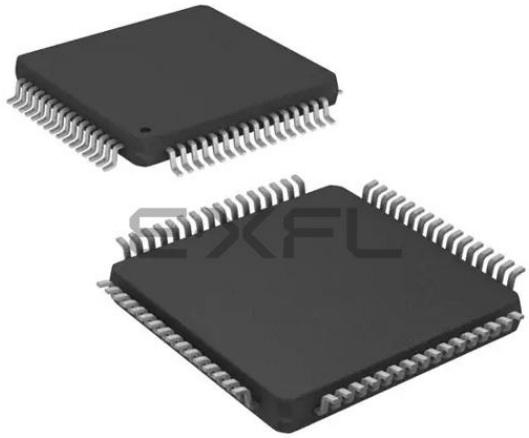


Welcome to E-XFL.COM

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	52
Program Memory Size	128KB (64K x 16)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	4K x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 3.6V
Data Converters	A/D 16x10b/12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-TQFP
Supplier Device Package	64-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f67j94t-i-pt

TABLE 1-4: PIC18FXXJ94 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	100	80	64			
ALE/SEG32/RJ0	77	62				
ALE				O	DIG	External memory address latch enable.
SEG32				O	Analog	SEG32 output for LCD.
RJ0				I/O	ST/DIG	General purpose I/O pin.
$\overline{OE}$/SEG33/RJ1	76	61				
$\overline{OE}$				O	DIG	External memory output enable.
SEG33				O	Analog	SEG33 output for LCD.
RJ1				I/O	ST/DIG	General purpose I/O pin.
$\overline{WRL}$/SEG34/RJ2	75	60				
$\overline{WRL}$				O	DIG	External memory write low control.
SEG34				O	Analog	SEG34 output for LCD.
RJ2				I/O	ST/DIG	General purpose I/O pin.
$\overline{WRH}$/SEG35/RJ3	74	59				
$\overline{WRH}$				O	DIG	External memory write high control.
SEG35				O	Analog	SEG35 output for LCD.
RJ3				I/O	ST/DIG	General purpose I/O pin.
BA0/SEG39/RJ4	49	39				
BA0				O	DIG	External Memory Byte Address 0 control
SEG39				O	Analog	SEG39 output for LCD.
RJ4				I/O	ST/DIG	General purpose I/O pin.
$\overline{CE}$/SEG38/RJ5	50	40				
$\overline{CE}$				O	DIG	External memory chip enable control.
SEG38				O	Analog	SEG38 output for LCD.
RJ5				I/O	ST/DIG	General purpose I/O pin.
$\overline{LB}$/SEG37/RJ6	51	41				
$\overline{LB}$				O	DIG	External memory low byte control.
SEG37				O	Analog	SEG37 output for LCD.
RJ6				I/O	ST/DIG	General purpose I/O pin.
$\overline{UB}$/SEG36/RJ7	52	42				
$\overline{UB}$				O	DIG	External memory high byte control.
SEG36				O	Analog	SEG36 output for LCD.
RJ7				I/O	ST/DIG	General purpose I/O pin.

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

3.2.1 CLOCK SWITCHING MODE CONFIGURATION BITS

The FSCMx Configuration bits (CONFIG3L<5:4>) are used to jointly configure device clock switching and the Fail-Safe Clock Monitor (FSCM). Clock switching is enabled only when FSCM1 is programmed ('0'). The FSCM is enabled only when FSCM<1:0> are both programmed ('00').

3.2.2 OSC1 AND OSC2 PIN FUNCTIONS IN NON-CRYSTAL MODES

When the Primary Oscillator on OSC1 and OSC2 is not configured as the clock source (POSCMD<1:0> = 11), the OSC1 pin is automatically reconfigured as a digital I/O. In this configuration, as well as when the Primary Oscillator is configured for EC mode (POSCMD<1:0> = 00), the OSC2 pin can also be configured as a digital I/O by programming the CLKOEN Configuration bit (CONFIG2L<5>).

When CLKOEN is unprogrammed ('1'), a FOSC/4 clock output is available on OSC2 for testing or synchronization purposes. With CLKOEN programmed ('0'), the OSC2 pin becomes a general purpose I/O pin. In both of these configurations, the feedback device between OSC1 and OSC2 is turned off to save current.

3.3 Control Registers

The operation of the oscillator is controlled by six Special Function Registers (SFRs):

- OSCCON
- OSCCON2
- OSCCON3
- OSCCON4
- ACTCON
- OSCTUNE

3.3.1 OSCILLATOR CONTROL REGISTER (OSCCON)

The OSCCON register (Register 3-1) is the main control register for the oscillator. It controls clock source switching and allows the monitoring of clock sources.

The COSCx (OSCCON<6:4>) Status bits are read-only bits that indicate the current oscillator source the device is operating from. The COSCx bits default to the Internal Fast RC Oscillator with Postscaler (FRCDIV), configured for 4 MHz, on a Power-on Reset (POR) and Master Clear Reset ($\overline{MCLR}$). A clock switch will automatically be performed to the new oscillator source selected by the FOSCx Configuration bits (CONFIG2L<2:0>). The COSCx bits will change to indicate the new oscillator source at the end of a clock switch operation.

The NOSCx Status bits select the clock source for the next clock switch operation. On POR and $\overline{MCLR}$s, these bits automatically select the oscillator source defined by the FOSCx Configuration bits. These bits can be modified by software.

Setting the CLKLOCK bit (OSCCON2<7>) prevents clock switching if the FSCM1 Configuration bit is set. If the FSCM1 bit is clear, the CLKLOCK bit state is ignored and clock switching can occur.

The IOLOCK bit (OSCCON2<6>) is used to unlock the Peripheral Pin Select (PPS) feature; it has no function in the system clock's operation.

The LOCK Status bit (OSCCON2<5>) is read-only and indicates the status of the PLL circuit. It is set when the PLL achieves a frequency lock and is reset when a valid clock switching sequence is initiated. It reads as '0' whenever the PLL is not used as part of the current clock source.

The CF Status bit (OSCCON2<3>) is a readable/clearable Status bit that indicates a clock failure; it is reset whenever a valid clock switch occurs.

The POSCEN bit (OSCCON2<2>) is used to control the operation of the Primary Oscillator in Sleep mode. Setting this bit bypasses the normal automatic shutdown of the oscillator whenever Sleep mode is invoked.

The Secondary Oscillator can be turned on by a variety of options:

- SOSCGO – OSCCON2<1>
- SOSCSEL – CONFIG2L<3>
- FOSC<2:0> – CONFIG2L<2:0>
- DSWDTOSC – CONFIG8H<1>
- RTCEN – RTCCON1<7>
- SOSCEN – T1CON<3>, T3CON<3> or T5CON<3>

The ACTCON register (Register 3-10) controls the Active Clock Tuning features.

TABLE 3-3: SYSTEM CLOCK OPTIONS DURING USB OPERATION

MCU Clock Division (CPDIV<1:0>)	System Clock Frequency (Instruction Rate in MIPS®)
None (00)	64 MHz (16)
÷2 (01)	32 MHz (8)
÷4 (10)	16 MHz (4)
÷8 (11)	8 MHz (2)(1)

Note 1: These options are not compatible with USB operation. They may be used whenever the PLL branch is selected and the USB module is disabled.

TABLE 3-4: VALID PRIMARY OSCILLATOR CONFIGURATIONS FOR USB OPERATIONS

Input Oscillator Frequency	Clock Mode	PLL Division (PLLDIV<2:0>)
48 MHz	ECPLL	÷12 (111)
32 MHz	ECPLL	÷ 8 (110)
24 MHz	HSPLL, ECPLL	÷6 (101)
20 MHz	HSPLL, ECPLL	÷5 (100)
16 MHz	HSPLL, ECPLL	÷4 (011)
12 MHz	HSPLL, ECPLL	÷3 (010)
8 MHz	ECPLL, MSPLL, FRCPLL(1)	÷2 (001)
4 MHz	ECPLL, MSPLL, FRCPLL(1)	÷1 (000)

Note 1: FRCPLL with ±0.25% accuracy can be used for USB operation.

Note:	Because of USB clocking accuracy requirements (±0.25%), not all PIC18F devices support the use of the FRCPLL system clock configuration for USB operation. Refer to the specific device data sheet for details on the FRC Oscillator module.

3.8.2 CONSIDERATIONS FOR USING THE PLL BLOCK

All PLL blocks use the LOCK bit (OSCCON2<5>) as a read-only Status bit to indicate the lock status of the PLL. It is automatically set after the typical time delay for the PLL to achieve lock, designated as TLOCK. It is cleared at a POR and on clock switches when the PLL is selected as a clock source. It remains clear when any clock source not using the PLL is selected.

If the PLL does not stabilize properly during start-up, the LOCK bit may not reflect the actual status of the PLL lock, nor does it detect when the PLL loses lock during normal operation. Refer to the **"Electrical Characteristics"** section in the specific device data sheet for further information on the PLL lock interval.

Using any PLL block with the FRC Oscillator provides a stable system clock for microcontroller operations. USB operation is only possible with FRC Oscillators that are implemented with ±1/4% frequency accuracy. Serial communications using USART are only possible when FRC Oscillators are implemented with ±2% frequency accuracy. The PIC18F97J94 family is able to meet the required oscillator accuracy for both USB and USART providing stable communication by use of its active clock tuning feature. Refer to **Section 3.13.3 "Active Clock Tuning (ACT) Module"** for more information.

If an application is being migrated between PIC18F platforms with different PLL blocks, the differences in PLL and clock options may require the reconfiguration of peripherals that use the system clock. This is particularly true with serial communication peripherals, such as the USARTs.

3.9 Secondary Oscillator (SOSC)

In most PIC18F devices, the low-power Secondary Oscillator (SOSC) is implemented to run with a 32.768 kHz crystal. The oscillator is located on the SOSCO and SOSCI device pins, and serves as a secondary crystal clock source for low-power operation. It is used to drive Timer1, Real-Time Clock and Calendar (RTCC) and other modules requiring a clock signal while in low-power operation.

3.9.1 ENABLING THE SECONDARY OSCILLATOR

The operation of the SOSC is selected by the FOSCx Configuration bits or by selection of the NOSCx bits (OSCCON<2:0>). The SOSC can also be enabled by setting the SOSCEN bit in Timer1, Timer3 or Timer5. The SOSC has a long start-up time; therefore, to avoid delays for peripheral start-up, the SOSC can be manually started using one of the SOSCEN bits.

3.13.1 ENABLING CLOCK SWITCHING

To enable clock switching, the FCKSM1 Configuration bit must be programmed to '0'. If the FCKSM1 Configuration bit is unprogrammed ('1'), the clock switching function and Fail-Safe Clock Monitor function are disabled; this is the default setting.

The NOSCx control bits (OSCCON<2:0>) do not control the clock selection when clock switching is disabled. However, the COSCx bits (OSCCON<6:4>) will reflect the clock source selected by the FOSC Configuration bits.

3.13.2 OSCILLATOR SWITCHING SEQUENCE

At a minimum, performing a clock switch requires this basic sequence:

1. If desired, read the COSCx bits (OSCCON<6:4>) to determine the current oscillator source.
2. Clear the CLKLOCK bit (OSCCON2<7>) to enable writes to the NOSCx bits (OSCCON<2:0>).
3. Write the appropriate value to the NOSCx control bits (OSCCON<2:0>) for the new oscillator source.

Once the basic sequence is completed, the system clock hardware responds automatically as follows:

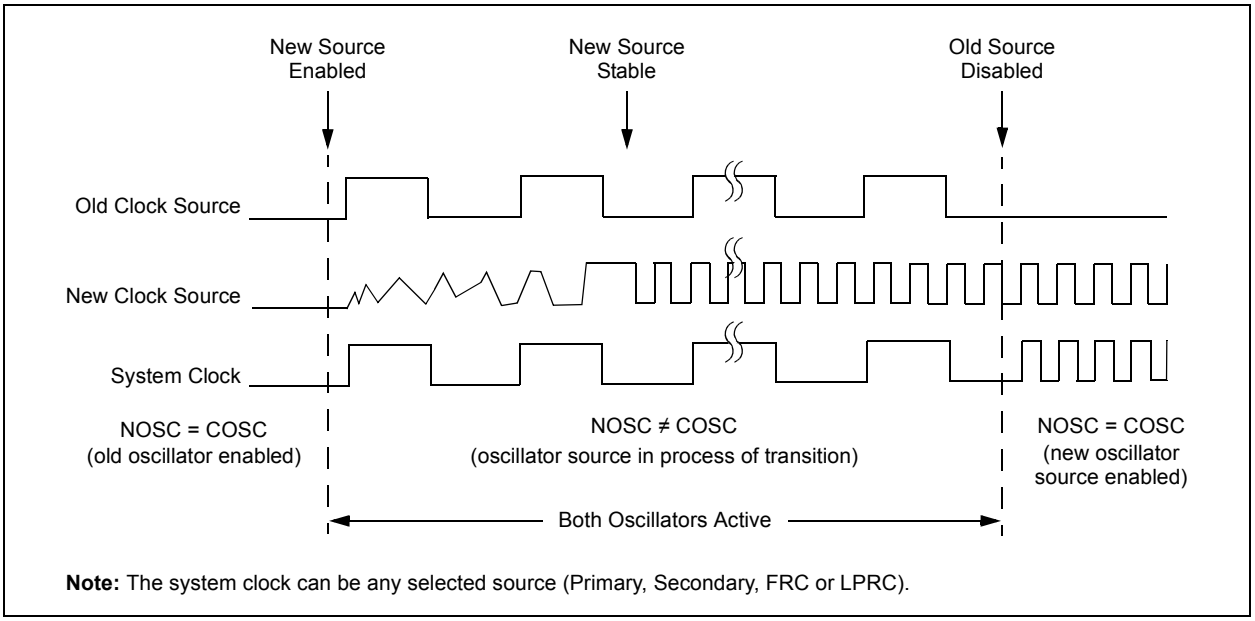
1. The clock switching hardware compares the COSC Status bits with the new value of the NOSC control bits. If they are the same, then the clock switch is a redundant operation. If they are different, then a valid clock switch has been initiated.
2. The new oscillator is turned on by the hardware if it is not currently running. If a crystal oscillator must be turned on, the hardware will wait until the OST expires. If the new source is using the PLL, then the hardware waits until a PLL lock is detected (LOCK = 1).
3. The hardware waits for the new clock source to stabilize and then performs the clock switch.
4. The NOSCx bit values are transferred to the COSCx Status bits.
5. The old clock source is turned off at this time, with the exception of LPRC (if WDT or FSCM is enabled) or SOSC (if it is enabled by one of the timer sources).

The timing of the transition between clock sources is shown in Figure 3-9.

Note 1: The processor will continue to execute code throughout the clock switching sequence. Timing-sensitive code should not be executed during this time.

2: Direct clock switches between any Primary Oscillator mode with PLL and FRCPLL mode are not permitted. This applies to clock switches in either direction. In these instances, the application must switch to FRC mode as a transition clock source between the two PLL modes.

FIGURE 3-9: CLOCK TRANSITION TIMING DIAGRAM



Note: The system clock can be any selected source (Primary, Secondary, FRC or LPRC).

5.2 Power-on Reset (POR)

The PIC18F97J94 family has two types of Power-on Resets:

- POR
- VBAT POR

POR is the legacy PIC18J series Power-on Reset which monitors core power supply. The second, VBAT POR, monitors voltage on the VBAT pin. These POR circuits use the same technique to enable and monitor their respective power source for adequate voltage levels to ensure proper chip operation. There are two threshold voltages associated with them. The first voltage is the device threshold voltage, VPOR. The device threshold voltage is the voltage at which the POR module becomes operable. The second voltage associated with a POR event is the POR circuit threshold voltage. Once the correct threshold voltage is detected, a power-on event occurs and the POR module hibernates to minimize current consumption.

A power-on event generates an internal POR pulse when a VDD rise is detected. The device supply voltage characteristics must meet the specified starting voltage, VPOR, and rise rate requirements, SVDD, to generate the POR pulse. In particular, VDD must fall below VPOR before a new POR is initiated. For more information on the VPOR and VDD rise rate specifications, refer to **Section 30.0 "Electrical Specifications"**.

5.2.1 POR CIRCUIT

The POR circuit behaves differently than VBAT POR once the POR state becomes active. The internal POR pulse resets the POR timer and places the device in the Reset state. The POR also selects the device clock source identified by the Oscillator Configuration bits. After the POR pulse is generated, the POR circuit inserts a small delay, TCSD, to ensure that internal device bias circuits are stable.

After the expiration of TCSD, a delay, TPWRT, is always inserted every time the device resumes operation after any power-down. During this time, code execution is disabled. The PWRT is used to extend the duration of a power-up sequence to permit the on-chip band gap and regulator to stabilize and to load the Configuration Word settings. The on-chip regulator is always enabled and its stabilization time is shorter than other concurrently running delays, and does not extend start-up time.

The power-on event clears the $\overline{\text{BOR}}$ and $\overline{\text{POR}}$ Status bits (RCON<1:0>); it does not change for any other Reset event. $\overline{\text{POR}}$ is not reset to '1' by any hardware event. To capture multiple events, the user manually resets the bit to '1' in software following any Power-on Reset. Alternatively, the VDDPOR (RCON3<2>) bit can be used; it is set on a VDD POR event. It must be cleared after any Power-on Reset to detect subsequent VDD POR events.

After TPWRT expires, an additional start-up time for the system clock (either TOST, TIOBST and TRC, depending on the source) occurs while the clock source becomes stable. Internal Reset is then released and the device is no longer held in Reset (Table 5-2). Once all of the delays have expired, the system clock is released and code execution can begin. Refer to **Section 30.0 "Electrical Specifications"** for more information on the values of the delay parameters.

Note:	When the device exits the Reset condition (begins normal operation), the device operating parameters (voltage, frequency, temperature, etc.) must be within their operating ranges; otherwise, the device will not function correctly. The user must ensure that the delay between the time power is first applied, and the time, INTERNAL RESET, becomes inactive, is long enough to get all operating parameters within specification.

6.4.3.2 FSR Registers and POSTINC, POSTDEC, PREINC and PLUSW

In addition to the INDF operand, each FSR register pair also has four additional indirect operands. Like INDF, these are “virtual” registers that cannot be indirectly read or written to. Accessing these registers actually accesses the associated FSR register pair, but also performs a specific action on its stored value.

These operands are:

- POSTDEC – Accesses the FSR value, then automatically decrements it by ‘1’ afterwards
- POSTINC – Accesses the FSR value, then automatically increments it by ‘1’ afterwards
- PREINC – Increments the FSR value by ‘1’, then uses it in the operation
- PLUSW – Adds the signed value of the W register (range of -127 to 128) to that of the FSR and uses the new value in the operation

In this context, accessing an INDF register uses the value in the FSR registers without changing them. Similarly, accessing a PLUSW register gives the FSR value, offset by the value in the W register, with neither value actually changed in the operation. Accessing the other virtual registers changes the value of the FSR registers.

Operations on the FSRs with POSTDEC, POSTINC and PREINC affect the entire register pair. Rollovers of the FSRnL register, from FFh to 00h, carry over to the FSRnH register. On the other hand, results of these operations do not change the value of any flags in the STATUS register (for example, Z, N and OV bits).

The PLUSW register can be used to implement a form of Indexed Addressing in the data memory space. By manipulating the value in the W register, users can reach addresses that are fixed offsets from pointer addresses. In some applications, this can be used to implement some powerful program control structure, such as software stacks, inside of data memory.

6.4.3.3 Operations by FSRs on FSRs

Indirect Addressing operations that target other FSRs or virtual registers represent special cases. For example, using an FSR to point to one of the virtual registers will not result in successful operations.

As a specific case, assume that the FSR0H:FSR0L registers contain FE7h, the address of INDF1. Attempts to read the value of the INDF1, using INDF0 as an operand, will return 00h. Attempts to write to INDF1, using INDF0 as the operand, will result in a NOP.

On the other hand, using the virtual registers to write to an FSR pair may not occur as planned. In these cases, the value will be written to the FSR pair, but without any incrementing or decrementing. Thus, writing to INDF2 or POSTDEC2 will write the same value to the FSR2H:FSR2L.

Since the FSRs are physical registers mapped in the SFR space, they can be manipulated through all direct operations. Users should proceed cautiously when working on these registers, however, particularly if their code uses Indirect Addressing.

Similarly, operations by Indirect Addressing are generally permitted on all other SFRs. Users should exercise the appropriate caution, so that they do not inadvertently change settings that might affect the operation of the device.

6.5 Program Memory and the Extended Instruction Set

The operation of program memory is unaffected by the use of the extended instruction set.

Enabling the extended instruction set adds five additional two-word commands to the existing PIC18 instruction set: ADDFSR, CALLW, MOVSF, MOVSS and SUBFSR. These instructions are executed as described in **Section 6.2.4 “Two-Word Instructions”**.

6.6 Data Memory and the Extended Instruction Set

Enabling the PIC18 extended instruction set (XINST Configuration bit = 1) significantly changes certain aspects of data memory and its addressing. Using the Access Bank for many of the core PIC18 instructions introduces a new addressing mode for the data memory space. This mode also alters the behavior of Indirect Addressing using FSR2 and its associated operands.

What does not change is just as important. The size of the data memory space is unchanged, as well as its linear addressing. The SFR map remains the same. Core PIC18 instructions can still operate in both Direct and Indirect Addressing mode. Inherent and literal instructions do not change at all. Indirect Addressing with FSR0 and FSR1 also remains unchanged.

6.6.1 INDEXED ADDRESSING WITH LITERAL OFFSET

Enabling the PIC18 extended instruction set changes the behavior of Indirect Addressing using the FSR2 register pair and its associated file operands. Under the proper conditions, instructions that use the Access Bank – that is, most bit-oriented and byte-oriented instructions – can invoke a form of Indexed Addressing using an offset specified in the instruction. This special addressing mode is known as Indexed Addressing with Literal Offset or the Indexed Literal Offset mode.

When using the extended instruction set, this addressing mode requires the following:

- Use of the Access Bank (‘a’ = 0)
- A file address argument that is less than or equal to 5Fh

Under these conditions, the file address of the instruction is not interpreted as the lower byte of an address (used with the BSR in Direct Addressing) or as an 8-bit address in the Access Bank. Instead, the value is interpreted as an offset value to an Address Pointer specified by FSR2. The offset and the contents of FSR2 are added to obtain the target address of the operation.

6.6.2 INSTRUCTIONS AFFECTED BY INDEXED LITERAL OFFSET MODE

Any of the core PIC18 instructions that can use Direct Addressing are potentially affected by the Indexed Literal Offset Addressing mode. This includes all byte-oriented and bit-oriented instructions, or almost one-half of the standard PIC18 instruction set. Instructions that only use Inherent or Literal Addressing modes are unaffected.

Additionally, byte-oriented and bit-oriented instructions are not affected if they do not use the Access Bank (Access RAM bit = 1), or include a file address of 60h or above. Instructions meeting these criteria will continue to execute as before. A comparison of the different possible addressing modes when the extended instruction set is enabled is shown in Figure 6-9.

Those who desire to use byte-oriented or bit-oriented instructions in the Indexed Literal Offset mode should note the changes to assembler syntax for this mode. This is described in more detail in **Section 29.2.1 “Extended Instruction Syntax”**.

8.8 Operation in Power-Managed Modes

In alternate, power-managed Run modes, the external bus continues to operate normally. If a clock source with a lower speed is selected, bus operations will run at that speed. In these cases, excessive access times for the external memory may result if Wait states have been enabled and added to external memory operations. If operations in a lower power Run mode are anticipated, users should provide in their applications for adjusting memory access times at the lower clock speeds.

In Sleep and Idle modes, the microcontroller core does not need to access data; bus operations are suspended. The state of the external bus is frozen, with the address/data pins and most of the control pins holding at the same state they were in when the mode was invoked. The only potential changes are to the $\overline{CE}$, $\overline{LB}$ and $\overline{UB}$ pins, which are held at logic high.

TABLE 8-3: REGISTERS ASSOCIATED WITH THE EXTERNAL MEMORY BUS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MEMCON(1)	EBDIS	—	WAIT1	WAIT0	—	—	WM1	WM0
PADCFG	RDPU	REPU	RFPU	RGPU	RHPU	RJPU	RKPU	RLPU
PMD4	CMP1MD	CMP2MD	CMP3MD	USBMD	IOCMD	LVDMD	—	EMBMD

Legend: — = unimplemented, read as '0'. Shaded cells are not used during External Memory Bus access.

Note 1: This register is unimplemented on 64-pin devices read as '0'.

10.0 INTERRUPTS

Members of the PIC18F97J94 family of devices have multiple interrupt sources and an interrupt priority feature that allows most interrupt sources to be assigned a high-priority level or a low-priority level. The high-priority interrupt vector is at 0008h and the low-priority interrupt vector is at 0018h. High-priority interrupt events will interrupt any low-priority interrupts that may be in progress.

The registers for controlling interrupt operation are:

- RCON
- INTCON
- INTCON2
- INTCON3
- PIR1, PIR2, PIR3, PIR4, PIR5 and PIR6
- PIE1, PIE2, PIE3, PIE4, PIE5 and PIE6
- IPR1, IPR2, IPR3, IPR5, IPR5 and IPR6

It is recommended that the Microchip header files, supplied with MPLAB® IDE, be used for the symbolic bit names in these registers. This allows the assembler/compiler to automatically take care of the placement of these bits within the specified register.

In general, interrupt sources have three bits to control their operation. They are:

- **Flag bit** – Indicating that an interrupt event occurred
- **Enable bit** – Enabling program execution to branch to the interrupt vector address when the flag bit is set
- **Priority bit** – Specifying high priority or low priority

10.1 Mid-Range Compatibility

When the IPEN bit is cleared (default state), the interrupt priority feature is disabled and interrupts are compatible with PIC® microcontroller mid-range devices. In Compatibility mode, the interrupt priority bits of the IPRx registers have no effect. The PEIE/GIEL bit of the INTCON register is the global interrupt enable for the peripherals. The PEIE/GIEL bit disables only the peripheral interrupt sources and enables the peripheral interrupt sources when the GIE/GIEH bit is also set. The GIE/GIEH bit of the INTCON register is the global interrupt enable which enables all non-peripheral interrupt sources and disables all interrupt sources, including the peripherals. All interrupts branch to address 0008h in Compatibility mode.

10.2 Interrupt Priority

The interrupt priority feature is enabled by setting the IPEN bit of the RCON register. When interrupt priority is enabled the GIE/GIEH and PEIE/GIEL global interrupt enable bits of Compatibility mode are replaced by the GIEH high priority, and GIEL low priority, global interrupt enables. When set, the GIEH bit of the INTCON register enables all interrupts that have their associated IPRx register or INTCONx register priority bit set (high priority). When clear, the GIEH bit disables all interrupt sources including those selected as low priority. When clear, the GIEL bit of the INTCON register disables only the interrupts that have their associated priority bit cleared (low priority). When set, the GIEL bit enables the low priority sources when the GIEH bit is also set. When the interrupt flag, enable bit and appropriate Global Interrupt Enable (GIE) bit are all set, the interrupt will vector immediately to address 0008h for high priority, or 0018h for low priority, depending on level of the interrupting source's priority bit. Individual interrupts can be disabled through their corresponding interrupt enable bits.

10.3 Interrupt Response

When an interrupt is responded to, the Global Interrupt Enable bit is cleared to disable further interrupts. The GIE/GIEH bit is the global interrupt enable when the IPEN bit is cleared. When the IPEN bit is set, enabling interrupt priority levels, the GIEH bit is the high priority global interrupt enable and the GIEL bit is the low priority global interrupt enable. High priority interrupt sources can interrupt a low priority interrupt. Low priority interrupts are not processed while high priority interrupts are in progress.

The return address is pushed onto the stack and the PC is loaded with the interrupt vector address (0008h or 0018h). Once in the Interrupt Service Routine, the source(s) of the interrupt can be determined by polling the interrupt flag bits in the INTCONx and PIRx registers. The interrupt flag bits must be cleared by software before re-enabling interrupts to avoid repeating the same interrupt.

The "return from interrupt" instruction, RETFIE, exits the interrupt routine and sets the GIE/GIEH bit (GIEH or GIEL if priority levels are used), which re-enables interrupts.

11.8 PORTG, LATG and TRISG Registers

PORTG width varies depending on pin count. For 64- and 80-pin devices, PORTG is a 6-bit wide, bidirectional port. For 100-pin devices, PORTG is an 8-bit wide bidirectional port. The corresponding Data Direction and Output Latch registers are TRISG and LATG.

PORTG is multiplexed with the EUSART, and CCP, ECCP, Analog, Comparator, RTCC and Timer input functions (Table 11-7). When operating as I/O, all PORTG pins have Schmitt Trigger input buffers. The open-drain functionality for the CCPx and EUSARTx can be configured using ODCONx.

When enabling peripheral functions, care should be taken in defining TRIS bits for each PORTG pin. Some peripherals override the TRIS bit to make a pin an output, while other peripherals override the TRIS bit to make a pin an input. The user should refer to the corresponding peripheral section for the correct TRIS bit settings. The pin override value is not loaded into the TRIS register. This allows read-modify-write of the TRIS register without concern due to peripheral overrides.

EXAMPLE 11-7: INITIALIZING PORTG

```
CLRF    PORTG       ;  Initialize PORTG by
                    ; clearing output
                    ; data latches
BCF     CM1CON, CON ; disable
                    ; comparator 1
CLRF    LATG        ; Alternate method
                    ; to clear output
                    ; data latches
BANKSEL ANCON2      ; Select bank with ACON2 register
MOVLW   0F0h        ; make AN16 to AN19
                    ; digital
MOVWF   ANCON2
BANKSEL TRISG       ; Select bank with TRISG register
MOVLW   04h         ; Value used to
                    ; initialize data
                    ; direction
MOVWF   TRISG       ; Set RG1:RG0 as
                    ; outputs
                    ; RG2 as input
                    ; RG4:RG3 as inputs
```

TABLE 11-7: PORTG FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RG0/RP46/AN8/ SEG28/COM4	RG0	0	O	DIG	LATG<0> data output; not affected by analog input.
		1	I	ST	PORTG<0> data input; disabled when analog input is enabled.
	RP46	x	x	DIG	Reconfigurable Pin 46 for PPS-Lite; TRIS must be set to match input/ output of module.
	AN8	1	I	ANA	A/D Input Channel 8. Default input configuration on POR; does not affect digital output.
	SEG28	0	O	ANA	LCD Segment 28 output; disables all other pin functions.
	COM4	x	O	ANA	LCD Common 4 output; disables all other outputs.
RG1/RP39/ AN19/SEG29/ COM5	RG1	0	O	DIG	LATG<1> data output; not affected by analog input.
		1	I	ST	PORTG<1> data input; disabled when analog input is enabled.
	RP39	x	x	DIG	Reconfigurable Pin 39 for PPS-Lite; TRIS must be set to match input/ output of module.
	AN19	1	I	ANA	A/D Input Channel 19. Default input configuration on POR; does not affect digital output.
	SEG29	0	O	ANA	LCD Segment 29 output; disables all other pin functions.
	COM5	x	O	ANA	LCD Common 5 output; disables all other outputs.
RG2/RP42/ C3INA/AN18/ SEG30/COM6	RG2	0	O	DIG	LATG<2> data output; not affected by analog input.
		1	I	ST	PORTG<2> data input; disabled when analog input is enabled.
	RP42	x	x	DIG	Reconfigurable Pin 42 for PPS-Lite; TRIS must be set to match input/ output of module.
	C3INA	1	I	ANA	Comparator 3 Input A.
	AN18	1	I	ANA	A/D Input Channel 18. Default input configuration on POR; does not affect digital output.
	SEG30	0	O	ANA	LCD Segment 30 output; disables all other pin functions.
	COM6	x	O	ANA	LCD Common 6 output; disables all other outputs.

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).

TABLE 11-13: RPIN REGISTERS AND AVAILABLE FUNCTIONS (CONTINUED)

PPS-Lite Input Peripheral Group 4n + 2	
(1) To Map this Signal	(4) to the Associated RPIN Register
$\overline{SS1}$	RPINR10_11<3:0>
INT2	RPINR26_27<7:4>
IOC2	RPINR20_21<3:0>
IOC6	RPINR24_25<3:0>
MDMIN	RPINR28_29<7:4>
U1TX	RPINR0_1<7:4>
U2RX	RPINR2_3<3:0>
SCK2	RPINR10_11<7:4>
ECCP3	RPINR16_17<7:4>
CCP6	RPINR34_35<3:0>
CCP10	RPINR38_39<3:0>
RVP2	RPINR48_49<3:0>
RVP6	RPINR52_53<3:0>
(2) with this RPn Pin	(3) Write this Corresponding Value
RP2	h'0
RP6	h'1
RP10	h'2
RP14	h'3
RP18	h'4
RP22	h'5
RP26	h'6
RP30	h'7
RP34	h'8
RP38	h'9
RP42	h'A
RP46	h'B
—	h'C
—	h'D
—	h'E
VSS	h'F

PPS-Lite Input Peripheral Group 4n + 3	
(1) To Map this Signal	(4) to the Associated RPIN Register
$\overline{SS2}$	RPINR12_13<7:4>
INT3	RPINR28_29<3:0>
IOC3	RPINR20_21<7:4>
IOC7	RPINR24_25<7:4>
U1RX	RPINR0_1<3:0>
U2TX	RPINR2_3<7:4>
SCK1	RPINR8_9<3:0>
ECCP1	RPINR14_15<7:4>
ECCP2	RPINR16_17<3:0>
CCP4	RPINR32_33<3:0>
RVP3	RPINR48_49<7:4>
RVP7	RPINR52_53<7:4>
(2) with this RPn Pin	(3) Write this Corresponding Value
RP3	h'0
RP7	h'1
RP11	h'2
RP15	h'3
RP19	h'4
RP23	h'5
RP27	h'6
RP31	h'7
RP35	h'8
RP39	h'9
RP43	h'A
—	h'B
—	h'C
—	h'D
—	h'E
VSS	h'F

11.15.3.2 Output Mapping

In contrast to the inputs, the outputs of the Peripheral Pin Select options are mapped on the basis of the pin. In this case, a bit field associated with a particular pin dictates the peripheral output to be mapped. The RPORx registers contain sets of 4-bit fields, with each associated with one RPn pin (see Register 11-5). The value of the bit field corresponds to one of the peripherals and that peripheral's output is mapped to the pin. Each pin has a limited set of peripherals to choose from.

The PPS-Lite peripheral outputs and associated RPn pins have been organized into four groups. It is not possible to map a peripheral to an RPn pin which is outside of its group. To map a peripheral output signal to an RPn pin, use the 4-step process, as indicated in Table 11-14. Choose the RPn pin and the signal; the column on the right shows which value to write to the associated RPORx register.

The peripheral outputs that support Peripheral Pin Selection have no default pins. Since the RPORx registers reset to all '0's, the outputs are all disconnected in the device's default (Reset) state.

The list of peripherals for output mapping also includes a null value of b'0000' because of the mapping technique. This allows unused peripherals to not be connected to a pin. Not all peripherals are available on all pins. For example, the "SDO2" signal is only available on RP0, RP4, RP8, etc. The "SDO2" signal is not available on RP1.

REGISTER 13-5: LCDDATAx: LCD DATA x REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
S(n)Cy	S(n)Cy	S(n)Cy	S(n)Cy	S(n)Cy	S(n)Cy	S(n)Cy	S(n)Cy
bit 7							bit 0

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

bit 7-0 **S(n)Cy:** Pixel On bits

For registers LCDDATA0 through LCDDATA7: n = (0-63), y = 0
For registers LCDDATA8 through LCDDATA15: n = (0-63), y = 1
For registers LCDDATA16 through LCDDATA23: n = (0-63), y = 2
For registers LCDDATA24 through LCDDATA31: n = (0-63), y = 3
For registers LCDDATA32 through LCDDATA39: n = (0-63), y = 4
For registers LCDDATA40 through LCDDATA47: n = (0-63), y = 5
For registers LCDDATA48 through LCDDATA55: n = (0-63), y = 6
For registers LCDDATA56 through LCDDATA63: n = (0-63), y = 7

1 = Pixel on
0 = Pixel off

TABLE 13-2: LCDDATA REGISTERS AND BITS FOR SEGMENT AND COM COMBINATIONS

COM Lines	Segments							
	0 to 7	8 to 15	16 to 23	24 to 31	32 to 39	40 to 47	48 to 55	56 to 63
0	LCDDATA0 S00C0:S07C0	LCDDATA1 S08C0:S15C0	LCDDATA2 S16C0:S23C0	LCDDATA3 S24C0:S31C0	LCDDATA4 S32C0:S39C0	LCDDATA5 S40C0:S47C0	LCDDATA6 S48C0:S55C0	LCDDATA7 S56C0:S63C0
1	LCDDATA8 S00C1:S07C1	LCDDATA9 S08C1:S15C1	LCDDATA10 S16C1:S23C1	LCDDATA11 S24C1:S31C1	LCDDATA12 S32C1:S39C1	LCDDATA13 S40C1:S47C1	LCDDATA14 S48C1:S55C1	LCDDATA15 S56C1:S63C1
2	LCDDATA16 S00C2:S07C2	LCDDATA17 S08C2:S15C2	LCDDATA18 S16C2:S23C2	LCDDATA19 S24C2:S31C2	LCDDATA20 S32C2:S39C2	LCDDATA21 40C2:S47C2	LCDDATA22 S48C2:S55C2	LCDDATA23 S56C2:S63C2
3	LCDDATA24 S00C3:S07C3	LCDDATA25 S08C3:S15C3	LCDDATA26 S16C3:S23C3	LCDDATA27 S24C3:S31C3	LCDDATA28 S32C3:S39C3	LCDDATA29 S40C3:S47C3	LCDDATA30 S48C3:S55C3	LCDDATA31 S56C3:S63C3
4	LCDDATA32 S00C4:S07C4	LCDDATA33 S08C4:S15C4	LCDDATA34 S16C4:S23C4	LCDDATA35 S24C4:S31C4	LCDDATA36 S32C4:S39C4	LCDDATA37 S40C4:S47C4	LCDDATA38 S48C4:S55C4	LCDDATA39 S56C4:S63C4
5	LCDDATA40 S00C5:S07C5	LCDDATA41 S08C5:S15C5	LCDDATA42 S16C5:S23C5	LCDDATA43 S24C5:S31C5	LCDDATA44 S32C5:S39C5	LCDDATA45 S40C5:S47C5	LCDDATA46 S48C5:S55C5	LCDDATA47 S56C5:S63C5
6	LCDDATA48 S00C6:S07C6	LCDDATA49 S08C6:S15C6	LCDDATA50 S16C6:S23C6	LCDDATA51 S24C6:S31C6	LCDDATA52 S32C6:S39C6	LCDDATA53 S40C6:S47C6	LCDDATA54 S48C6:S55C6	LCDDATA55 S56C6:S63C6
7	LCDDATA56 S00C7:S07C7	LCDDATA57 S08C7:S15C7	LCDDATA58 S16C7:S23C7	LCDDATA59 S24C7:S31C7	LCDDATA60 S32C7:S39C7	LCDDATA61 S40C7:S47C7	LCDDATA62 S48C7:S55C7	LCDDATA63 S56C7:S63C7

REGISTER 17-7: YEAR: YEAR VALUE REGISTER(1) (RTCVALL when RTCPTR<1:0> = 11)

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
YRTEN3	YRTEN2	YRTEN1	YRTEN0	YRONE3	YRONE2	YRONE1	YRONE0
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 7-4 **YRTEN<3:0>:** Binary Coded Decimal Value of Year's Tens Digit bits
Contains a value from 0 to 9.

bit 3-0 **YRONE<3:0>:** Binary Coded Decimal Value of Year's Ones Digit bits
Contains a value from 0 to 9.

Note 1: A write to the YEAR register is only allowed when RTCWREN = 1.

REGISTER 17-8: MONTH: MONTH VALUE REGISTER(1) (RTCVALH when RTCPTR<1:0> = 10)

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	MTHTEN0	MTHONE3	MTHONE2	MTHONE1	MTHONE0
bit 7							bit 0

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

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

bit 4 **MTHTEN0:** Binary Coded Decimal Value of Month's Tens Digit bit
Contains a value of 0 or 1.

bit 3-0 **MTHONE<3:0>:** Binary Coded Decimal Value of Month's Ones Digit bits
Contains a value from 0 to 9.

Note 1: A write to this register is only allowed when RTCWREN = 1.

REGISTER 20-11: SSPxCON3: MSSP CONTROL REGISTER 3 (I^2C SLAVE MODE)

R/HS/HC-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ACKTIM	PCIE	SCIE	BOEN	SDAHT	SBCDE	AHEN	DHEN
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 **ACKTIM:** Acknowledge Time Status bit
1 = Indicates the I^2C bus is in an Acknowledge sequence, set on 8th falling edge of SCL clock
0 = Not an Acknowledge sequence, cleared on 9th rising edge of SCL clock

bit 6 **PCIE:** Stop Condition Interrupt Enable bit(1)
1 = Enable interrupt on detection of Stop condition
0 = Stop detection interrupts are disabled

bit 5 **SCIE:** Start Condition Interrupt Enable bit(1)
1 = Enable interrupt on detection of Start or Restart conditions
0 = Start detection interrupts are disabled

bit 4 **BOEN:** Buffer Overwrite Enable bit
1 = SSPBUF is updated every time a new data byte is available, ignoring the SSPOV effect on updating the buffer
0 = SSPBUF is only updated when SSPOV is clear

bit 3 **SDAHT:** SDA Hold Time Selection bit
1 = Minimum of 300ns hold time on SDA after the falling edge of SCL
0 = Minimum of 100ns hold time on SDA after the falling edge of SCL

bit 2 **SBCDE:** Slave Mode Bus Collision Detect Enable bit
If, on the rising edge of SCL, SDA is sampled low when the module is outputting a high state, the BCLIF bit is set, and bus goes Idle.
1 = Enable slave bus collision interrupts
0 = Slave bus collision interrupts are disabled

bit 1 **AHEN:** Address Hold Enable bit
1 = Following the 8th falling edge of SCL for a matching received address byte; CKP bit of SSPxCON1 will be cleared and the SCL will be held low.
0 = Address holding is disabled

bit 0 **DHEN:** Data Hold Enable bit
1 = Following the 8th falling edge of SCL for a received data byte; slave hardware clears the CKP bit of SSPCON register and SCL is held low.
0 = Data holding is disabled

Note 1: This bit has no effect in Slave modes that Start and Stop condition detection is explicitly listed as enabled.

21.5 Infrared Support

This module provides support for two types of infrared USART port implementations:

- IrDA clock output to support an external IrDA encoder/decoder device
- Full implementation of the IrDa encoder and decoder as part of the USART logic

Since the 16x clock is required to perform the IrDA encoding, both by this module and the external transmitter, this feature only works in the 16x Baud Rate mode and is not available in the 4x mode.

21.5.1 EXTERNAL IrDA SUPPORT – IRDA CLOCK OUTPUT

The 16x Baud Clock is provided on the BCLK (Baud Clock) pin if the EUSARTx is enabled (SPEN = 1); it is configured for Asynchronous mode (SYNC = 0) when Clock Source Select is active (CSRC = 1). Note that the BCLK can be active in regular or IrDA mode (IREN bit is ignored).

21.5.1.1 BCLK Output

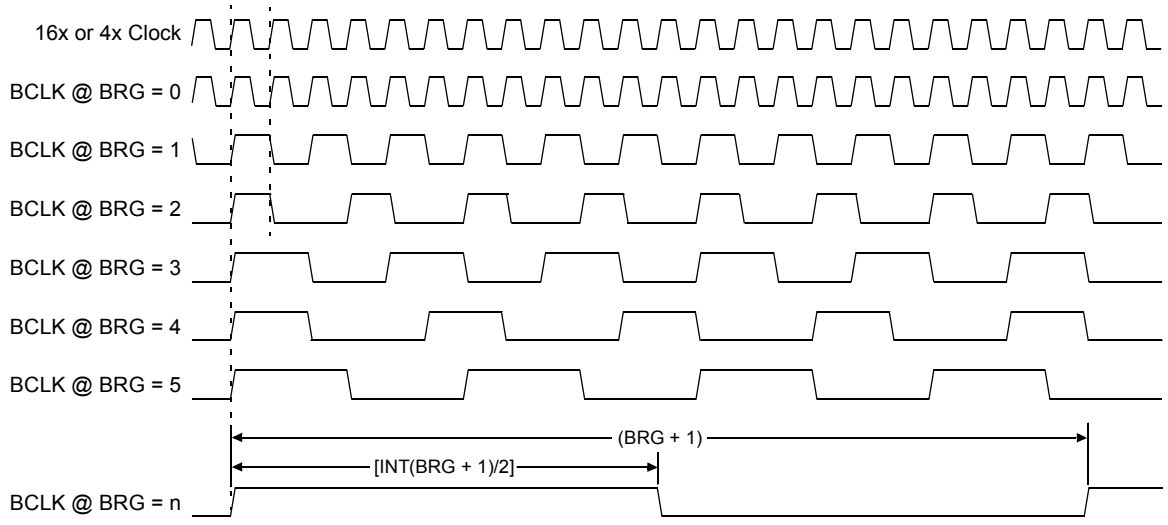
The timing of the Baud Clock (BCLK) output is independent of the 16x or 4x Baud Rate mode, resulting in the same output for a particular BRG value (since the 4x mode is four times faster, but has four times less pulses per period).

When the BCLK pin mode is active, the RXx Baud Rate Generator will be turned on, independent of a TXx or RXx operation. This will cause the RXx stream to synchronize to the already running RXx Baud Clock. This is acceptable only when BCLK is enabled for use.

The BCLK output goes inactive and stays low during Sleep mode.

The BCLK pin is taken over by the EUSARTx module and forced as an output, irrespective of port latch and TRIS latch bits. BCLK remains an output as long as USART is kept enabled in this mode.

FIGURE 21-14: BCLK OUTPUT vs. BRG PROGRAMMING



Note: The BCLK has 50% duty cycle only for odd BRG values. This is due to having all BCLK edges synchronous to the rising edge of the 16x/4x clock.

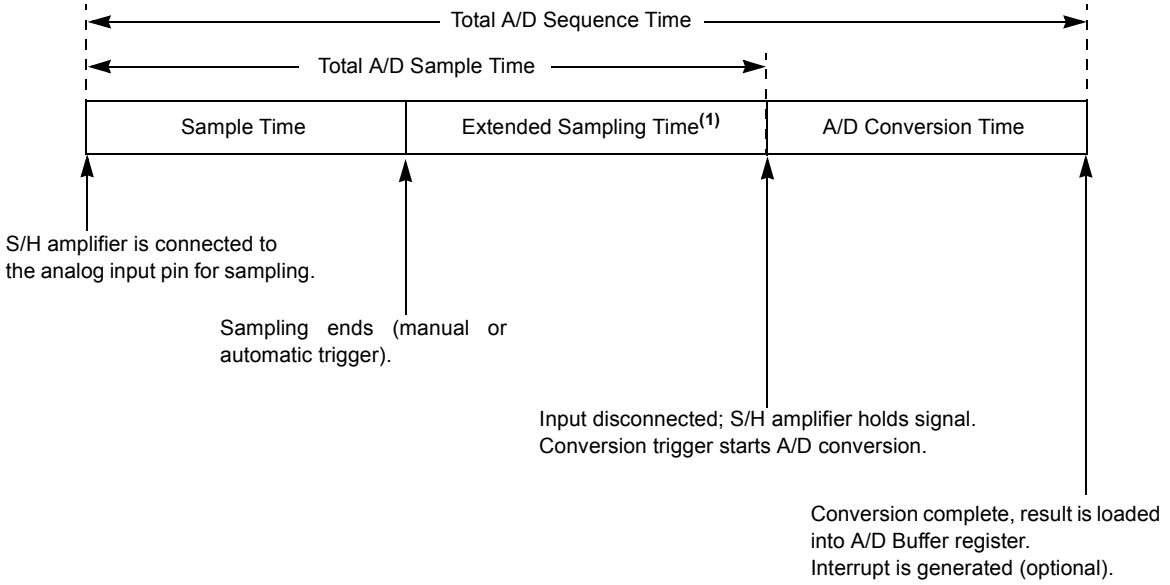
22.2 A/D Terminology and Conversion Sequence

Sample time is the time that the A/D module's S/H amplifier is connected to the analog input pin. The sample time may be started and ended automatically by the A/D Converter's hardware or under direct program control. There is a minimum sample time to ensure that the S/H amplifier will give sufficient accuracy for the A/D conversion.

The conversion trigger ends the sampling time and begins an A/D conversion or a repeating sequence. The conversion trigger sources can be taken from a variety of hardware sources or can be controlled directly in software. One of the conversion trigger options is an auto-conversion, which uses a counter and the A/D clock to set the time between auto-conversions. The Auto-Sample mode and auto-conversion trigger can be used together to provide continuous, automatic conversions without software intervention. When automatic sampling is used, an extended sampling interval is extended between the time the sampling ends and the conversion starts.

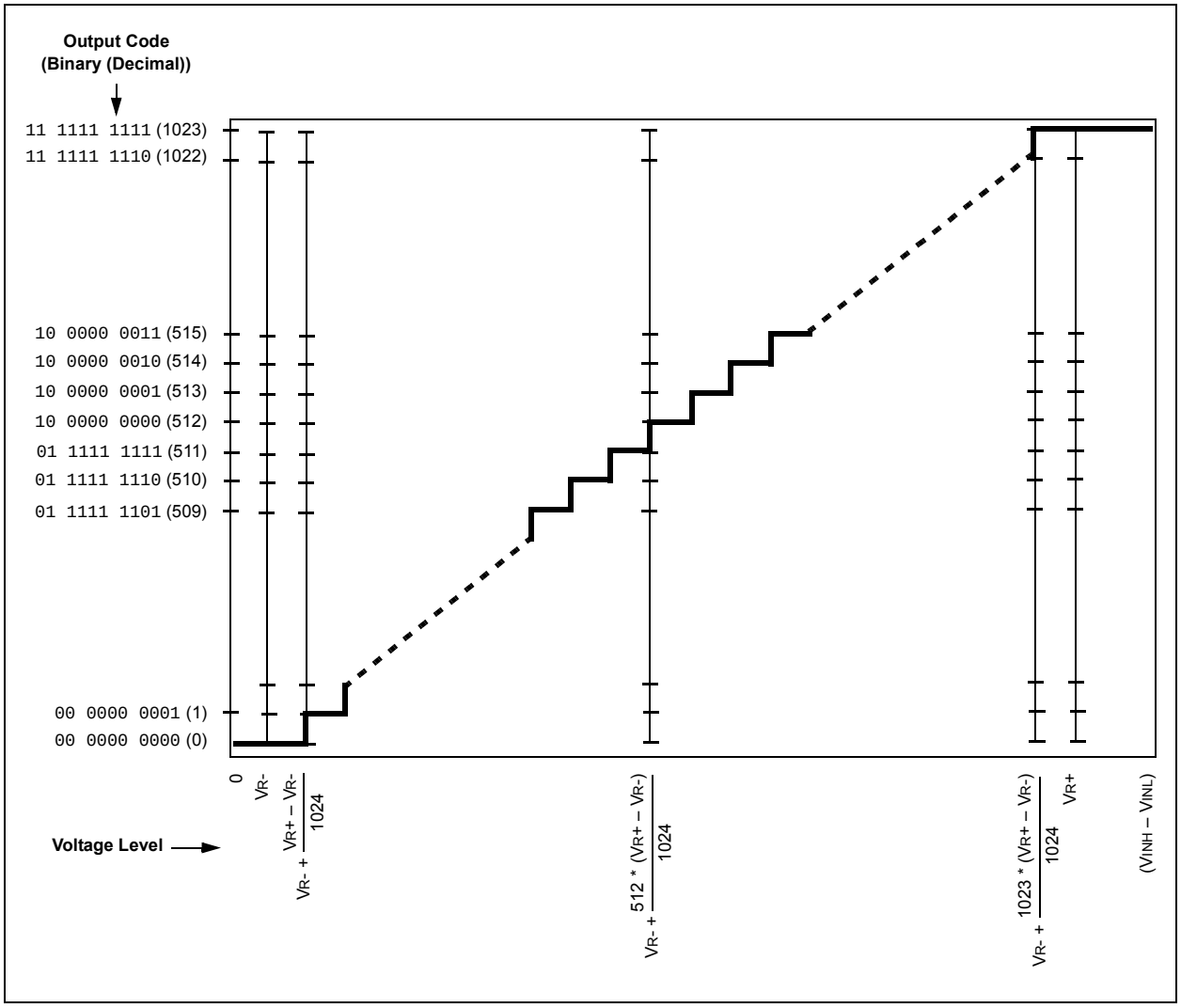
Conversion time is the time required for the A/D Converter to convert the voltage held by the S/H amplifier. An A/D conversion requires one A/D clock cycle (TAD) to convert each bit of the result, plus two additional clock cycles, or a total of 14 TAD cycles for a 12-bit conversion. When the conversion is complete, the result is loaded into one of the A/D result buffers. The S/H can be reconnected to the input pin and a CPU interrupt may be generated. The sum of the sample time(s) and the A/D conversion time provides the total A/D sequence time. Figure 22-2 shows the basic conversion sequence and the relationship between intervals.

FIGURE 22-2: A/D SAMPLE/CONVERT SEQUENCE



Note 1: In Automatic Sampling modes, Extended Sampling Time is added to the sequence when the value for the Auto-Sampling Time is greater than 0. Otherwise, sampling ends and conversion starts whenever the SAMP bit is cleared.

FIGURE 22-22: 10-BIT A/D TRANSFER FUNCTION



22.11 Operation During Sleep and Idle Modes

Sleep and Idle modes are useful for minimizing conversion noise because the digital activity of the CPU, buses and other peripherals is minimized.

22.11.1 CPU SLEEP MODE WITHOUT RC A/D CLOCK

When the device enters Sleep mode, all clock sources to the module are shut down and stay at logic '0'.

If Sleep occurs in the middle of a conversion, the conversion is aborted unless the A/D is clocked from its internal RC clock generator. The converter will not resume a partially completed conversion on exiting from Sleep mode.

Register contents are not affected by the device entering or leaving Sleep mode.

22.11.2 CPU SLEEP MODE WITH RC A/D CLOCK

The A/D module can operate during Sleep mode if the A/D clock source is set to the internal A/D RC oscillator (ADRC = 1). This eliminates digital switching noise from the conversion. When the conversion is completed, the DONE bit will be set and the result is loaded into the A/D Result Buffer n, ADCBUFn.

If the A/D interrupt is enabled (ADIE = 1), the device will wake-up from Sleep when the A/D interrupt occurs. Program execution will resume at the A/D Interrupt Service Routine (ISR). After the ISR completes execution will continue from the instruction after the SLEEP instruction that placed the device in Sleep mode.

If the A/D interrupt is not enabled, the A/D module will then be turned off, although the ADON bit will remain set.

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:

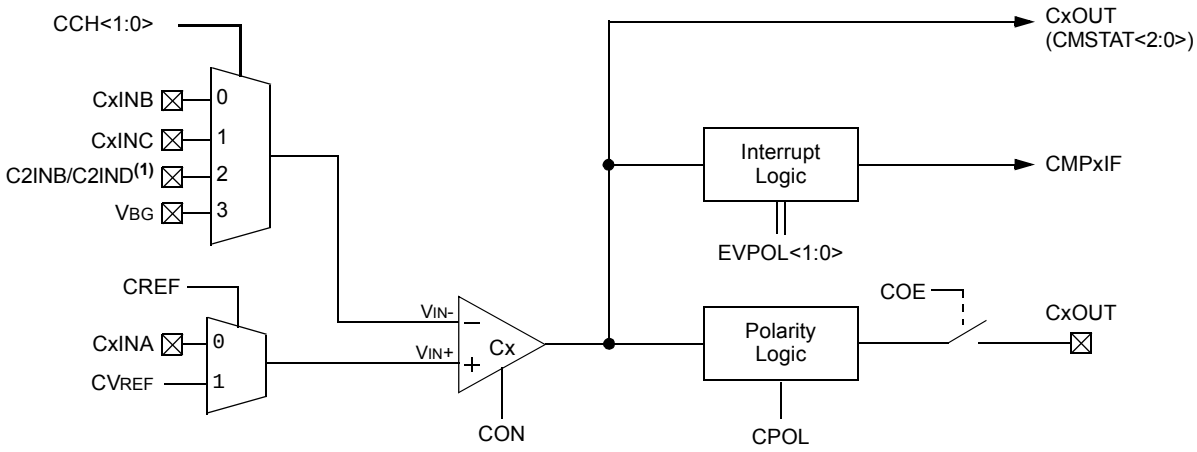
- Independent comparator control
- Programmable input configuration
- Output to both pin and register levels
- Programmable output polarity
- Independent interrupt generation for each comparator with configurable interrupt-on-change

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.

29.0 INSTRUCTION SET SUMMARY

The PIC18FXXJ94 of devices incorporates the standard set of 75 PIC18 core instructions, as well as an extended set of eight new instructions for the optimization of code that is recursive or that utilizes a software stack. The extended set is discussed later in this section.

29.1 Standard Instruction Set

The standard PIC18 MCU instruction set adds many enhancements to the previous PIC® MCU instruction sets, while maintaining an easy migration from these PIC MCU instruction sets. Most instructions are a single program memory word (16 bits), but there are four instructions that require two program memory locations.

Each single-word instruction is a 16-bit word divided into an opcode, which specifies the instruction type and one or more operands, which further specify the operation of the instruction.

The instruction set is highly orthogonal and is grouped into four basic categories:

- **Byte-oriented** operations
- **Bit-oriented** operations
- **Literal** operations
- **Control** operations

The PIC18 instruction set summary in Table 29-2 lists **byte-oriented**, **bit-oriented**, **literal** and **control** operations. Table 29-1 shows the opcode field descriptions.

Most **byte-oriented** instructions have three operands:

1. The file register (specified by ‘f’)
2. The destination of the result (specified by ‘d’)
3. The accessed memory (specified by ‘a’)

The file register designator, ‘f’, specifies which file register is to be used by the instruction. The destination designator, ‘d’, specifies where the result of the operation is to be placed. If ‘d’ is zero, the result is placed in the WREG register. If ‘d’ is one, the result is placed in the file register specified in the instruction.

All **bit-oriented** instructions have three operands:

1. The file register (specified by ‘f’)
2. The bit in the file register (specified by ‘b’)
3. The accessed memory (specified by ‘a’)

The bit field designator, ‘b’, selects the number of the bit affected by the operation, while the file register designator, ‘f’, represents the number of the file in which the bit is located.

The **literal** instructions may use some of the following operands:

- A literal value to be loaded into a file register (specified by ‘k’)
- The desired FSR register to load the literal value into (specified by ‘f’)
- No operand required (specified by ‘—’)

The **control** instructions may use some of the following operands:

- A program memory address (specified by ‘n’)
- The mode of the CALL or RETURN instructions (specified by ‘s’)
- The mode of the table read and table write instructions (specified by ‘m’)
- No operand required (specified by ‘—’)

All instructions are a single word, except for four double-word instructions. These instructions were made double-word to contain the required information in 32 bits. In the second word, the 4 MSbs are ‘1’s. If this second word is executed as an instruction (by itself), it will execute as a NOP.

All single-word instructions are executed in a single instruction cycle, unless a conditional test is true or the Program Counter is changed as a result of the instruction. In these cases, the execution takes two instruction cycles with the additional instruction cycle(s) executed as a NOP.

The double-word instructions execute in two instruction cycles.

One instruction cycle consists of four oscillator periods. Thus, for an oscillator frequency of 4 MHz, the normal instruction execution time is 1 μs. If a conditional test is true, or the Program Counter is changed as a result of an instruction, the instruction execution time is 2 μs. Two-word branch instructions (if true) would take 3 μs.

Figure 29-1 shows the general formats that the instructions can have. All examples use the convention ‘nnh’ to represent a hexadecimal number.

The Instruction Set Summary, shown in Table 29-2, lists the standard instructions recognized by the Microchip MPASM™ Assembler.

Section 29.1.1 “Standard Instruction Set” provides a description of each instruction.

TABLE 29-2: PIC18F97J94 FAMILY INSTRUCTION SET

Mnemonic, Operands	Description	Cycles	16-Bit Instruction Word MSb LSb	Status Affected	Notes
BYTE-ORIENTED OPERATIONS					
ADDWF f, d, a	Add WREG and f	1	0010 01da ffff ffff	C, DC, Z, OV, N	1, 2
ADDWFC f, d, a	Add WREG and Carry bit to f	1	0010 00da ffff ffff	C, DC, Z, OV, N	1, 2
ANDWF f, d, a	AND WREG with f	1	0001 01da ffff ffff	Z, N	1, 2
CLRF f, a	Clear f	1	0110 101a ffff ffff	Z	2
COMF f, d, a	Complement f	1	0001 11da ffff ffff	Z, N	1, 2
CPFSEQ f, a	Compare f with WREG, Skip =	1 (2 or 3)	0110 001a ffff ffff	None	4
CPFSGT f, a	Compare f with WREG, Skip >	1 (2 or 3)	0110 010a ffff ffff	None	4
CPFSLT f, a	Compare f with WREG, Skip <	1 (2 or 3)	0110 000a ffff ffff	None	1, 2
DECF f, d, a	Decrement f	1	0000 01da ffff ffff	C, DC, Z, OV, N	1, 2, 3, 4
DECFSZ f, d, a	Decrement f, Skip if 0	1 (2 or 3)	0010 11da ffff ffff	None	1, 2, 3, 4
DCFSNZ f, d, a	Decrement f, Skip if Not 0	1 (2 or 3)	0100 11da ffff ffff	None	1, 2
INCF f, d, a	Increment f	1	0010 10da ffff ffff	C, DC, Z, OV, N	1, 2, 3, 4
INCFSZ f, d, a	Increment f, Skip if 0	1 (2 or 3)	0011 11da ffff ffff	None	4
INFSNZ f, d, a	Increment f, Skip if Not 0	1 (2 or 3)	0100 10da ffff ffff	None	1, 2
IORWF f, d, a	Inclusive OR WREG with f	1	0001 00da ffff ffff	Z, N	1, 2
MOVF f, d, a	Move f	1	0101 00da ffff ffff	Z, N	1
MOVFF f_s, f_d	Move f_s (source) to 1st word f_d (destination) 2nd word	2	1100 ffff ffff ffff 1111 ffff ffff ffff	None	
MOVWF f, a	Move WREG to f	1	0110 111a ffff ffff	None	
MULWF f, a	Multiply WREG with f	1	0000 001a ffff ffff	None	1, 2
NEGF f, a	Negate f	1	0110 110a ffff ffff	C, DC, Z, OV, N	
RLCF f, d, a	Rotate Left f through Carry	1	0011 01da ffff ffff	C, Z, N	1, 2
RLNCF f, d, a	Rotate Left f (No Carry)	1	0100 01da ffff ffff	Z, N	
RRCF f, d, a	Rotate Right f through Carry	1	0011 00da ffff ffff	C, Z, N	
RRNCF f, d, a	Rotate Right f (No Carry)	1	0100 00da ffff ffff	Z, N	
SETF f, a	Set f	1	0110 100a ffff ffff	None	1, 2
SUBFWB f, d, a	Subtract f from WREG with Borrow	1	0101 01da ffff ffff	C, DC, Z, OV, N	
SUBWF f, d, a	Subtract WREG from f	1	0101 11da ffff ffff	C, DC, Z, OV, N	1, 2
SUBWFB f, d, a	Subtract WREG from f with Borrow	1	0101 10da ffff ffff	C, DC, Z, OV, N	
SWAPF f, d, a	Swap Nibbles in f	1	0011 10da ffff ffff	None	4
TSTFSZ f, a	Test f, Skip if 0	1 (2 or 3)	0110 011a ffff ffff	None	1, 2
XORWF f, d, a	Exclusive OR WREG with f	1	0001 10da ffff ffff	Z, N	

Note 1: When a PORT register is modified as a function of itself (e.g., `MOVF PORTB, 1, 0`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

2: If this instruction is executed on the TMR0 register (and where applicable, d = 1), the prescaler will be cleared if assigned.

3: If the Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a `NOP`.

4: Some instructions are two-word instructions. The second word of these instructions will be executed as a `NOP` unless the first word of the instruction retrieves the information embedded in these 16 bits. This ensures that all program memory locations have a valid instruction.