to the SSPxBUF did not occur.

The following events will cause the MSSPx Interrupt Flag bit, SSPxIF, to be set (and MSSPx interrupt if enabled):

- Start condition
- Stop condition
- Data transfer byte transmitted/received
- Acknowledge transmitted
- Repeated Start

FIGURE 20-18: MSSPx BLOCK DIAGRAM (I^2C MASTER MODE)



23.0 COMPARATOR MODULE

The analog comparator module contains three comparators that can be independently configured in a variety of ways. The inputs can be selected from the analog inputs and two internal voltage references. The digital outputs are available at the pin level, via PPS-Lite, and can also be read through the control register. Multiple output and interrupt event generations are also available. A generic single comparator from the module is shown in Figure 23-1.

Key features of the module includes:

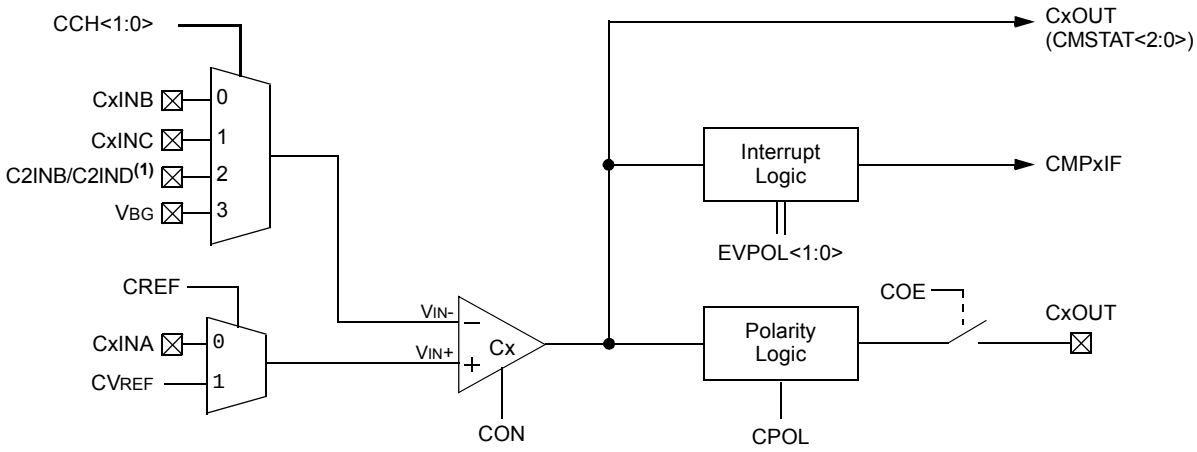
- Independent comparator control
- Programmable input configuration
- Output to both pin and register levels
- Programmable output polarity
- Independent interrupt generation for each comparator with configurable interrupt-on-change

23.1 Registers

The CMxCON registers (CM1CON, CM2CON and CM3CON) select the input and output configuration for each comparator, as well as the settings for interrupt generation (see Register 23-1).

The CMSTAT register (Register 23-2) provides the output results of the comparators. The bits in this register are read-only.

FIGURE 23-1: COMPARATOR SIMPLIFIED BLOCK DIAGRAM



Note 1: Comparator 1 and Comparator 3 use C2INB as an input to the inverted terminal. Comparator 2 uses C2IND as an input to the inverted terminal.

23.6 Comparator Interrupts

The comparator interrupt flag is set whenever any of the following occurs:

- Low-to-high transition of the comparator output
- High-to-low transition of the comparator output
- Any change in the comparator output

The comparator interrupt selection is done by the EVPOL<1:0> bits in the CMxCON register (CMxCON<4:3>).

In order to provide maximum flexibility, the output of the comparator may be inverted using the CPOL bit in the CMxCON register (CMxCON<5>). This is functionally identical to reversing the inverting and non-inverting inputs of the comparator for a particular mode.

An interrupt is generated on the low-to-high or high-to-low transition of the comparator output. This mode of interrupt generation is dependent on EVPOL<1:0> in the CMxCON register. When EVPOL<1:0> = 01 or 10, the interrupt is generated on a low-to-high or high-to-low transition of the comparator output. Once the interrupt is generated, it is required to clear the interrupt flag by software.

When EVPOL<1:0> = 11, the comparator interrupt flag is set whenever there is a change in the output value of either comparator. Software will need to maintain information about the status of the output bits, as read from CMSTAT<2:0>, to determine the actual change that occurred.

The CMPxIF<2:0> (PIR6<2:0>) bits are the Comparator Interrupt Flags. The CMPxIF bits must be reset by clearing them. Since it is also possible to write a '1' to this register, a simulated interrupt may be initiated. Table 23-2 shows the interrupt generation with respect to comparator input voltages and EVPOL bit settings.

Both the CMPxIE bits (PIE6<2:0>) and the PEIE bit (INTCON<6>) must be set to enable the interrupt. In addition, the GIE bit (INTCON<7>) must also be set. If any of these bits are clear, the interrupt is not enabled, though the CMPxIF bits will still be set if an interrupt condition occurs.

A simplified diagram of the interrupt section is shown in Figure 23-3.

Note:	CMPxIF will not be set when EVPOL<1:0> = 00.

TABLE 23-2: COMPARATOR INTERRUPT GENERATION

CPOL	EVPOL<1:0>	Comparator Input Change	CxOUT Transition	Interrupt Generated
0	00	VIN+ > VIN-	Low-to-High	No
		VIN+ < VIN-	High-to-Low	No
	01	VIN+ > VIN-	Low-to-High	Yes
		VIN+ < VIN-	High-to-Low	No
	10	VIN+ > VIN-	Low-to-High	No
		VIN+ < VIN-	High-to-Low	Yes
	11	VIN+ > VIN-	Low-to-High	Yes
		VIN+ < VIN-	High-to-Low	Yes
1	00	VIN+ > VIN-	High-to-Low	No
		VIN+ < VIN-	Low-to-High	No
	01	VIN+ > VIN-	High-to-Low	No
		VIN+ < VIN-	Low-to-High	Yes
	10	VIN+ > VIN-	High-to-Low	Yes
		VIN+ < VIN-	Low-to-High	No
	11	VIN+ > VIN-	High-to-Low	Yes
		VIN+ < VIN-	Low-to-High	Yes

26.1 CTMU Registers

The control registers for the CTMU are:

- CTMUCON1
- CTMUCON2
- CTMUCON3
- CTMUCON4

The CTMUCON1 and CTMUCON3 registers (Register 26-1 and Register 26-3) contain control bits for configuring the CTMU module edge source selection, edge source polarity selection, edge sequencing, A/D trigger, analog circuit capacitor discharge and enables. The CTMUCON2 register (Register 26-2) has bits for selecting the current source range and current source trim.

REGISTER 26-1: CTMUCON1: CTMU CONTROL REGISTER 1

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CTMUEN	—	CTMUSIDL	TGEN	EDGEN	EDGSEQEN	IDISSEN	CTTRIG
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 **CTMUEN:** CTMU Enable bit
1 = Module is enabled
0 = Module is disabled

bit 6 **Unimplemented:** Read as ‘0’

bit 5 **CTMUSIDL:** Stop in Idle Mode bit
1 = Discontinues module operation when device enters Idle mode
0 = Continues module operation in Idle mode

bit 4 **TGEN:** Time Generation Enable bit
1 = Enables edge delay generation
0 = Disables edge delay generation

bit 3 **EDGEN:** Edge Enable bit
1 = Edges are not blocked
0 = Edges are blocked

bit 2 **ESGSEQEN:** Edge Sequence Enable bit
1 = Edge 1 event must occur before Edge 2 event can occur
0 = No edge sequence is needed

bit 1 **IDISSEN:** Analog Current Source Control bit
1 = Analog current source output is grounded
0 = Analog current source output is not grounded

bit 0 **CTTRIG:** CTMU Special Event Trigger bit
1 = CTMU Special Event Trigger is enabled
0 = CTMU Special Event Trigger is disabled

26.4.2 CAPACITANCE CALIBRATION

There is a small amount of capacitance from the internal A/D Converter sample capacitor, as well as stray capacitance from the circuit board traces and pads that affect the precision of capacitance measurements. A measurement of the stray capacitance can be taken by making sure the desired capacitance to be measured has been removed.

After removing the capacitance to be measured:

1. Initialize the A/D Converter and the CTMU.
2. Set EDG1STAT (= 1).
3. Wait for a fixed delay of time, *t*.
4. Clear EDG1STAT.
5. Perform an A/D conversion.
6. Calculate the stray and A/D sample capacitances:

$$C_{OFFSET} = C_{STRAY} + C_{AD} = (I \bullet t)/V$$

Where:

- I is known from the current source measurement step
- t is a fixed delay
- V is measured by performing an A/D conversion

This measured value is then stored and used for calculations of time measurement or subtracted for capacitance measurement. For calibration, it is expected that the capacitance of C_{STRAY} + C_{AD} is approximately known; C_{AD} is approximately 4 pF.

An iterative process may be required to adjust the time, t, that the circuit is charged to obtain a reasonable voltage reading from the A/D Converter. The value of *t* may be determined by setting C_{OFFSET} to a theoretical value and solving for *t*. For example, if C_{STRAY} is theoretically calculated to be 11 pF, and *V* is expected to be 70% of V_{DD} or 2.31V, t would be:

$$(4\ pF + 11\ pF) \bullet 2.31V/0.55\ mA$$

or 63 μs.

See Example 26-3 for a typical routine for CTMU capacitance calibration.

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

Example 1: 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

DECF **Decrement f**

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

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

Operation: (f) – 1 → dest

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

Encoding:

0000	01da	ffff	ffff

Description: Decrement register, 'f'. 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 ≤ 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: DECF CNT, 1, 0

Before Instruction
CNT = 01h
Z = 0
After Instruction
CNT = 00h
Z = 1

LFSR **Load FSR**

Syntax: LFSR f, k

Operands: $0 \le f \le 2$
$0 \le k \le 4095$

Operation: $k \rightarrow FSRf$

Status Affected: None

Encoding:

1110 1111	1110 0000	00ff $k_7$kkk	k_{11}kkk kkkk

Description: The 12-bit literal 'k' is loaded into the file select register pointed to by 'f'.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k' MSB	Process Data	Write literal 'k' MSB to FSRfH
Decode	Read literal 'k' LSB	Process Data	Write literal 'k' to FSRfL

Example: `LFSR 2, 3ABh`

After Instruction
FSR2H = 03h
FSR2L = ABh

MOVF **Move f**

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

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

Operation: $f \rightarrow dest$

Status Affected: N, Z

Encoding:

0101	00da	ffff	ffff

Description: The contents of register 'f' are moved to a destination dependent upon the status of 'd'. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f'. Location 'f' can be anywhere in the 256-byte bank.

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

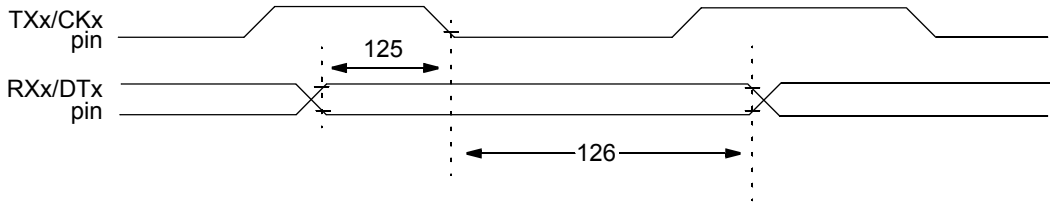
Q Cycle Activity:

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

Example: `MOVF REG, 0, 0`

Before Instruction
REG = 22h
W = FFh
After Instruction
REG = 22h
W = 22h

FIGURE 30-21: EUSARTx/AUSARTx SYNCHRONOUS RECEIVE (MASTER/SLAVE) TIMING



Note: Refer to Figure 30-2 for load conditions.

TABLE 30-39: EUSARTx/AUSARTx SYNCHRONOUS RECEIVE REQUIREMENTS

Param. No.	Symbol	Characteristic	Min.	Max.	Units	Conditions
125	TDTV2CKL	SYNC RCV (MASTER and SLAVE) Data Hold before CKx ↓ (DTx hold time)	10	—	ns	—
126	TCKL2DTL	Data Hold after CKx ↓ (DTx hold time)	15	—	ns	—