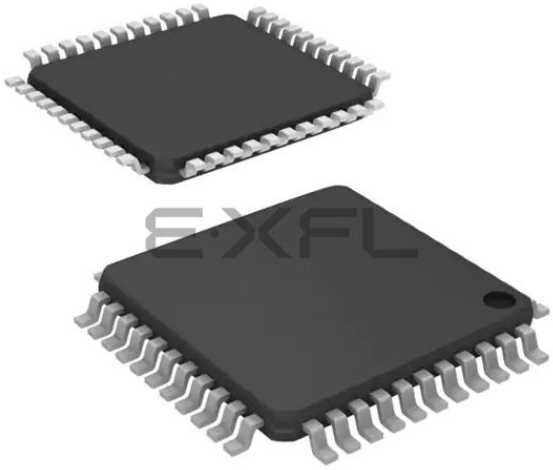


Welcome to E-XFL.COM

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	48MHz
Connectivity	I²C, LINbus, SPI, UART/USART, USB
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	34
Program Memory Size	64KB (32K x 16)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	3.8K x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 2.75V
Data Converters	A/D 13x10b/12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-TQFP
Supplier Device Package	44-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18lf46j53-i-pt

1.0 DEVICE OVERVIEW

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

- PIC18F26J53
- PIC18F27J53
- PIC18F46J53
- PIC18F47J53
- PIC18LF26J53
- PIC18LF27J53
- PIC18LF46J53
- PIC18LF47J53

This family introduces a new line of low-voltage Universal Serial Bus (USB) microcontrollers with 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 PIC18F47J53 family a logical choice for many high-performance applications, where cost is a primary consideration.

1.1 Core Features

1.1.1 XLP TECHNOLOGY

All of the devices in the PIC18F47J53 family incorporate a range of features that can significantly reduce power consumption during operation. Key features are:

- **Alternate Run Modes:** By clocking the controller from the Timer1 source or the internal RC oscillator, power consumption during code execution can be reduced by as much as 90%.
- **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, to as little as 4% of normal operational requirements.
- **On-the-Fly Mode Switching:** The power-managed modes are invoked by user code during operation, allowing the users to incorporate power-saving ideas into their application's software design.
- **Deep Sleep:** The 2.5V internal core voltage regulator on F parts can be shutdown to cut power consumption to as low as 15 nA (typical). Certain features can remain operating during Deep Sleep, such as the Real-Time Clock Calendar.
- **Ultra Low Power Wake-Up:** Waking from Sleep or Deep Sleep modes after a period of time can be done without an oscillator/clock source, saving power for applications requiring periodic activity.

1.1.2 UNIVERSAL SERIAL BUS (USB)

Devices in the PIC18F47J53 family incorporate a fully-featured USB communications module with a built-in transceiver that is compliant with the USB Specification Revision 2.0. The module supports both low-speed and full-speed communication for all supported data transfer types.

1.1.3 OSCILLATOR OPTIONS AND FEATURES

All of the devices in the PIC18F47J53 family offer five different oscillator options, allowing users a range of choices in developing application hardware. These include:

- Two Crystal modes, using crystals or ceramic resonators.
- Two External Clock modes, offering the option of a divide-by-4 clock output.
- An internal oscillator block, which provides an 8 MHz clock and an INTRC source (approximately 31 kHz, stable over temperature and V_{DD}), as well as a range of six user-selectable clock frequencies, between 125 kHz to 4 MHz, for a total of eight clock frequencies. This option frees an oscillator pin for use as an additional general purpose I/O.
- A Phase Lock Loop (PLL) frequency multiplier available to the high-speed crystal, and external and internal oscillators, providing a clock speed up to 48 MHz.
- Dual clock operation, allowing the USB module to run from a high-frequency oscillator while the rest of the microcontroller is clocked at a different frequency.

The internal oscillator block provides a stable reference source that gives the PIC18F47J53 family additional features for robust operation:

- **Fail-Safe Clock Monitor:** 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:** This option allows the internal oscillator to serve as the clock source from Power-on Reset (POR), or wake-up from Sleep mode, until the primary clock source is available.

1.1.4 EXPANDED MEMORY

The PIC18F47J53 family provides ample room for application code, from 64 Kbytes to 128 Kbytes of code space. The Flash cells for program memory are rated to last in excess of 10000 erase/write cycles. Data retention without refresh is conservatively estimated to be greater than 20 years.

The Flash program memory is readable and writable during normal operation. The PIC18F47J53 family also provides plenty of room for dynamic application data with up to 3.8 Kbytes of data RAM.

TABLE 1-3: PIC18F2XJ53 PINOUT I/O DESCRIPTIONS

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	28-SPDIP/ SSOP/ SOIC	28-QFN			
$\overline{MCLR}$	1(2)	26(2)	I	ST	Master Clear (Reset) input. This pin is an active-low Reset to the device.
OSC1/CLKI/RA7	9	6			Oscillator crystal or external clock input.
OSC1			I	ST	Oscillator crystal input or external clock source input. ST buffer when configured in RC mode; CMOS otherwise. Main oscillator input connection.
CLKI			I	CMOS	External clock source input; always associated with pin function OSC1 (see related OSC1/CLKI pins).
RA7(1)			I/O	TTL/DIG	Digital I/O.
OSC2/CLKO/RA6	10	7			Oscillator crystal or clock output.
OSC2			O	—	Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode.
CLKO			O	DIG	Main oscillator feedback output connection. In RC mode, OSC2 pin outputs CLKO, which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate.
RA6(1)			I/O	TTL/DIG	Digital I/O.

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

Note 1: RA7 and RA6 will be disabled if OSC1 and OSC2 are used for the clock function.

2: 5.5V tolerant.

TABLE 1-3: PIC18F2XJ53 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	28-SPDIP/ SSOP/ SOIC	28-QFN			
					PORTA is a bidirectional I/O port.
RA0/AN0/C1INA/ULPWU/RP0	2	27			
RA0			I/O	TTL/DIG	Digital I/O.
AN0			I	Analog	Analog Input 0.
C1INA			I	Analog	Comparator 1 Input A.
ULPWU			I	Analog	Ultra low-power wake-up input.
RP0			I/O	ST/DIG	Remappable Peripheral Pin 0 input/output.
RA1/AN1/C2INA/V_{BG}/RP1	3	28			
RA1			I/O	TTL/DIG	Digital I/O.
AN1			O	Analog	Analog Input 1.
C2INA			I	Analog	Comparator 2 Input A.
V_{BG}			O	Analog	Band Gap Reference Voltage (V_{BG}) output.
RP1			I/O	ST/DIG	Remappable Peripheral Pin 1 input/output.
RA2/AN2/C2INB/C1IND/ C3INB/V_{REF-}/CV_{REF}	4	1			
RA2			I/O	TTL/DIG	Digital I/O.
AN2			I	Analog	Analog Input 2.
C2INB			I	Analog	Comparator 2 Input B.
C1IND			I	Analog	Comparator 1 Input D.
C3INB			I	Analog	Comparator 3 Input B.
V_{REF-}			O	Analog	A/D reference voltage (low) input.
CV_{REF}			I	Analog	Comparator reference voltage output.
RA3/AN3/C1INB/V_{REF+}	5	2			
RA3			I/O	TTL/DIG	Digital I/O.
AN3			I	Analog	Analog Input 3.
C1INB			I	Analog	Comparator 1 Input B.
V_{REF+}			I	Analog	A/D reference voltage (high) input.
RA5/AN4/C1INC/$\overline{SS1}$/ HLVDIN/RCV/RP2	7	4			
RA5			I/O	TTL/DIG	Digital I/O.
AN4			I	Analog	Analog Input 4.
C1INC			I	Analog	Comparator 1 Input C.
$\overline{SS1}$			I	TTL	SPI slave select input.
HLVDIN			I	Analog	High/Low-Voltage Detect input.
RCV			I	Analog	External USB transceiver RCV input.
RP2			I/O	ST/DIG	Remappable Peripheral Pin 2 input/output.
RA6(1)					See the OSC2/CLKO/RA6 pin.
RA7(1)					See the OSC1/CLKI/RA7 pin.

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

Note 1: RA7 and RA6 will be disabled if OSC1 and OSC2 are used for the clock function.

2: 5.5V tolerant.

4.0 LOW-POWER MODES

The PIC18F47J53 family devices can manage power consumption through clocking to the CPU and the peripherals. In general, reducing the clock frequency and number of circuits being clocked reduces power consumption.

For managing power in an application, the primary modes of operation are:

- Run Mode
- Idle Mode
- Sleep Mode
- Deep Sleep Mode

Additionally, there is an Ultra Low-Power Wake-up (ULPWU) mode for generating an interrupt-on-change on RA0.

These modes define which portions of the device are clocked and at what speed.

- The Run and Idle modes can use any of the three available clock sources (primary, secondary or internal oscillator blocks).
- The Sleep mode does not use a clock source.

The ULPWU mode on RA0 allows a slow falling voltage to generate an interrupt-on-change on RA0 without excess current consumption. See **Section 4.7 "Ultra Low-Power Wake-up"**.

The power-managed modes include several power-saving features offered on previous PIC® devices, such as clock switching, ULPWU and Sleep mode. In addition, the PIC18F47J53 family devices add a new power-managed Deep Sleep mode.

4.1 Selecting Power-Managed Modes

Selecting a power-managed mode requires these decisions:

- Will the CPU be clocked?
- If so, which clock source will be used?

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

4.1.1 CLOCK SOURCES

The SCS<1:0> bits allow the selection of one of three clock sources for power-managed modes. They are:

- Primary clock source – Defined by the FOSC<2:0> Configuration bits
- Timer1 clock – Provided by the secondary oscillator
- Postscaled internal clock – Derived from the internal oscillator block

4.1.2 ENTERING POWER-MANAGED MODES

Switching from one clock source to another begins by loading the OSCCON register. The SCS<1:0> bits select the clock source.

Changing these bits causes an immediate switch to the new clock source, assuming that it is running. The switch also may be subject to clock transition delays. These delays are discussed in **Section 4.1.3 "Clock Transitions and Status Indicators"** and subsequent sections.

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

Depending on the current mode and the mode being switched to, a change to a power-managed mode does not always require setting all of these bits. Many transitions may be done by changing the oscillator select bits, the IDLEN bit, or the DSEN bit prior to issuing a SLEEP instruction.

If the IDLEN and DSEN bits are already configured correctly, it may only be necessary to perform a SLEEP instruction to switch to the desired mode.

6.6.3 MAPPING THE ACCESS BANK IN INDEXED LITERAL OFFSET MODE

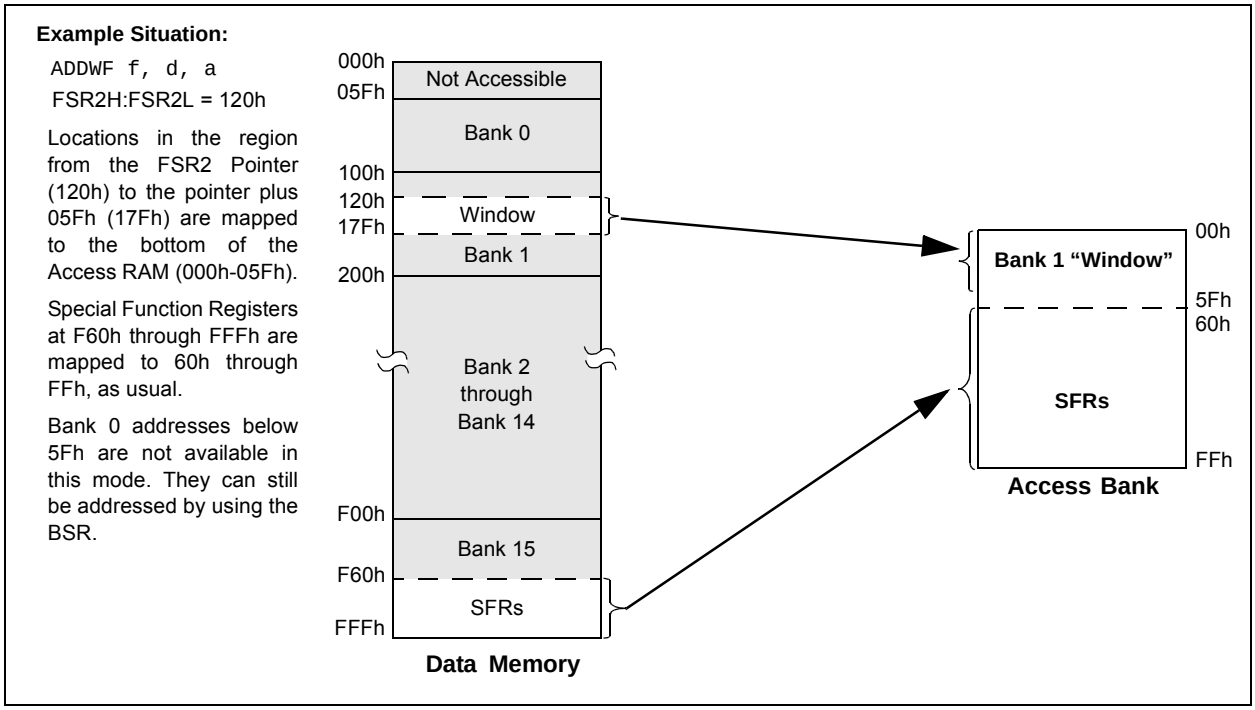
The use of Indexed Literal Offset Addressing mode effectively changes how the lower part of Access RAM (00h to 5Fh) is mapped. Rather than containing just the contents of the bottom part of Bank 0, this mode maps the contents from Bank 0 and a user-defined "window" that can be located anywhere in the data memory space. The value of FSR2 establishes the lower boundary of the addresses mapped to the window, while the upper boundary is defined by FSR2 plus 95 (5Fh). Addresses in the Access RAM above 5Fh are mapped as previously described (see **Section 6.3.3 "Access Bank"**). Figure 6-10 provides an example of Access Bank remapping in this addressing mode.

Remapping of the Access Bank applies *only* to operations using the Indexed Literal Offset mode. Operations that use the BSR (Access RAM bit is '1') will continue to use Direct Addressing as before. Any Indirect or Indexed Addressing operation that explicitly uses any of the indirect file operands (including FSR2) will continue to operate as standard Indirect Addressing. Any instruction that uses the Access Bank, but includes a register address of greater than 05Fh, will use Direct Addressing and the normal Access Bank map.

6.6.4 BSR IN INDEXED LITERAL OFFSET MODE

Although the Access Bank is remapped when the extended instruction set is enabled, the operation of the BSR remains unchanged. Direct Addressing, using the BSR to select the data memory bank, operates in the same manner as previously described.

FIGURE 6-10: REMAPPING THE ACCESS BANK WITH INDEXED LITERAL OFFSET ADDRESSING



11.1.2 DATA REGISTERS

The PMP module uses eight registers for transferring data into and out of the microcontroller. They are arranged as four pairs to allow the option of 16-bit data operations:

- PMDIN1H and PMDIN1L
- PMDIN2H and PMDIN2L
- PMADDRH/PMDOUT1H and PMADDRL/PMDOUT1L
- PMDOUT2H and PMDOUT2L

The PMDIN1 registers are used for incoming data in Slave modes, and both input and output data in Master modes. The PMDIN2 registers are used for buffering input data in select Slave modes.

The PMADDR/PMDOUT1 registers are actually a single register pair; the name and function are dictated by the module's operating mode. In Master modes, the registers function as the PMADDRH and PMADDRL registers, and contain the address of any incoming or outgoing data. In Slave modes, the registers function as PMDOUT1H and PMDOUT1L, and are used for outgoing data.

PMADDRH differs from PMADDRL in that it can also have limited PMP control functions. When the module is operating in select Master mode configurations, the upper two bits of the register can be used to determine the operation of chip select signals. If these are not used, PMADDR simply functions to hold the upper 8 bits of the address. Register 11-9 provides the function of the individual bits in PMADDRH.

The PMDOUT2H and PMDOUT2L registers are only used in Buffered Slave modes and serve as a buffer for outgoing data.

11.1.3 PAD CONFIGURATION CONTROL REGISTER

In addition to the module level configuration options, the PMP module can also be configured at the I/O pin for electrical operation. This option allows users to select either the normal Schmitt Trigger input buffer on digital I/O pins shared with the PMP, or use TTL level compatible buffers instead. Buffer configuration is controlled by the PMPTTL bit in the PADCFG1 register.

11.2.2 WRITE TO SLAVE PORT

When chip select is active and a write strobe occurs (PMCSx = 1 and PMWR = 1), the data from PMD<7:0> is captured into the lower PMDIN1L register. The PMPIF and IBF flag bits are set when the write ends.The timing for the control signals in Write mode is displayed in Figure 11-3. The polarity of the control signals are configurable.

11.2.3 READ FROM SLAVE PORT

When chip select is active and a read strobe occurs (PMCSx = 1 and PMRD = 1), the data from the PMDOUT1L register (PMDOUT1L<7:0>) is presented on to PMD<7:0>. Figure 11-4 provides the timing for the control signals in Read mode.

FIGURE 11-3: PARALLEL SLAVE PORT WRITE WAVEFORMS

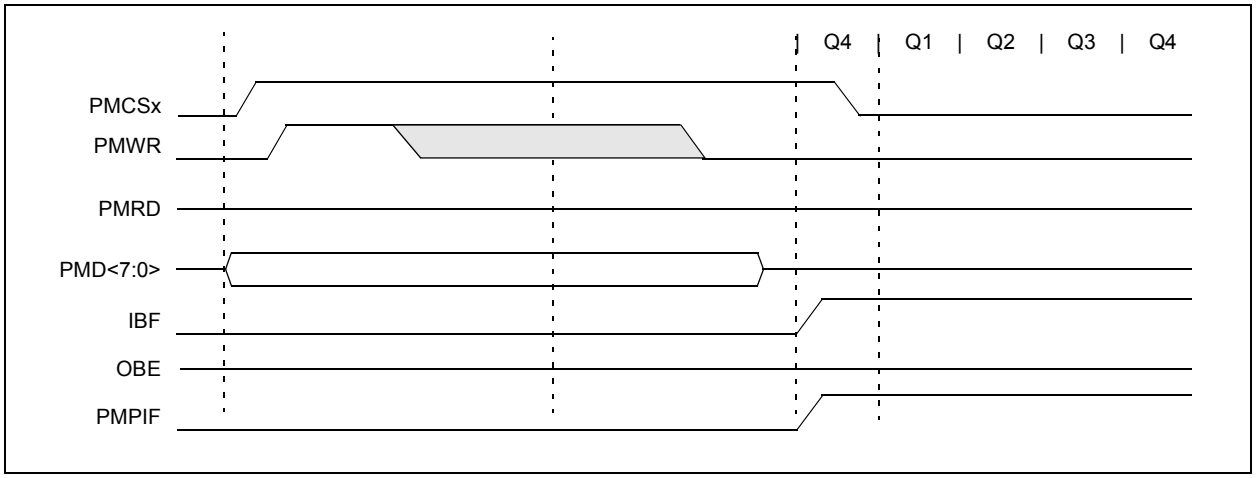
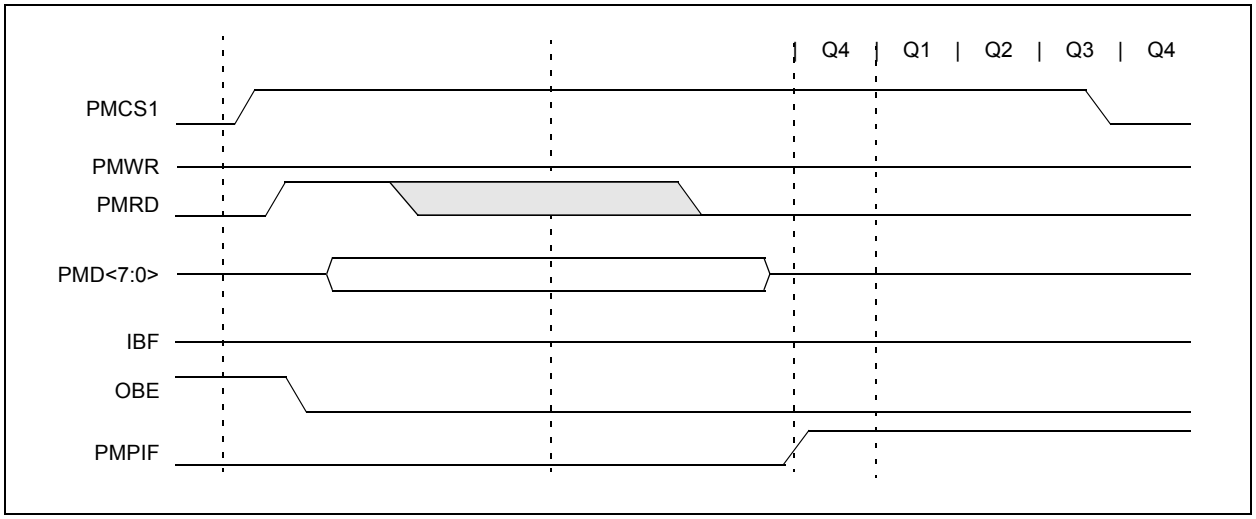


FIGURE 11-4: PARALLEL SLAVE PORT READ WAVEFORMS



11.2.4 BUFFERED PARALLEL SLAVE PORT MODE

Buffered Parallel Slave Port mode is functionally identical to the legacy PSP mode with one exception, the implementation of 4-level read and write buffers. Buffered PSP mode is enabled by setting the INCM bits in the PMMODEH register. If the INCM<1:0> bits are set to '11', the PMP module will act as the buffered PSP.

When the Buffered PSP mode is active, the PMDIN1L, PMDIN1H, PMDIN2L and PMDIN2H registers become the write buffers and the PMDOUT1L, PMDOUT1H, PMDOUT2L and PMDOUT2H registers become the read buffers. Buffers are numbered, 0 through 3, starting with the lower byte of PMDIN1L to PMDIN2H as the read buffers and PMDOUT1L to PMDOUT2H as the write buffers.

11.2.4.1 READ FROM SLAVE PORT

For read operations, the bytes will be sent out sequentially, starting with Buffer 0 (PMDOUT1L<7:0>) and ending with Buffer 3 (PMDOUT2H<7:0>) for every read strobe. The module maintains an internal pointer to keep track of which buffer is to be read. Each buffer has a corresponding read status bit, OBxE, in the PMSTATL register. This bit is cleared when a buffer contains data that has not been written to the bus, and is set when data is written to the bus. If the current buffer location being read from is empty, a buffer underflow is generated and the Buffer Overflow Flag bit, OBUF, is set. If all four OBxE status bits are set, then the Output Buffer Empty flag (OBE) will also be set.

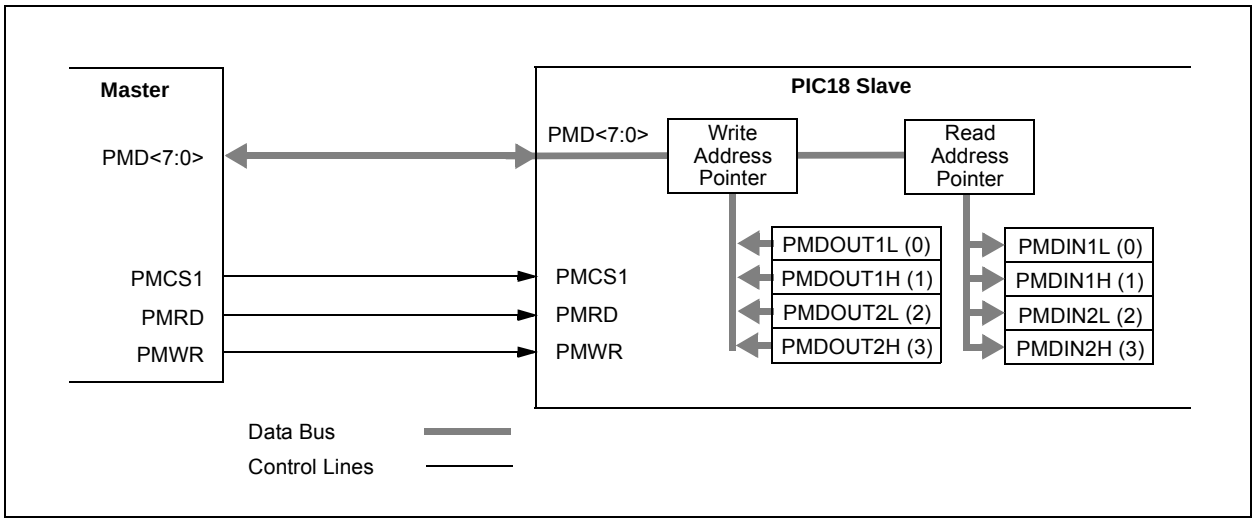
11.2.4.2 WRITE TO SLAVE PORT

For write operations, the data has to be stored sequentially, starting with Buffer 0 (PMDIN1L<7:0>) and ending with Buffer 3 (PMDIN2H<7:0>). As with read operations, the module maintains an internal pointer to the buffer that is to be written next.

The input buffers have their own write status bits: IBxF in the PMSTATH register. The bit is set when the buffer contains unread incoming data and cleared when the data has been read. The flag bit is set on the write strobe. If a write occurs on a buffer when its associated IBxF bit is set, the Buffer Overflow flag, IBOV, is set; any incoming data in the buffer will be lost. If all four IBxF flags are set, the Input Buffer Full Flag (IBF) is set.

In Buffered Slave mode, the module can be configured to generate an interrupt on every read or write strobe (IRQM<1:0> = 01). It can be configured to generate an interrupt on a read from Read Buffer 3 or a write to Write Buffer 3, which is essentially an interrupt every fourth read or write strobe (RQM<1:0> = 11). When interrupting every fourth byte for input data, all input buffer registers should be read to clear the IBxF flags. If these flags are not cleared, then there is a risk of hitting an overflow condition.

FIGURE 11-5: PARALLEL MASTER/SLAVE CONNECTION BUFFERED EXAMPLE



12.3 Prescaler

An 8-bit counter is available as a prescaler for the Timer0 module. The prescaler is not directly readable or writable. Its value is set by the PSA and T0PS<2:0> bits (T0CON<3:0>), which determine the prescaler assignment and prescale ratio.

Clearing the PSA bit assigns the prescaler to the Timer0 module. When it is assigned, prescale values from 1:2 through 1:256, in power-of-2 increments, are selectable.

When assigned to the Timer0 module, all instructions writing to the TMR0 register (e.g., `CLRF TMR0`, `MOVWF TMR0`, `BSF TMR0`, etc.) clear the prescaler count.

Note:	Writing to TMR0 when the prescaler is assigned to Timer0 will clear the prescaler count but will not change the prescaler assignment.

12.3.1 SWITCHING PRESCALER ASSIGNMENT

The prescaler assignment is fully under software control and can be changed “on-the-fly” during program execution.

12.4 Timer0 Interrupt

The TMR0 interrupt is generated when the TMR0 register overflows from FFh to 00h in 8-bit mode, or from FFFFh to 0000h in 16-bit mode. This overflow sets the TMR0IF flag bit. The interrupt can be masked by clearing the TMR0IE bit (INTCON<5>). Before re-enabling the interrupt, the TMR0IF bit must be cleared in software by the Interrupt Service Routine (ISR).

Since Timer0 is shut down in Sleep mode, the TMR0 interrupt cannot awaken the processor from Sleep.

TABLE 12-1: REGISTERS ASSOCIATED WITH TIMER0

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TMR0L	Timer0 Register Low Byte							
TMR0H	Timer0 Register High Byte							
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0

Legend: Shaded cells are not used by Timer0.

REGISTER 17-18: ALRMMIN: ALARM MINUTES VALUE REGISTER

U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	MINTEN2	MINTEN1	MINTEN0	MINONE3	MINONE2	MINONE1	MINONE0
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 **Unimplemented:** Read as ‘0’

bit 6-4 **MINTEN<2:0>:** Binary Coded Decimal Value of Minute’s Tens Digit bits
Contains a value from 0 to 5.

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

REGISTER 17-19: ALRMSEC: ALARM SECONDS VALUE REGISTER

U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	SECTEN2	SECTEN1	SECTEN0	SECONE3	SECONE2	SECONE1	SECONE0
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 **Unimplemented:** Read as ‘0’

bit 6-4 **SECTEN<2:0>:** Binary Coded Decimal Value of Second’s Tens Digit bits
Contains a value from 0 to 5.

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

When a shutdown event occurs, two things happen:

- The ECCPxASE bit is set to '1'. The ECCPxASE will remain set until cleared in firmware or an auto-restart occurs. (See **Section 19.4.5 "Auto-Restart Mode"**.)
- The enabled PWM pins are asynchronously placed in their shutdown states. The PWM output pins are grouped into pairs (PxA/PxC) and (PxB/PxD). The state of each pin pair is determined by the PSSxAC and PSSxBD bits (ECCPxAS<3:0>).

Each pin pair may be placed into one of three states:

- Drive logic '1'
- Drive logic '0'
- Tri-state (high-impedance)

REGISTER 19-4: ECCPxAS: ECCP1/2/3 AUTO-SHUTDOWN CONTROL REGISTER (1, ACCESS FBEh; 2, FB8h; 3, BANKED F19h)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ECCPxASE	ECCPxAS2	ECCPxAS1	ECCPxAS0	PSSxAC1	PSSxAC0	PSSxBD1	PSSxBD0
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 **ECCPxASE:** ECCP Auto-Shutdown Event Status bit
1 = A shutdown event has occurred; ECCP outputs are in a shutdown state
0 = ECCP outputs are operating

bit 6-4 **ECCPxAS<2:0>:** ECCP Auto-Shutdown Source Select bits
000 = Auto-shutdown is disabled
001 = Comparator, C1OUT, output is high
010 = Comparator, C2OUT, output is high
011 = Either comparator, C1OUT or C2OUT, is high
100 = VIL on FLT0 pin
101 = VIL on FLT0 pin or comparator, C1OUT, output is high
110 = VIL on FLT0 pin or comparator, C2OUT, output is high
111 = VIL on FLT0 pin or comparator, C1OUT, or comparator, C2OUT, is high

bit 3-2 **PSSxAC<1:0>:** PxA and PxC Pins Shutdown State Control bits
00 = Drive pins, PxA and PxC, to '0'
01 = Drive pins, PxA and PxC, to '1'
1x = PxA and PxC pins tri-state

bit 1-0 **PSSxBD<1:0>:** PxB and PxD Pins Shutdown State Control bits
00 = Drive pins, PxB and PxD, to '0'
01 = Drive pins, PxB and PxD, to '1'
1x = PxB and PxD pins tri-state

Note 1: The auto-shutdown condition is a level-based signal, not an edge-based signal. As long as the level is present, the auto-shutdown will persist.
2: Writing to the ECCPxASE bit is disabled while an auto-shutdown condition persists.
3: Once the auto-shutdown condition has been removed and the PWM restarted (either through firmware or auto-restart), the PWM signal will always restart at the beginning of the next PWM period.

20.2 Control Registers

Each MSSP module has three associated control registers. These include a status register (SSPxSTAT) and two control registers (SSPxCON1 and SSPxCON2). The use of these registers and their individual Configuration bits differs significantly depending on whether the MSSP module is operated in SPI or I^2C mode.

Additional details are provided under the individual sections.

Note:	In devices with more than one MSSP module, it is very important to pay close attention to the SSPxCON register names. SSP1CON1 and SSP1CON2 control different operational aspects of the same module, while SSP1CON1 and SSP2CON1 control the same features for two different modules.

20.3 SPI Mode

The SPI mode allows 8 bits of data to be synchronously transmitted and received simultaneously. All four modes of SPI are supported.

When MSSP2 is used in SPI mode, it can optionally be configured to work with the SPI DMA submodule described in **Section 20.4 “SPI DMA Module”**.

To accomplish communication, typically three pins are used:

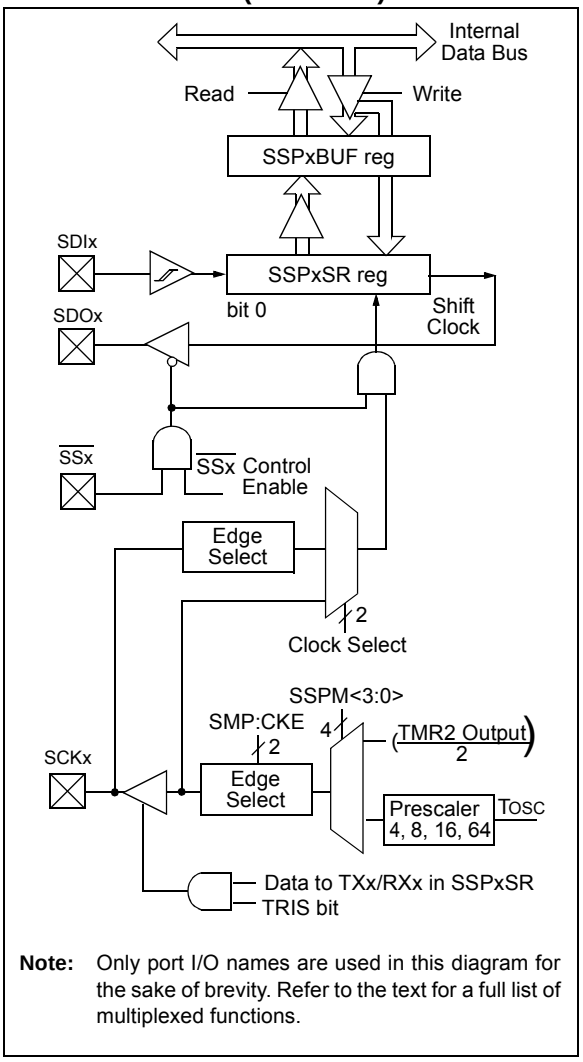
- Serial Data Out (SDOx) – RC7/CCP10/RX1/DT1/SDO1/RP18 or SDO2/Remappable
- Serial Data In (SDIx) – RB5/CCP5/KBI1/SDI1/SDA1/RP8 or SDI2/Remappable
- Serial Clock (SCKx) – RB4/CCP4/KBI0/SCK1/SCL1/RP7 or SCK2/Remappable

Additionally, a fourth pin may be used when in a Slave mode of operation:

- Slave Select ($\overline{SSx}$) – RA5/AN4/C1INC/$\overline{SS1}$/HLVDIN/RCV/RP2 or $\overline{SS2}$/Remappable

Figure 20-1 depicts the block diagram of the MSSP module when operating in SPI mode.

FIGURE 20-1: MSSPx BLOCK DIAGRAM (SPI MODE)



Note: Only port I/O names are used in this diagram for the sake of brevity. Refer to the text for a full list of multiplexed functions.

FIGURE 20-23: I²C MASTER MODE WAVEFORM (TRANSMISSION, 7-BIT OR 10-BIT ADDRESS)

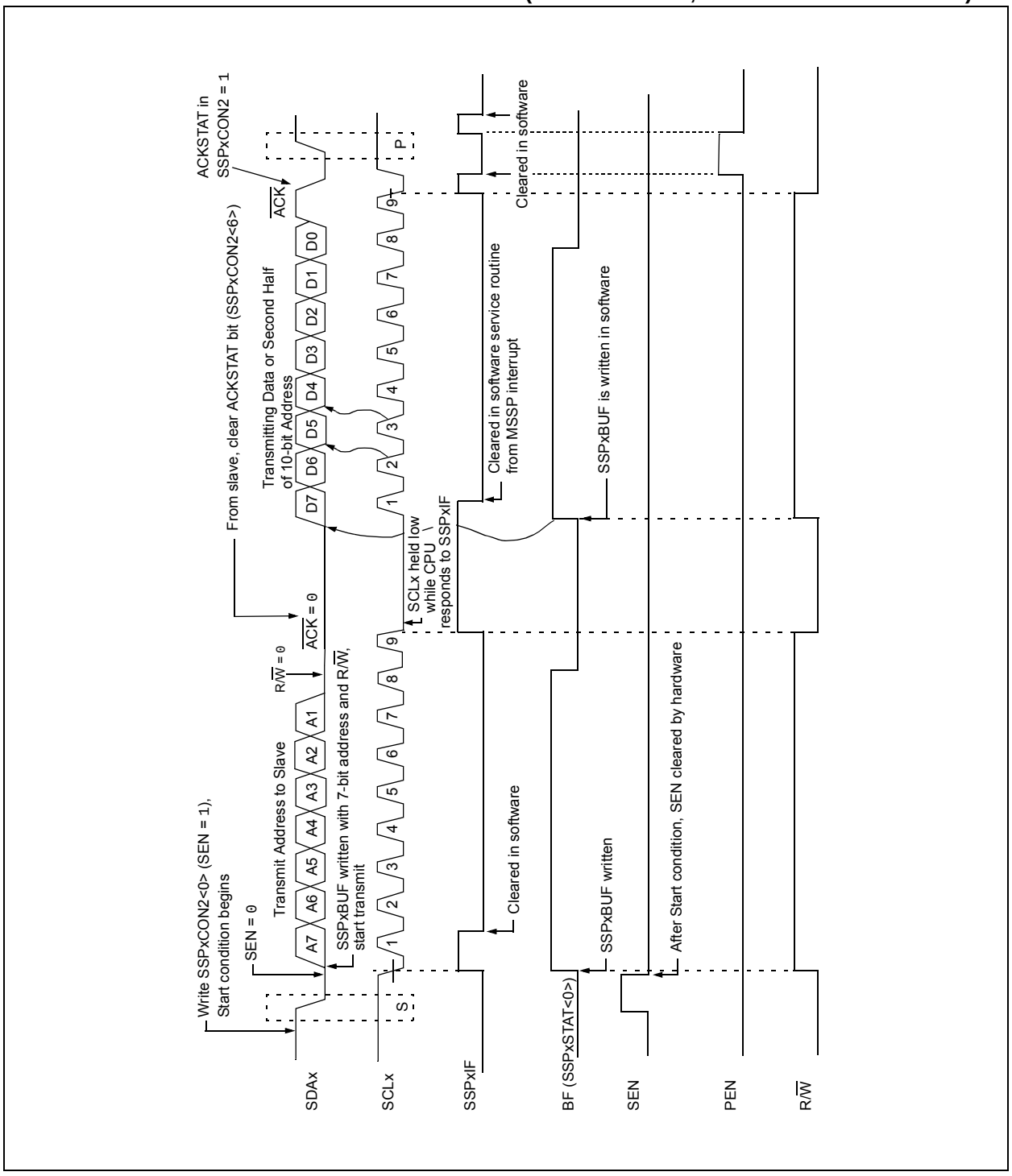


TABLE 21-3: BAUD RATES FOR ASYNCHRONOUS MODES (CONTINUED)

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 1											
	FOSC = 40.000 MHz			FOSC = 20.000 MHz			FOSC = 10.000 MHz			FOSC = 8.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.00	8332	0.300	0.02	4165	0.300	0.02	2082	0.300	-0.04	1665
1.2	1.200	0.02	2082	1.200	-0.03	1041	1.200	-0.03	520	1.201	-0.16	415
2.4	2.402	0.06	1040	2.399	-0.03	520	2.404	0.16	259	2.403	-0.16	207
9.6	9.615	0.16	259	9.615	0.16	129	9.615	0.16	64	9.615	-0.16	51
19.2	19.231	0.16	129	19.231	0.16	64	19.531	1.73	31	19.230	-0.16	25
57.6	58.140	0.94	42	56.818	-1.36	21	56.818	-1.36	10	55.555	3.55	8
115.2	113.636	-1.36	21	113.636	-1.36	10	125.000	8.51	4	—	—	—

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 1								
	FOSC = 4.000 MHz			FOSC = 2.000 MHz			FOSC = 1.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.04	832	0.300	-0.16	415	0.300	-0.16	207
1.2	1.202	0.16	207	1.201	-0.16	103	1.201	-0.16	51
2.4	2.404	0.16	103	2.403	-0.16	51	2.403	-0.16	25
9.6	9.615	0.16	25	9.615	-0.16	12	—	—	—
19.2	19.231	0.16	12	—	—	—	—	—	—
57.6	62.500	8.51	3	—	—	—	—	—	—
115.2	125.000	8.51	1	—	—	—	—	—	—

BAUD RATE (K)	SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1											
	FOSC = 40.000 MHz			FOSC = 20.000 MHz			FOSC = 10.000 MHz			FOSC = 8.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.00	33332	0.300	0.00	16665	0.300	0.00	8332	0.300	-0.01	6665
1.2	1.200	0.00	8332	1.200	0.02	4165	1.200	0.02	2082	1.200	-0.04	1665
2.4	2.400	0.02	4165	2.400	0.02	2082	2.402	0.06	1040	2.400	-0.04	832
9.6	9.606	0.06	1040	9.596	-0.03	520	9.615	0.16	259	9.615	-0.16	207
19.2	19.193	-0.03	520	19.231	0.16	259	19.231	0.16	129	19.230	-0.16	103
57.6	57.803	0.35	172	57.471	-0.22	86	58.140	0.94	42	57.142	0.79	34
115.2	114.943	-0.22	86	116.279	0.94	42	113.636	-1.36	21	117.647	-2.12	16

BAUD RATE (K)	SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1								
	FOSC = 4.000 MHz			FOSC = 2.000 MHz			FOSC = 1.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.01	3332	0.300	-0.04	1665	0.300	-0.04	832
1.2	1.200	0.04	832	1.201	-0.16	415	1.201	-0.16	207
2.4	2.404	0.16	415	2.403	-0.16	207	2.403	-0.16	103
9.6	9.615	0.16	103	9.615	-0.16	51	9.615	-0.16	25
19.2	19.231	0.16	51	19.230	-0.16	25	19.230	-0.16	12
57.6	58.824	2.12	16	55.555	3.55	8	—	—	—
115.2	111.111	-3.55	8	—	—	—	—	—	—

22.1 A/D Acquisition Requirements

For the A/D Converter to meet its specified accuracy, the charge holding capacitor (CHOLD) must be allowed to fully charge to the input channel voltage level. The analog input model is illustrated in Figure 22-2. The source impedance (RS) and the internal sampling switch (RSS) impedance directly affect the time required to charge the capacitor, CHOLD. The sampling switch (RSS) impedance varies over the device voltage (VDD). The source impedance affects the offset voltage at the analog input (due to pin leakage current). **The maximum recommended impedance for analog sources is 2.5 kΩ for 10-bit conversions and 1 kΩ for 12-bit conversions**. After the analog input channel is selected (changed), the channel must be sampled for at least the minimum acquisition time before starting a conversion.

Note:	When the conversion is started, the holding capacitor is disconnected from the input pin.

To calculate the minimum acquisition time, Equation 22-1 may be used. This equation assumes that 1/2 LSb error is used (1,024 steps for the 10-bit A/D). The 1/2 LSb error is the maximum error allowed for the A/D to meet its specified resolution.

Equation 22-3 provides the calculation of the minimum required acquisition time, TACQ. This calculation is based on the following application system assumptions:

CHOLD	=	25 pF
Rs	=	2.5 kΩ
Conversion Error	≤	1/2 LSb
VDD	=	3V → Rss = 2 kΩ
Temperature	=	85°C (system max.)

EQUATION 22-1: ACQUISITION TIME

$$T_{ACQ} = \text{Amplifier Settling Time} + \text{Holding Capacitor Charging Time} + \text{Temperature Coefficient}$$
$$= T_{AMP} + T_C + T_{COFF}$$

EQUATION 22-2: A/D MINIMUM CHARGING TIME

$$V_{HOLD} = (V_{REF} - (V_{REF}/2048)) \bullet (1 - e^{(-T_C/C_{HOLD}(R_{IC} + R_{SS} + R_S))})$$
or
$$T_C = -(C_{HOLD})(R_{IC} + R_{SS} + R_S)\ln(1/2048)$$

EQUATION 22-3: CALCULATING THE MINIMUM REQUIRED ACQUISITION TIME

$$T_{ACQ} = T_{AMP} + T_C + T_{COFF}$$
$$T_{AMP} = 0.2\ \mu s$$
$$T_{COFF} = (\text{Temp} - 25°C)(0.02\ \mu s/°C)$$
$$(85°C - 25°C)(0.02\ \mu s/°C)$$
$$1.2\ \mu s$$

Temperature coefficient is only required for temperatures > 25°C. Below 25°C, $T_{COFF} = 0\ \mu s$.

$$T_C = -(C_{HOLD})(R_{IC} + R_{SS} + R_S)\ln(1/2048)\ \mu s$$
$$-(25\ pF)(1\ k\Omega + 2\ k\Omega + 2.5\ k\Omega)\ln(0.0004883)\ \mu s$$
$$1.05\ \mu s$$
$$T_{ACQ} = 0.2\ \mu s + 1.05\ \mu s + 1.2\ \mu s$$
$$2.45\ \mu s$$

REGISTER 23-2: UCFG: USB CONFIGURATION REGISTER (BANKED F39h)

R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
UTEYE	UOEMON	—	UPUEN(1,2)	UTRDIS(1,3)	FSEN(1)	PPB1	PPB0
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 **UTEYE:** USB Eye Pattern Test Enable bit
1 = Eye pattern test is enabled
0 = Eye pattern test is disabled

bit 6 **UOEMON:** USB OE Monitor Enable bit
1 = $\overline{\text{UOE}}$ signal active, indicating intervals during which the D+/D- lines are driving
0 = $\overline{\text{UOE}}$ signal inactive

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

bit 4 **UPUEN:** USB On-Chip Pull-up Enable bit(1,2)
1 = On-chip pull-up is enabled (pull-up on D+ with FSEN = 1 or D- with FSEN = 0)
0 = On-chip pull-up is disabled

bit 3 **UTRDIS:** On-Chip Transceiver Disable bit(1,3)
1 = On-chip transceiver is disabled
0 = On-chip transceiver is active

bit 2 **FSEN:** Full-Speed Enable bit(1)
1 = Full-speed device: controls transceiver edge rates; requires input clock at 48 MHz
0 = Low-speed device: controls transceiver edge rates; requires input clock at 6 MHz

bit 1-0 **PPB<1:0>:** Ping-Pong Buffers Configuration bits
11 = Even/Odd ping-pong buffers are enabled for Endpoints 1 to 15
10 = Even/Odd ping-pong buffers are enabled for all endpoints
01 = Even/Odd ping-pong buffer are enabled for OUT Endpoint 0
00 = Even/Odd ping-pong buffers are disabled

Note 1: The UPUEN, UTRDIS and FSEN bits should never be changed while the USB module is enabled. These values must be preconfigured prior to enabling the module.
2: This bit is only valid when the on-chip transceiver is active (UTRDIS = 0); otherwise, it is ignored.
3: If UTRDIS is set, the $\overline{\text{UOE}}$ signal will be active – independent of the UOEMON bit setting.

TABLE 29-2: PIC18F47J53 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 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.

TABLE 29-2: PIC18F47J53 FAMILY INSTRUCTION SET (CONTINUED)

Mnemonic, Operands	Description	Cycles	16-Bit Instruction Word MSb LSb	Status Affected	Notes
LITERAL OPERATIONS					
ADDLW k	Add Literal and WREG	1	0000 1111 kkkk kkkk	C, DC, Z, OV, N	
ANDLW k	AND Literal with WREG	1	0000 1011 kkkk kkkk	Z, N	
IORLW k	Inclusive OR Literal with WREG	1	0000 1001 kkkk kkkk	Z, N	
LFSR f, k	Move literal (12-bit) 2nd word to FSR(f) 1st word	2	1110 1110 00ff kkkk 1111 0000 kkkk kkkk	None	
MOVLB k	Move Literal to BSR<3:0>	1	0000 0001 0000 kkkk	None	
MOVLW k	Move Literal to WREG	1	0000 1110 kkkk kkkk	None	
MULLW k	Multiply Literal with WREG	1	0000 1101 kkkk kkkk	None	
RETLW k	Return with Literal in WREG	2	0000 1100 kkkk kkkk	None	
SUBLW k	Subtract WREG from Literal	1	0000 1000 kkkk kkkk	C, DC, Z, OV, N	
XORLW k	Exclusive OR Literal with WREG	1	0000 1010 kkkk kkkk	Z, N	
DATA MEMORY ↔ PROGRAM MEMORY OPERATIONS					
TBLRD*	Table Read	2	0000 0000 0000 1000	None	
TBLRD*+	Table Read with Post-Increment		0000 0000 0000 1001	None	
TBLRD*-	Table Read with Post-Decrement		0000 0000 0000 1010	None	
TBLRD+*	Table Read with Pre-Increment		0000 0000 0000 1011	None	
TBLWT*	Table Write	2	0000 0000 0000 1100	None	
TBLWT*+	Table Write with Post-Increment		0000 0000 0000 1101	None	
TBLWT*-	Table Write with Post-Decrement		0000 0000 0000 1110	None	
TBLWT+*	Table Write with Pre-Increment		0000 0000 0000 1111	None	

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 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.

BTFSC **Bit Test File, Skip if Clear**

Syntax: BTFSC f, b {,a}

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

Operation: skip if (f<b>) = 0

Status Affected: None

Encoding:

1011	bbba	ffff	ffff

Description: If bit 'b' in register 'f' is '0', then the next instruction is skipped. If bit 'b' is '0', then the next instruction fetched during the current instruction execution is discarded and a NOP is executed instead, making this a 2-cycle instruction.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).

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

Q Cycle Activity:

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

If skip:

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

If skip and followed by 2-word instruction:

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

Example:
```
HERE    BTFSC   FLAG, 1, 0
FALSE   :
TRUE    :
```

Before Instruction
PC = address (HERE)
After Instruction
If FLAG<1> = 0;
PC = address (TRUE)
If FLAG<1> = 1;
PC = address (FALSE)

BTFSS **Bit Test File, Skip if Set**

Syntax: BTFSS f, b {,a}

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

Operation: skip if (f<b>) = 1

Status Affected: None

Encoding:

1010	bbba	ffff	ffff

Description: If bit 'b' in register 'f' is '1', then the next instruction is skipped. If bit 'b' is '1', then the next instruction fetched during the current instruction execution is discarded and a NOP is executed instead, making this a 2-cycle instruction.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).

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

Q Cycle Activity:

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

If skip:

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

If skip and followed by 2-word instruction:

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

Example:
```
HERE    BTFSS   FLAG, 1, 0
FALSE   :
TRUE    :
```

Before Instruction
PC = address (HERE)
After Instruction
If FLAG<1> = 0;
PC = address (FALSE)
If FLAG<1> = 1;
PC = address (TRUE)

29.2 Extended Instruction Set

In addition to the standard 75 instructions of the PIC18 instruction set, the PIC18F47J53 family of devices also provide an optional extension to the core CPU functionality. The added features include eight additional instructions that augment Indirect and Indexed Addressing operations and the implementation of Indexed Literal Offset Addressing for many of the standard PIC18 instructions.

The additional features of the extended instruction set are enabled by default on unprogrammed devices. Users must properly set or clear the XINST Configuration bit during programming to enable or disable these features.

The instructions in the extended set can all be classified as literal operations, which either manipulate the File Select Registers, or use them for Indexed Addressing. Two of the instructions, ADDFSR and SUBFSR, each have an additional special instantiation for using FSR2. These versions (ADDULNK and SUBULNK) allow for automatic return after execution.

The extended instructions are specifically implemented to optimize re-entrant program code (that is, code that is recursive or that uses a software stack) written in high-level languages, particularly C. Among other things, they allow users working in high-level languages to perform certain operations on data structures more efficiently. These include:

- Dynamic allocation and deallocation of software stack space when entering and leaving subroutines
- Function Pointer invocation
- Software Stack Pointer manipulation
- Manipulation of variables located in a software stack

A summary of the instructions in the extended instruction set is provided in Table 29-3. Detailed descriptions are provided in **Section 29.2.2 "Extended Instruction Set"**. The opcode field descriptions in Table 29-1 (page 448) apply to both the standard and extended PIC18 instruction sets.

Note:	The instruction set extension and the Indexed Literal Offset Addressing mode were designed for optimizing applications written in C; the user may likely never use these instructions directly in assembler. The syntax for these commands is provided as a reference for users who may be reviewing code that has been generated by a compiler.

29.2.1 EXTENDED INSTRUCTION SYNTAX

Most of the extended instructions use indexed arguments, using one of the File Select Registers and some offset to specify a source or destination register. When an argument for an instruction serves as part of Indexed Addressing, it is enclosed in square brackets ("[ ]"). This is done to indicate that the argument is used as an index or offset. The MPASM™ Assembler will flag an error if it determines that an index or offset value is not bracketed.

When the extended instruction set is enabled, brackets are also used to indicate index arguments in byte-oriented and bit-oriented instructions. This is in addition to other changes in their syntax. For more details, see **Section 29.2.3.1 "Extended Instruction Syntax with Standard PIC18 Commands"**.

Note:	In the past, square brackets have been used to denote optional arguments in the PIC18 and earlier instruction sets. In this text and going forward, optional arguments are denoted by braces ("{ }").

TABLE 29-3: EXTENSIONS TO THE PIC18 INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word				Status Affected
				MSb			LSb	
ADDFSR	f, k	Add Literal to FSR	1	1110	1000	ffkk	kkkk	None
ADDULNK	k	Add Literal to FSR2 and Return	2	1110	1000	11kk	kkkk	None
CALLW		Call Subroutine using WREG	2	0000	0000	0001	0100	None
MOVSF	z_s, f_d	Move z_s (source) to 1st word	2	1110	1011	0zzz	zzzz	None
		f_d (destination) 2nd word		1111	ffff	ffff	ffff	
MOVSS	z_s, z_d	Move z_s (source) to 1st word	2	1110	1011	1zzz	zzzz	None
		z_d (destination) 2nd word		1111	xxxx	xzzz	zzzz	
PUSHL	k	Store Literal at FSR2, Decrement FSR2	1	1110	1010	kkkk	kkkk	None
SUBFSR	f, k	Subtract Literal from FSR	1	1110	1001	ffkk	kkkk	None
SUBULNK	k	Subtract Literal from FSR2 and return	2	1110	1001	11kk	kkkk	None