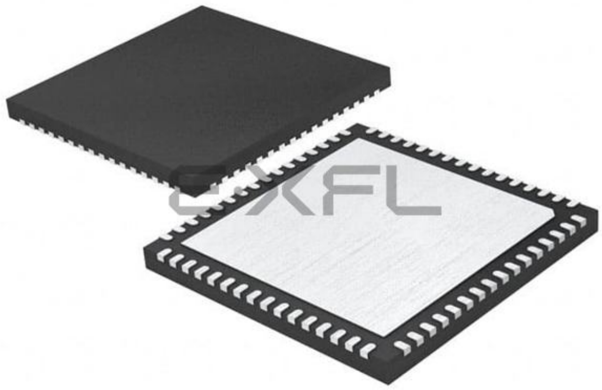


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	64MHz
Connectivity	I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	53
Program Memory Size	64KB (32K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	4K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.5V
Data Converters	A/D 16x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-VFQFN Exposed Pad
Supplier Device Package	64-VQFN (9x9)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f66k22t-i-mr

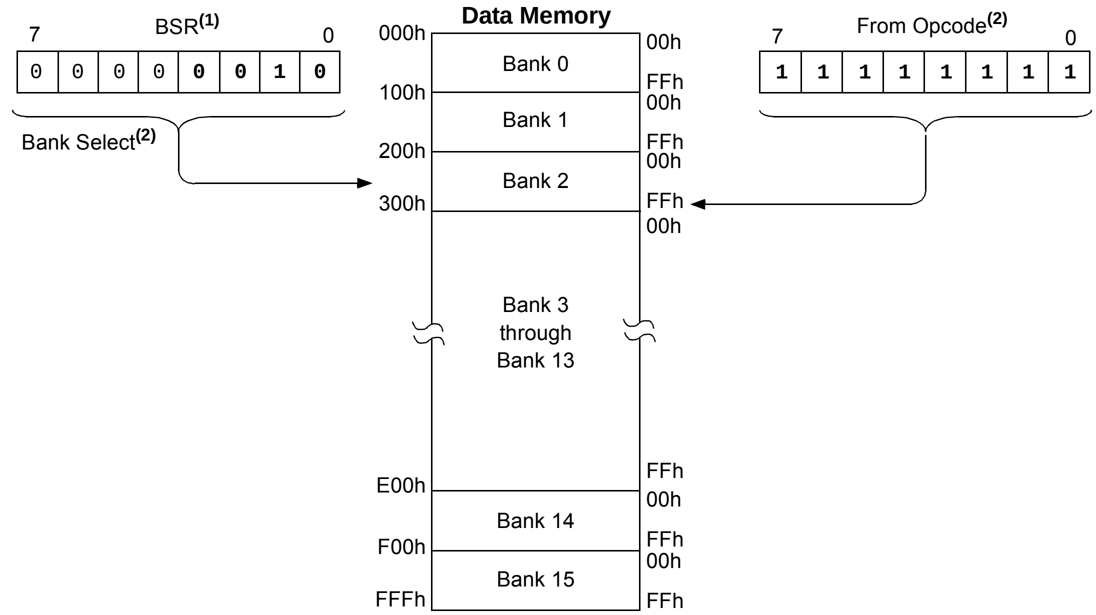
TABLE 1-4: PIC18F8XK22 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number TQFP	Pin Type	Buffer Type	Description
				PORTD is a bidirectional I/O port.
RD0/PSP0/CTPLS/AD0	72			
RD0		I/O	ST	Digital I/O
PSP0(4)		I/O	TTL	Parallel Slave Port data
CTPLS		O	ST	CTMU pulse generator output
AD0		I/O	TTL	External Memory Address/Data 0
RD1/T5CKI/T7G/PSP1/AD1	69			
RD1		I/O	ST	Digital I/O
T5CKI		I	ST	Timer5 clock input
T7G		I	ST	Timer7 external clock gate input
PSP1(4)		I/O	TTL	Parallel Slave Port data
AD1		I/O	TTL	External Memory Address/Data 1
RD2/PSP2/AD2	68			
RD2		I/O	ST	Digital I/O.
PSP2(4)		I/O	TTL	Parallel Slave Port data.
AD2		I/O	TTL	External Memory Address/Data 2.
RD3/PSP3/AD3	67			
RD3		I/O	ST	Digital I/O.
PSP3(4)		I/O	TTL	Parallel Slave Port data.
AD3		I/O	TTL	External Memory Address/Data 3.
RD4/SDO2/PSP4/AD4	66			
RD4		I/O	ST	Digital I/O.
SDO2		O	—	SPI data out.
PSP4(4)		I/O	TTL	Parallel Slave Port data.
AD4		I/O	TTL	External Memory Address/Data 4.
RD5/SDI2/SDA2/PSP5/AD5	65			
RD5		I/O	ST	Digital I/O.
SDI2		I	ST	SPI data in.
SDA2		I/O	I^2C	I^2C™ data I/O.
PSP5(4)		I/O	TTL	Parallel Slave Port data.
AD5		I/O	TTL	External Memory Address/Data 5.

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})

Note 1: Default assignment for ECCP2 when the CCP2MX Configuration bit is set.
2: Alternate assignment for ECCP2 when the CCP2MX Configuration bit is cleared.
3: Not available on PIC18F65K22 and PIC18F85K22 devices.
4: PSP is available only in Microcontroller mode.
5: The CC6, CCP7, CCP8 and CCP9 pin placement depends on the setting of the ECCPMX Configuration bit (CONFIG3H<1>).

FIGURE 6-7: USE OF THE BANK SELECT REGISTER (DIRECT ADDRESSING)



Note 1: The Access RAM bit of the instruction can be used to force an override of the selected bank (BSR<3:0>) to the registers of the Access Bank.

2: The `MOVFF` instruction embeds the entire 12-bit address in the instruction.

6.3.2 ACCESS BANK

While the use of the BSR, with an embedded 8-bit address, allows users to address the entire range of data memory, it also means that the user must ensure that the correct bank is selected. If not, data may be read from, or written to, the wrong location. This can be disastrous if a GPR is the intended target of an operation, but an SFR is written to instead. Verifying and/or changing the BSR for each read or write to data memory can become very inefficient.

To streamline access for the most commonly used data memory locations, the data memory is configured with an Access Bank, which allows users to access a mapped block of memory without specifying a BSR. The Access Bank consists of the first 96 bytes of memory (00h-5Fh) in Bank 0 and the last 160 bytes of memory (60h-FFh) in Bank 15. The lower half is known as the “Access RAM” and is composed of GPRs. The upper half is where the device’s SFRs are mapped. These two areas are mapped contiguously in the Access Bank and can be addressed in a linear fashion by an eight-bit address (Figure 6-6).

The Access Bank is used by core PIC18 instructions that include the Access RAM bit (the ‘a’ parameter in the instruction). When ‘a’ is equal to ‘`1`’, the instruction uses the BSR and the 8-bit address included in the opcode for the data memory address. When ‘a’ is ‘`0`’, however, the instruction is forced to use the Access Bank address map. In that case, the current value of the BSR is ignored entirely.

Using this “forced” addressing allows the instruction to operate on a data address in a single cycle without updating the BSR first. For 8-bit addresses of 60h and above, this means that users can evaluate and operate on SFRs more efficiently. The Access RAM below 60h is a good place for data values that the user might need to access rapidly, such as immediate computational results or common program variables.

Access RAM also allows for faster and more code efficient context saving and switching of variables.

The mapping of the Access Bank is slightly different when the extended instruction set is enabled (XINST Configuration bit = `1`). This is discussed in more detail in **Section 6.6.3 “Mapping the Access Bank in Indexed Literal Offset Mode”**.

6.3.3 GENERAL PURPOSE REGISTER FILE

PIC18 devices may have banked memory in the GPR area. This is data RAM which is available for use by all instructions. GPRs start at the bottom of Bank 0 (address 000h) and grow upwards towards the bottom of the SFR area. GPRs are not initialized by a Power-on Reset and are unchanged on all other Resets.

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”**.

7.0 FLASH PROGRAM MEMORY

The Flash program memory is readable, writable and erasable during normal operation over the entire VDD range.

A read from program memory is executed on one byte at a time. For execution of a write to, or erasure of, program memory:

- Memory of 32 Kbytes and 64 Kbytes (PIC18FX5K22 and PIC18FX6K22 devices) – Blocks of 64 bytes
- Memory of 128 Kbytes (PIC18FX7K22 devices) – Blocks of 128 bytes

Writing or erasing program memory will cease instruction fetches until the operation is complete. The program memory cannot be accessed during the write or erase, therefore, code cannot execute. An internal programming timer terminates program memory writes and erases.

A value written to program memory does not need to be a valid instruction. Executing a program memory location that forms an invalid instruction results in a NOP.

7.1 Table Reads and Table Writes

In order to read and write program memory, there are two operations that allow the processor to move bytes between the program memory space and the data RAM:

- Table Read (TBLRD)
- Table Write (TBLWT)

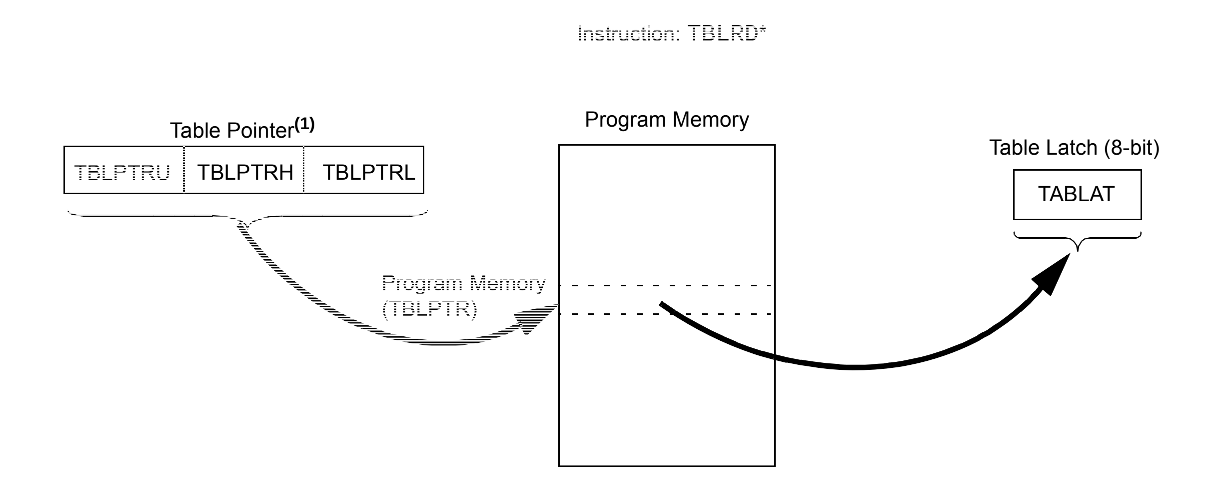
The program memory space is 16 bits wide, while the data RAM space is 8 bits wide. Table reads and table writes move data between these two memory spaces through an 8-bit register (TABLAT).

Table read operations retrieve data from program memory and place it into the data RAM space. Figure 7-1 shows the operation of a table read with program memory and data RAM.

Table write operations store data from the data memory space into holding registers in program memory. The procedure to write the contents of the holding registers into program memory is detailed in **Section 7.5 "Writing to Flash Program Memory"**. Figure 7-2 shows the operation of a table write with program memory and data RAM.

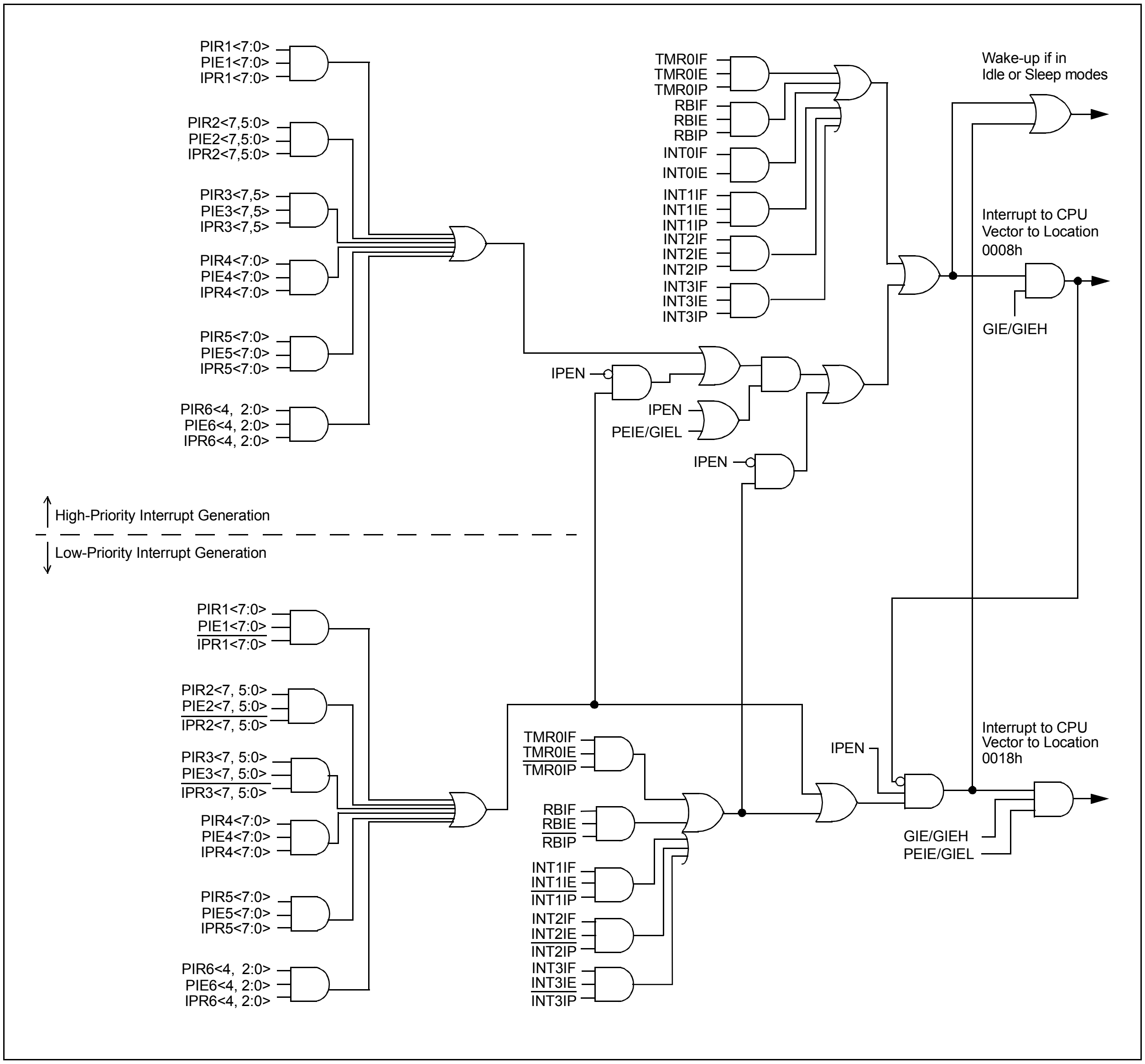
Table operations work with byte entities. A table block containing data, rather than program instructions, is not required to be word-aligned. Therefore, a table block can start and end at any byte address. If a table write is being used to write executable code into program memory, program instructions will need to be word-aligned.

FIGURE 7-1: TABLE READ OPERATION



Note 1: The Table Pointer register points to a byte in program memory.

FIGURE 11-1: PIC18F87K22 FAMILY INTERRUPT LOGIC



11.1 INTCON Registers

The INTCON registers are readable and writable registers that contain various enable, priority and flag bits.

Note:	Interrupt flag bits are set when an interrupt condition occurs regardless of the state of its corresponding enable bit or the Global Interrupt Enable bit. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt. This feature allows for software polling.

REGISTER 11-1: INTCON: INTERRUPT CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF(1)
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 **GIE/GIEH:** Global Interrupt Enable bit
When IPEN = 0:
1 = Enables all unmasked interrupts
0 = Disables all interrupts
When IPEN = 1:
1 = Enables all high-priority interrupts
0 = Disables all interrupts

bit 6 **PEIE/GIEL:** Peripheral Interrupt Enable bit
When IPEN = 0:
1 = Enables all unmasked peripheral interrupts
0 = Disables all peripheral interrupts
When IPEN = 1:
1 = Enables all low-priority peripheral interrupts
0 = Disables all low-priority peripheral interrupts

bit 5 **TMR0IE:** TMR0 Overflow Interrupt Enable bit
1 = Enables the TMR0 overflow interrupt
0 = Disables the TMR0 overflow interrupt

bit 4 **INT0IE:** INT0 External Interrupt Enable bit
1 = Enables the INT0 external interrupt
0 = Disables the INT0 external interrupt

bit 3 **RBIE:** RB Port Change Interrupt Enable bit
1 = Enables the RB port change interrupt
0 = Disables the RB port change interrupt

bit 2 **TMR0IF:** TMR0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed (must be cleared in software)
0 = TMR0 register did not overflow

bit 1 **INT0IF:** INT0 External Interrupt Flag bit
1 = The INT0 external interrupt occurred (must be cleared in software)
0 = The INT0 external interrupt did not occur

bit 0 **RBIF:** RB Port Change Interrupt Flag bit(1)
1 = At least one of the RB<7:4> pins changed state (must be cleared in software)
0 = None of the RB<7:4> pins have changed state

Note 1: A mismatch condition will continue to set this bit. Reading PORTB, and then waiting one additional instruction cycle, will end the mismatch condition and allow the bit to be cleared.

REGISTER 11-21: IPR6: PERIPHERAL INTERRUPT PRIORITY REGISTER 6

U-0	U-0	U-0	R/W-1	U-0	R/W-1	R/W-1	R/W-1
—	—	—	EEIP	—	CMP3IP	CMP2IP	CMP1IP
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 **EEIP:** EE Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **Unimplemented:** Read as ‘0’

bit 2 **CMP3IP:** CMP3 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **CMP2IP:** CMP2 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **CMP1IP:** CMP1 Interrupt Priority bit
1 = High priority
0 = Low priority

TABLE 12-15: PORTH FUNCTIONS (CONTINUED)

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RH4/CCP9/P3C/AN12/C2INC	RH4	0	O	DIG	LATH<4> data output.
		1	I	ST	PORTH<4> data input.
	CCP9	0	O	DIG	CCP9 compare/PWM output; takes priority over port data.
		1	I	ST	CCP9 capture input.
	P3C	0	O	—	ECCP3 PWM Output C. May be configured for tri-state during Enhanced PWM.
	AN12	1	I	ANA	A/D Input Channel 12. Default input configuration on POR; does not affect digital input.
	C2INC	x	I	ANA	Comparator 2 Input C.
RH5/CCP8/P3B/AN13/C2IND	RH5	0	O	DIG	LATH<5> data output.
		1	I	ST	PORTH<5> data input.
	CCP8	0	O	DIG	CCP8 compare/PWM output; takes priority over port data.
		1	I	ST	CCP8 capture input.
	P3B	0	O	—	ECCP3 PWM Output B. May be configured for tri-state during Enhanced PWM.
	AN13	1	I	ANA	A/D Input Channel 13. Default input configuration on POR; does not affect digital input.
	C2IND	x	I	ANA	Comparator 2 Input D.
RH6/CCP7/P1C/AN14/C1INC	RH6	0	O	DIG	LATH<6> data output.
		1	I	ST	PORTH<6> data input.
	CCP7	0	O	DIG	CCP7 compare/PWM output; takes priority over port data.
		1	I	ST	CCP7 capture input.
	P1C	0	O	—	ECCP1 PWM Output C. May be configured for tri-state during Enhanced PWM.
	AN14	1	I	ANA	A/D Input Channel 14. Default input configuration on POR; does not affect digital input.
	C1INC	x	I	ANA	Comparator 1 Input C.
RH7/CCP6/P1B/AN15	RH7	0	O	DIG	LATH<7> data output.
		1	I	ST	PORTH<7> data input.
	CCP6	0	O	DIG	CCP6 compare/PWM output; takes priority over port data.
		1	I	ST	CCP6 capture input.
	P1B	0	O	—	ECCP1 PWM Output B. May be configured for tri-state during Enhanced PWM.
	AN15	1	I	ANA	A/D Input Channel 15. Default input configuration on POR; does not affect digital input.

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).

14.0 TIMER1 MODULE

The Timer1 timer/counter module incorporates these features:

- Software-selectable operation as a 16-bit timer or counter
- Readable and writable 8-bit registers (TMR1H and TMR1L)
- Selectable clock source (internal or external) with device clock or SOSC oscillator internal options
- Interrupt-on-overflow
- Reset on ECCP Special Event Trigger
- Timer with gated control

Figure 14-1 displays a simplified block diagram of the Timer1 module.

The Timer1 oscillator can also be used as a low-power clock source for the microcontroller in power-managed operation. The Timer1 can also work on the SOSC oscillator.

Timer1 is controlled through the T1CON Control register (Register 14-1). It also contains the Secondary Oscillator Enable bit (SOSCEN). Timer1 can be enabled or disabled by setting or clearing control bit, TMR1ON (T1CON<0>).

The FOSC clock source should not be used with the ECCP capture/compare features. If the timer will be used with the capture or compare features, always select one of the other timer clocking options.

REGISTER 14-1: T1CON: TIMER1 CONTROL 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
TMR1CS1	TMR1CS0	T1CKPS1	T1CKPS0	SOSCEN	$\overline{\text{T1SYNC}}$	RD16	TMR1ON
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 7-6 **TMR1CS<1:0>:** Timer1 Clock Source Select bits
`10` = Timer1 clock source is either from a pin or oscillator, depending on the SOSCEN bit:
SOSCEN = `0`:
External clock from the T1CKI pin (on the rising edge).
SOSCEN = `1`:
Depending on the SOSCSEL Configuration bit, the clock source is either a crystal oscillator on the SOSCI/SOSCO pins or an internal clock from the SCLKI pin.
`01` = Timer1 clock source is the system clock (FOSC)[(1)]
`00` = Timer1 clock source is the instruction clock (FOSC/4)

bit 5-4 **T1CKPS<1:0>:** Timer1 Input Clock Prescale Select bits
`11` = 1:8 Prescale value
`10` = 1:4 Prescale value
`01` = 1:2 Prescale value
`00` = 1:1 Prescale value

bit 3 **SOSCEN:** SOSC Oscillator Enable bit
`1` = SOSC is enabled and available for Timer1
`0` = SOSC is disabled for Timer1
The oscillator inverter and feedback resistor are turned off to eliminate power drain.

bit 2 **$\overline{\text{T1SYNC}}$:** Timer1 External Clock Input Synchronization Select bit
TMR1CS<1:0> = `10`:
`1` = Do not synchronize external clock input
`0` = Synchronize external clock input
TMR1CS<1:0> = `0x`:
This bit is ignored. Timer1 uses the internal clock when TMR1CS<1:0> = `1x`.

bit 1 **RD16:** 16-Bit Read/Write Mode Enable bit
`1` = Enables register read/write of Timer1 in one 16-bit operation
`0` = Enables register read/write of Timer1 in two 8-bit operations

bit 0 **TMR1ON:** Timer1 On bit
`1` = Enables Timer1
`0` = Stops Timer1

Note 1: The FOSC clock source should not be selected if the timer will be used with the ECCP capture/compare features.

17.0 TIMER4/6/8/10/12 MODULES

The Timer4/6/8/10/12 timer modules have the following features:

- Eight-bit Timer register (TMRx)
- Eight-bit Period register (PRx)
- Readable and writable (all registers)
- Software programmable prescaler (1:1, 1:4, 1:16)
- Software programmable postscaler (1:1 to 1:16)
- Interrupt on TMRx match of PRx

Timer10 and Timer12 are unimplemented for devices with a program memory of 32 Kbytes (PIC18FX5K22).

Note:	Throughout this section, generic references are used for register and bit names that are the same, except for an 'x' variable that indicates the item's association with the Timer4, Timer6, Timer8, Timer10 or Timer12 module. For example, the control register is named TxCON and refers to T4CON, T6CON, T8CON, T10CON and T12CON.

The Timer4/6/8/10/12 modules have a control register, which is shown in Register 17-1. Timer4/6/8/10/12 can be shut off by clearing control bit, TMRxON (TxCON<2>), to minimize power consumption. The prescaler and postscaler selection of Timer4/6/8/10/12 are also controlled by this register. Figure 17-1 is a simplified block diagram of the Timer4/6/8/10/12 modules.

17.1 Timer4/6/8/10/12 Operation

Timer4/6/8/10/12 can be used as the PWM time base for the PWM mode of the ECCP modules. The TMRx registers are readable and writable, and are cleared on any device Reset. The input clock (FOSC/4) has a prescale option of 1:1, 1:4 or 1:16, selected by control bits, TxCKPS<1:0> (TxCON<1:0>). The match output of TMRx goes through a four-bit postscaler (that gives a 1:1 to 1:16 inclusive scaling) to generate a TMRx interrupt, latched in the flag bit, TMRxIF. Table 17-1 shows each module's flag bit.

TABLE 17-1: TIMER4/6/8/10/12 FLAG BITS

Timer Module	Flag Bit PIR5<x>	Timer Module	Flag Bit PIR5<x>
4	0	10	5
6	2	12	6
8	4		

The interrupt can be enabled or disabled by setting or clearing the Timerx Interrupt Enable bit (TMRxIE), shown in Table 17-2.

TABLE 17-2: TIMER4/6/8/10/12 INTERRUPT ENABLE BITS

Timer Module	Flag Bit PIE5<x>	Timer Module	Flag Bit PIE5<x>
4	0	10	5
6	2	12	6
8	4		

The prescaler and postscaler counters are cleared when any of the following occurs:

- A write to the TMRx register
- A write to the TxCON register
- Any device Reset – Power-on Reset (POR), $\overline{\text{MCLR}}$ Reset, Watchdog Timer Reset (WDTR) or Brown-out Reset (BOR)

A TMRx is not cleared when a TxCON is written.

Note:	The CCP and ECCP modules use Timers, 1 through 8, for some modes. The assignment of a particular timer to a CCP/ECCP module is determined by the Timer to CCP enable bits in the CCPTMRSx registers. For more details, see Register 19-2, Register 19-3 and Register 20-2.

19.4 PWM Mode

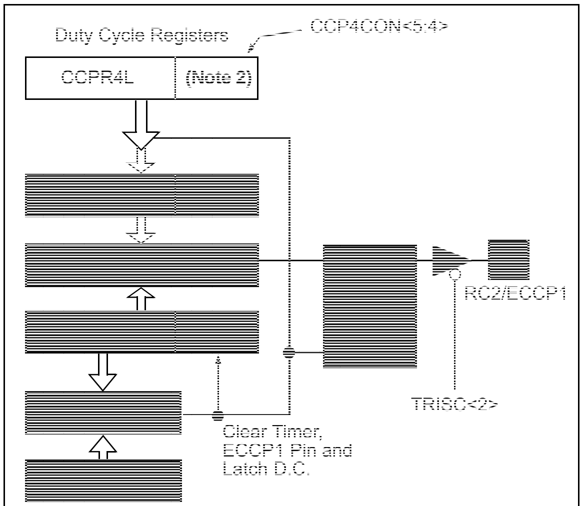
In Pulse-Width Modulation (PWM) mode, the CCP4 pin produces up to a 10-bit resolution PWM output. Since the CCP4 pin is multiplexed with a PORTC or PORTE data latch, the appropriate TRIS bit must be cleared to make the CCP4 pin an output.

Note:	Clearing the CCP4CON register will force the RC1 or RE7 output latch (depending on device configuration) to the default low level. This is not the PORTC or PORTE I/O data latch.

Figure 19-3 shows a simplified block diagram of the ECCP1 module in PWM mode.

For a step-by-step procedure on how to set up the CCP module for PWM operation, see **Section 19.4.3 "Setup for PWM Operation"**.

FIGURE 19-3: SIMPLIFIED PWM BLOCK DIAGRAM

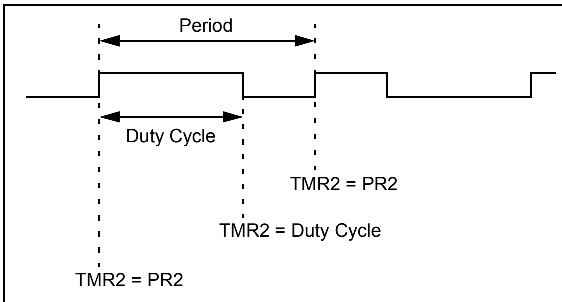


Note 1: The 8-bit TMR2 value is concatenated with the 2-bit internal Q clock, or 2 bits of the prescaler, to create the 10-bit time base.

2: CCP4 and its appropriate timers are used as an example. For details on all of the CCP modules and their timer assignments, see Table 19-2 and Table 19-3.

A PWM output (Figure 19-4) has a time base (period) and a time that the output stays high (duty cycle). The frequency of the PWM is the inverse of the period (1/period).

FIGURE 19-4: PWM OUTPUT



19.4.1 PWM PERIOD

The PWM period is specified by writing to the PR2 register. The PWM period can be calculated using the following formula:

EQUATION 19-1:

$$\text{PWM Period} = [(PR2) + 1] \cdot 4 \cdot T_{OSC} \cdot (\text{TMR2 Prescale Value})$$

PWM frequency is defined as 1/[PWM period].

When TMR2 is equal to PR2, the following three events occur on the next increment cycle:

- TMR2 is cleared
- The CCP4 pin is set
  (An exception: If PWM duty cycle = 0%, the CCP4 pin will not be set)
- The PWM duty cycle is latched from CCPR4L into CCPR4H

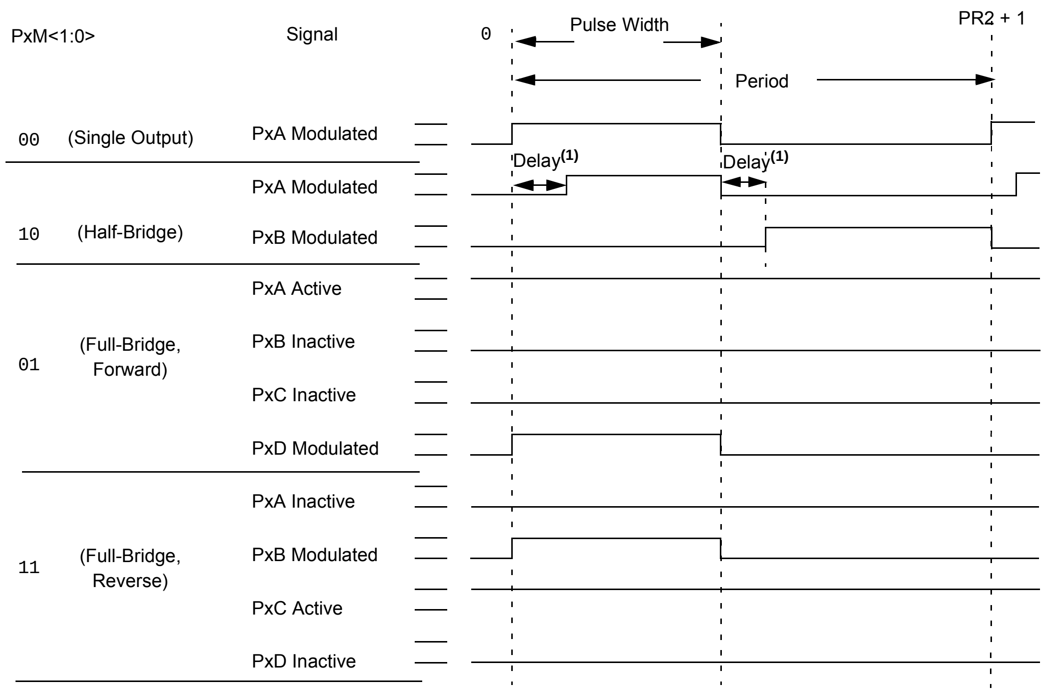
Note:	The Timer2 postscalers (see **Section 15.0 "Timer2 Module"**) are not used in the determination of the PWM frequency. The postscaler could be used to have a servo update rate at a different frequency than the PWM output.

TABLE 20-3: EXAMPLE PIN ASSIGNMENTS FOR VARIOUS PWM ENHANCED MODES

ECCP Mode	PxM<1:0>	PxA	PxB	PxC	PxD
Single	00	Yes[1]	Yes[1]	Yes[1]	Yes[1]
Half-Bridge	10	Yes	Yes	No	No
Full-Bridge, Forward	01	Yes	Yes	Yes	Yes
Full-Bridge, Reverse	11	Yes	Yes	Yes	Yes

Note 1: Outputs are enabled by pulse steering in Single mode (see Register 20-5).

FIGURE 20-4: EXAMPLE PWM (ENHANCED MODE) OUTPUT RELATIONSHIPS (ACTIVE-HIGH STATE)



Relationships:

- Period = 4 * TOSC * (PR2 + 1) * (TMR2 Prescale Value)
- Pulse Width = TOSC * (CCPRxL<7:0>:CCPxCON<5:4>) * (TMR2 Prescale Value)
- Delay = 4 * TOSC * (ECCPxDEL<6:0>)

Note 1: Dead-band delay is programmed using the ECCPxDEL register (see **Section 20.4.6 “Programmable Dead-Band Delay Mode”**).

TABLE 21-2: REGISTERS ASSOCIATED WITH SPI OPERATION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
PIR2	OSCFIF	—	SSP2IF	BCL2IF	BCL1IF	HLVDIF	TMR3IF	TMR3GIF
PIE2	OSCFIE	—	SSP2IE	BCL2IE	BCL1IE	HLVDIE	TMR3IE	TMR3GIE
IPR2	OSCFIP	—	SSP2IP	BCL2IP	BCL1IP	HLVDIP	TMR3IP	TMR3GIP
PIR3	TMR5GIF	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	RTCCIF
PIE3	TMR5GIE	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	RTCCIE
IPR3	TMR5GIP	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	RTCCIP
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0
TRISD	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	—
SSP1BUF	MSSP1 Receive Buffer/Transmit Register							
SSP1CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
SSP1CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN
SSP1STAT	SMP	CKE	$D/\overline{A}$	P	S	$R/\overline{W}$	UA	BF
SSP2CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
SSP2CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN
SSP2STAT	SMP	CKE	$D/\overline{A}$	P	S	$R/\overline{W}$	UA	BF
SSP2BUF	MSSP2 Receive Buffer/Transmit Register							
ODCON1	SSP1OD	CCP2OD	CCP1OD	—	—	—	—	SSP2OD
PMD0	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSP2MD	SSP1MD	ADCMD

Legend: Shaded cells are not used by the MSSP module in SPI mode.

23.8 Use of the Special Event Triggers

A/D conversion can be started by the Special Event Trigger of any of these modules:

- ECCP2 – Requires CCP2M<3:0> bits (CCP2CON<3:0>) set at '`1011`'
- CTMU – Requires the setting of the CTTRIG bit (CTMUCONH<0>)
- Timer1
- RTCC

To start an A/D conversion:

- The A/D module must be enabled (ADON = `1`)
- The appropriate analog input channel is selected
- The minimum acquisition period is set one of these ways:
  - Timing provided by the user
  - Selection made of an appropriate TACQ time

With these conditions met, the trigger sets the GO/$\overline{\text{DONE}}$ bit and the A/D acquisition starts.

If the A/D module is not enabled (ADON = `0`), the module ignores the Special Event Trigger.

> **Note:** With an ECCP2 trigger, Timer1 or Timer 3 is cleared. The timers reset to automatically repeat the A/D acquisition period with minimal software overhead (moving ADRESH:ADRESL to the desired location). If the A/D module is not enabled, the Special Event Trigger is ignored by the module, but the timer's counter resets.

23.9 Operation in Power-Managed Modes

The selection of the automatic acquisition time and A/D conversion clock is determined, in part, by the clock source and frequency while in a power-managed mode.

If the A/D is expected to operate while the device is in a power-managed mode, the ACQT<2:0> and ADCS<2:0> bits in ADCON2 should be updated in accordance with the power-managed mode clock that will be used.

After the power-managed mode is entered (either of the power-managed Run modes), an A/D acquisition or conversion may be started. Once an acquisition or conversion is started, the device should continue to be clocked by the same power-managed mode clock source until the conversion has been completed. If desired, the device may be placed into the corresponding power-managed Idle mode during the conversion.

If the power-managed mode clock frequency is less than 1 MHz, the A/D RC clock source should be selected.

Operation in Sleep mode requires that the A/D RC clock be selected. If bits, ACQT<2:0>, are set to '`000`' and a conversion is started, the conversion will be delayed one instruction cycle to allow execution of the `SLEEP` instruction and entry into Sleep mode. The IDLEN and SCS<1:0> bits in the OSCCON register must have already been cleared prior to starting the conversion.

NOTES:

26.6 Operation During Sleep

When enabled, the HLVD circuitry continues to operate during Sleep. If the device voltage crosses the trip point, the HLVDIF bit will be set and the device will wake up from Sleep. Device execution will continue from the interrupt vector address if interrupts have been globally enabled.

26.7 Effects of a Reset

A device Reset forces all registers to their Reset state. This forces the HLVD module to be turned off.

TABLE 26-1: REGISTERS ASSOCIATED WITH HIGH/LOW-VOLTAGE DETECT MODULE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
HLVDCON	VDIRMAG	BGVST	IRVST	HLVDEN	HLVDL3	HLVDL2	HLVDL1	HLVDL0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
PIR2	OSCFIF	—	SSP2IF	BLC2IF	BCL1IF	HLVDIF	TMR3IF	TMR3GIF
PIE2	OSCFIE	—	SSP2IE	BLC2IE	BCL1IE	HLVDIE	TMR3IE	TMR3GIE
IPR2	OSCFIP	—	SSP2IP	BLC2IP	BCL1IP	HLVDIP	TMR3IP	TMR3GIP
TRISA	TRISA7(1)	TRISA6(1)	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0
ANCON0	ANSEL7	ANSEL6	ANSEL5	ANSEL4	ANSEL3	ANSEL2	ANSEL1	ANSEL0

Legend: — = unimplemented, read as ‘0’. Shaded cells are unused by the HLVD module.

Note 1: PORTA<7:6> and their direction bits are individually configured as port pins based on various primary oscillator modes. When disabled, these bits read as ‘0’.

EXAMPLE 27-3: CAPACITANCE CALIBRATION ROUTINE

```
#include "p18cxxx.h"

#define COUNT 25                         //@ 8MHz INTFRC = 62.5 us.
#define ETIME COUNT*2.5                  //time in uS
#define DELAY for(i=0;i<COUNT;i++)
#define ADSCALE 1023                     //for unsigned conversion 10 sig bits
#define ADREF 3.3                        //Vdd connected to A/D Vr+
#define RCAL .027                        //R value is 4200000 (4.2M)
                                         //scaled so that result is in
                                         //1/100th of uA

int main(void)
{
    int i;
    int j = 0;                           //index for loop
    unsigned int Vread = 0;
    float CTMUISrc, CTMUCap, Vavg, VTot, Vcal;

//assume CTMU and A/D have been setup correctly
//see Example 25-1 for CTMU & A/D setup
setup();

CTMUCONHbits.CTMUEN = 1;                 //Enable the CTMU
    for(j=0;j<10;j++)
    {
        CTMUCONHbits.IDISSEN = 1;        //drain charge on the circuit
        DELAY;                           //wait 125us
        CTMUCONHbits.IDISSEN = 0;        //end drain of circuit

        CTMUCONLbits.EDG1STAT = 1;       //Begin charging the circuit
                                         //using CTMU current source
        DELAY;                           //wait for 125us
        CTMUCONLbits.EDG1STAT = 0;       //Stop charging circuit

        PIR1bits.ADIF = 0;               //make sure A/D Int not set
        ADCON0bits.GO=1;                 //and begin A/D conv.
        while(!PIR1bits.ADIF);           //Wait for A/D convert complete


        Vread = ADRES;                   //Get the value from the A/D
        PIR1bits.ADIF = 0;               //Clear A/D Interrupt Flag
        VTot += Vread;                   //Add the reading to the total
    }

    Vavg = (float)(VTot/10.000);         //Average of 10 readings
    Vcal = (float)(Vavg/ADSCALE*ADREF);
    CTMUISrc = Vcal/RCAL;                //CTMUISrc is in 1/100ths of uA
    CTMUCap = (CTMUISrc*ETIME/Vcal)/100;
}
```

30.0 DEVELOPMENT SUPPORT

The PIC® microcontrollers and dsPIC® digital signal controllers are supported with a full range of software and hardware development tools:

- Integrated Development Environment
  - MPLAB® IDE Software
- Compilers/Assemblers/Linkers
  - MPLAB C Compiler for Various Device Families
  - HI-TECH C for Various Device Families
  - MPASM™ Assembler
  - MPLINK™ Object Linker/ MPLIB™ Object Librarian
  - MPLAB Assembler/Linker/Librarian for Various Device Families
- Simulators
  - MPLAB SIM Software Simulator
- Emulators
  - MPLAB REAL ICE™ In-Circuit Emulator
- In-Circuit Debuggers
  - MPLAB ICD 3
  - PICkit™ 3 Debug Express
- Device Programmers
  - PICkit™ 2 Programmer
  - MPLAB PM3 Device Programmer
- Low-Cost Demonstration/Development Boards, Evaluation Kits, and Starter Kits

30.1 MPLAB Integrated Development Environment Software

The MPLAB IDE software brings an ease of software development previously unseen in the 8/16/32-bit microcontroller market. The MPLAB IDE is a Windows® operating system-based application that contains:

- A single graphical interface to all debugging tools
  - Simulator
  - Programmer (sold separately)
  - In-Circuit Emulator (sold separately)
  - In-Circuit Debugger (sold separately)
- A full-featured editor with color-coded context
- A multiple project manager
- Customizable data windows with direct edit of contents
- High-level source code debugging
- Mouse over variable inspection
- Drag and drop variables from source to watch windows
- Extensive on-line help
- Integration of select third party tools, such as IAR C Compilers

The MPLAB IDE allows you to:

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

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

31.2 DC Characteristics: Power-Down and Supply Current PIC18F87K22 Family (Industrial/Extended) (Continued)

PIC18F87K22 Family (Industrial/Extended)		Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial; -40°C ≤ TA ≤ +125°C for extended					
Param No.	**Device**	**Typ**	**Max**	**Units**	**Conditions**		
	Supply Current (IDD) Cont.(2,3)						
	All devices	130	390	μA	-40°C	VDD = 1.8V(4) Regulator Disabled	FOSC = 1 MHz (**PRI_RUN** mode, EC oscillator)
		130	390	μA	+25°C		
		130	390	μA	+85°C		
		250	500	μA	+125ºC		
	All devices	270	790	μA	-40°C	VDD = 3.3V(4) Regulator Disabled	
		270	790	μA	+25°C		
		270	790	μA	+85°C		
		400	900	μA	+125ºC		
	All devices	430	990	μA	-40°C	VDD = 5V(5) Regulator Enabled	
		450	980	μA	+25°C		
		460	980	μA	+85°C		
		600	1300	μA	+125ºC		
	All devices	430	860	μA	-40°C	VDD = 1.8V(4) Regulator Disabled	FOSC = 4 MHz (**PRI_RUN** mode, EC oscillator)
		530	900	μA	+25°C		
		490	880	μA	+85°C		
		750	1600	μA	+125ºC		
	All devices	850	1750	μA	-40°C	VDD = 3.3V(4) Regulator Disabled	
		850	1700	μA	+25°C		
		850	1800	μA	+85°C		
		1150	2400	μA	+125ºC		
	All devices	1.1	2.7	mA	-40°C	VDD = 5V(5) Regulator Enabled	
		1.1	2.6	mA	+25°C		
		1.1	2.6	mA	+85°C		
		2.0	4.0	mA	+125ºC		
	All devices	12	19	mA	-40°C	VDD = 3.3V(4) Regulator Disabled	FOSC = 64 MHz (**PRI_RUN** mode, EC oscillator)
		12	19	mA	+25°C		
		12	19	mA	+85°C		
		13	22	mA	+125ºC(6)		
	All devices	13	20	mA	-40°C	VDD = 5V(4) Regulator Enabled	
		13	20	mA	+25°C		
		13	20	mA	+85°C		
		14	23	mA	+125ºC(6)		

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in a high-impedance state and tied to VDD or VSS, and all features that add delta current are disabled (such as WDT, SOSC oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in Active Operation mode are:
OSC1 = External square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
MCLR = VDD; WDT enabled/disabled as specified.

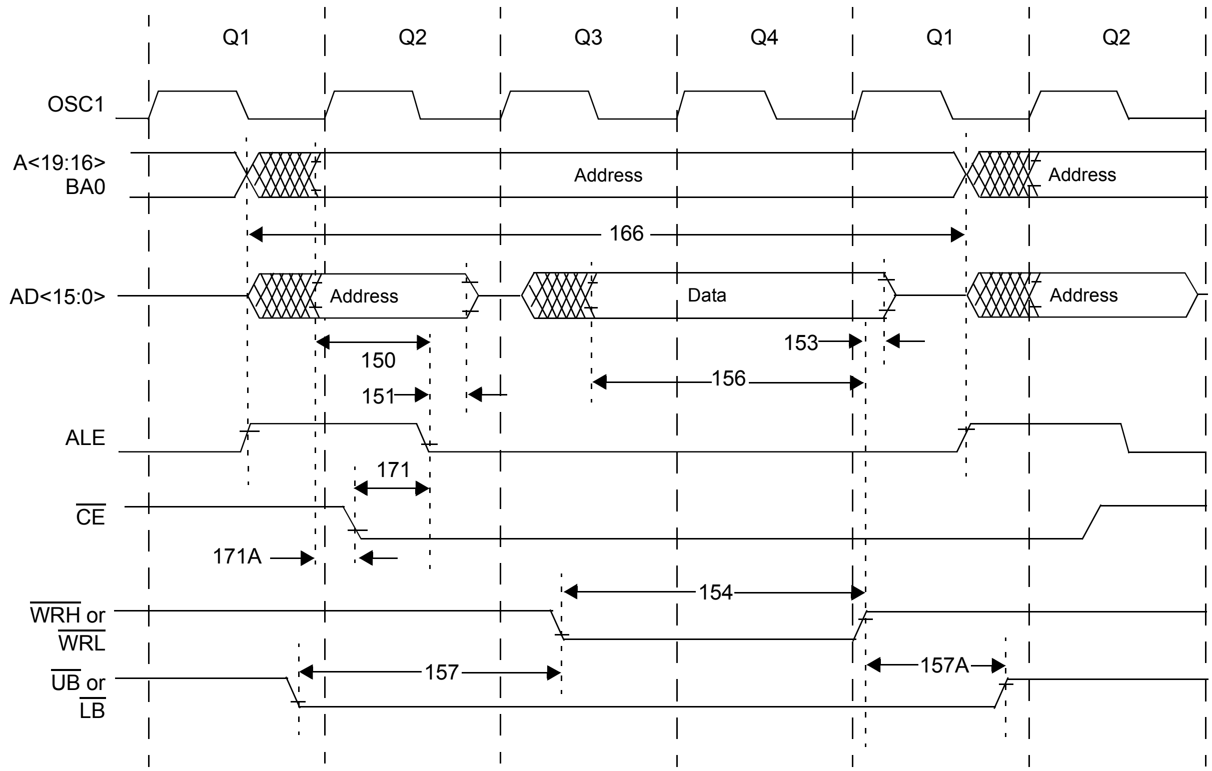
3: Standard, low-cost 32 kHz crystals have an operating temperature range of -10°C to +70°C. Extended temperature crystals are available at a much higher cost.

4: Voltage regulator disabled (ENVREG = 0, tied to VSS, RETEN (CONFIG1L<0>) = 1).

5: Voltage regulator enabled (ENVREG = 1, tied to VDD, SRETEN (WDTCON<4>) = 1 and RETEN (CONFIG1L<0>) = 0).

6: 48 MHz, maximum frequency at +125ºC.

FIGURE 31-8: PROGRAM MEMORY WRITE TIMING DIAGRAM



Operating Conditions: 2.0V < VCC < 3.6V, -40°C < TA < +125°C unless otherwise stated.

TABLE 31-12: PROGRAM MEMORY WRITE TIMING REQUIREMENTS

Param. No	Symbol	Characteristics	Min	Typ	Max	Units
150	TadV2alL	Address Out Valid to ALE ↓ (address setup time)	0.25 TCY – 10	—	—	ns
151	TalL2adl	ALE ↓ to Address Out Invalid (address hold time)	5	—	—	ns
153	TwrH2adl	$\overline{\text{WRn}}$ ↑ to Data Out Invalid (data hold time)	5	—	—	ns
154	TwrL	$\overline{\text{WRn}}$ Pulse Width	0.5 TCY – 5	0.5 TCY	—	ns
156	TadV2wrH	Data Valid before $\overline{\text{WRn}}$ ↑ (data setup time)	0.5 TCY – 10	—	—	ns
157	TbsV2wrL	Byte Select Valid before $\overline{\text{WRn}}$ ↓ (byte select setup time)	0.25 TCY	—	—	ns
157A	TwrH2bsl	$\overline{\text{WRn}}$ ↑ to Byte Select Invalid (byte select hold time)	0.125 TCY – 5	—	—	ns
166	TalH2alH	ALE ↑ to ALE ↑ (cycle time)	—	TCY	—	ns
171	TalH2csL	Chip Enable Active to ALE ↓	0.25 TCY – 20	—	—	ns
171A	TubL2oeH	AD Valid to Chip Enable Active	—	—	10	ns