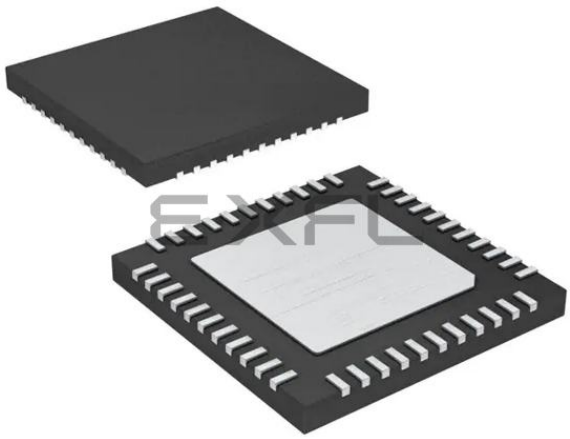


Welcome to E-XFL.COM

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	ECANbus, I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	35
Program Memory Size	64KB (32K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	3.6K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.5V
Data Converters	A/D 11x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-VQFN Exposed Pad
Supplier Device Package	44-QFN (8x8)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f46k80-i-ml

TABLE 1-4: PIC18F2XK80 I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	QFN	SSOP/ SPDIP /SOIC			
					PORTA is a bidirectional I/O port.
RA0/CVREF/AN0/ULPWU	27	2			
RA0			I/O	ST/ CMOS	General purpose I/O pin.
CVREF			O	Analog	Comparator reference voltage output.
AN0			I	Analog	Analog Input 0.
ULPWU			I	Analog	Ultra Low-Power Wake-up input.
RA1/AN1	28	3			
RA1			I/O	ST/ CMOS	Digital I/O.
AN1			I	Analog	Analog Input 1.
RA2/VREF-/AN2	1	4			
RA2			I/O	ST/ CMOS	Digital I/O.
VREF-			I	Analog	A/D reference voltage (low) input.
AN2			I	Analog	Analog Input 2.
RA3/VREF+/AN3	2	5			
RA3			I/O	ST/ CMOS	Digital I/O.
VREF+			I	Analog	A/D reference voltage (high) input.
AN3			I	Analog	Analog Input 3.
RA5/AN4/C2INB/HLVDIN/ T1CKI/SS/CTMUI	4	7			
RA5			I/O	ST/ CMOS	Digital I/O.
AN4			I	Analog	Analog Input 4.
C2INB			I	Analog	Comparator 2 Input B.
HLVDIN			I	Analog	High/Low-Voltage Detect input.
T1CKI			I	ST	Timer1 clock input.
SS			I	ST	SPI slave select input.
CTMUI					CTMU pulse generator charger for the C2INB.

Legend: CMOS = CMOS compatible input or output; I^2C™ = I^2C/SMBus input buffer
ST = Schmitt Trigger input with CMOS levels; Analog = Analog input
I = Input; O = Output
P = Power

TABLE 1-4: PIC18F2XK80 I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	QFN	SSOP/ SPDIP /SOIC			
RC7/CANRX/RX1/DT1/ CCP4	15	18			
RC7			I/O	ST/ CMOS	Digital I/O.
CANRX			I	ST	CAN bus RX.
RX1			I	ST	EUSART asynchronous receive.
DT1			I/O	ST	EUSART synchronous data. (See related TX2/CK2.)
CCP4			I/O	ST CMOS	Capture 4 input/Compare 4 output/PWM4 output.
VSS	5	8	P		
VSS					Ground reference for logic and I/O pins.
VSS	16	19			
VSS					Ground reference for logic and I/O pins.
VDDCORE/VCAP	3	6	P		
VDDCORE					External filter capacitor connection.
VCAP					External filter capacitor connection
VDD	17	20	P		
VDD					Positive supply for logic and I/O pins.

Legend: CMOS = CMOS compatible input or output
ST = Schmitt Trigger input with CMOS levels
I = Input
P = Power
I^2C™ = I^2C/SMBus input buffer
Analog = Analog input
O = Output

8.0 DATA EEPROM MEMORY

The data EEPROM is a nonvolatile memory array, separate from the data RAM and program memory, that is used for long-term storage of program data. It is not directly mapped in either the register file or program memory space, but is indirectly addressed through the Special Function Registers (SFRs). The EEPROM is readable and writable during normal operation over the entire VDD range.

Five SFRs are used to read and write to the data EEPROM, as well as the program memory. They are:

- EECON1
- EECON2
- EEDATA
- EEADR
- EEADRH

The data EEPROM allows byte read and write. When interfacing to the data memory block, EEDATA holds the 8-bit data for read/write and the EEADRH:EEADR register pair holds the address of the EEPROM location being accessed.

The EEPROM data memory is rated for high erase/write cycle endurance. A byte write automatically erases the location and writes the new data (erase-before-write). The write time is controlled by an on-chip timer; it will vary with voltage and temperature, as well as from chip-to-chip. Please refer to Parameter D122 (Table 31-1 in **Section 31.0 "Electrical Characteristics"**) for exact limits.

8.1 EEADR and EEADRH Registers

The EEADRH:EEADR register pair is used to address the data EEPROM for read and write operations. EEADRH holds the two MSbs of the address; the upper 6 bits are ignored. The 10-bit range of the pair can address a memory range of 1024 bytes (00h to 3FFh).

8.2 EECON1 and EECON2 Registers

Access to the data EEPROM is controlled by two registers: EECON1 and EECON2. These are the same registers which control access to the program memory and are used in a similar manner for the data EEPROM.

The EECON1 register (Register 8-1) is the control register for data and program memory access. Control bit, EEPGD, determines if the access will be to program memory or data EEPROM memory. When clear, operations will access the data EEPROM memory. When set, program memory is accessed.

Control bit, CFGS, determines if the access will be to the Configuration registers or to program memory/data EEPROM memory. When set, subsequent operations access Configuration registers. When CFGS is clear, the EEPGD bit selects either program Flash or data EEPROM memory.

The WREN bit, when set, will allow a write operation. On power-up, the WREN bit is clear. The WRERR bit is set in hardware when the WREN bit is set and cleared, when the internal programming timer expires and the write operation is complete.

Note:	During normal operation, the WRERR is read as '1'. This can indicate that a write operation was prematurely terminated by a Reset, or a write operation was attempted improperly.

The WR control bit initiates write operations. The bit cannot be cleared, only set, in software; it is cleared in hardware at the completion of the write operation.

Note:	The EEIF interrupt flag bit (PIR4<6>) is set when the write is complete. It must be cleared in software.

Control bits, RD and WR, start read and erase/write operations, respectively. These bits are set by firmware and cleared by hardware at the completion of the operation.

The RD bit cannot be set when accessing program memory (EEPGD = 1). Program memory is read using table read instructions. See **Section 7.1 "Table Reads and Table Writes"** regarding table reads.

The EECON2 register is not a physical register. It is used exclusively in the memory write and erase sequences. Reading EECON2 will read all '0's.

9.0 8 x 8 HARDWARE MULTIPLIER

9.1 Introduction

All PIC18 devices include an 8 x 8 hardware multiplier as part of the ALU. The multiplier performs an unsigned operation and yields a 16-bit result that is stored in the product register pair, PRODH:PRODL. The multiplier's operation does not affect any flags in the STATUS register.

Making multiplication a hardware operation allows it to be completed in a single instruction cycle. This has the advantages of higher computational throughput and reduced code size for multiplication algorithms and allows PIC18 devices to be used in many applications previously reserved for digital-signal processors. A comparison of various hardware and software multiply operations, along with the savings in memory and execution time, is shown in Table 9-1.

9.2 Operation

Example 9-1 shows the instruction sequence for an 8 x 8 unsigned multiplication. Only one instruction is required when one of the arguments is already loaded in the WREG register.

Example 9-2 shows the sequence to do an 8 x 8 signed multiplication. To account for the sign bits of the arguments, each argument's Most Significant bit (MSb) is tested and the appropriate subtractions are done.

EXAMPLE 9-1: 8 x 8 UNSIGNED MULTIPLY ROUTINE

```
MOVF    ARG1, W     ;
MULWF   ARG2        ; ARG1 * ARG2 ->
                    ; PRODH:PRODL
```

EXAMPLE 9-2: 8 x 8 SIGNED MULTIPLY ROUTINE

```
MOVF    ARG1, W
MULWF   ARG2        ; ARG1 * ARG2 ->
                    ; PRODH:PRODL
BTFSC   ARG2, SB    ; Test Sign Bit
SUBWF   PRODH, F    ; PRODH = PRODH
                    ;         - ARG1
MOVF    ARG2, W
BTFSC   ARG1, SB    ; Test Sign Bit
SUBWF   PRODH, F    ; PRODH = PRODH
                    ;         - ARG2
```

TABLE 9-1: PERFORMANCE COMPARISON FOR VARIOUS MULTIPLY OPERATIONS

Routine	Multiply Method	Program Memory (Words)	Cycles (Max)	Time			
				@ 64 MHz	@ 48 MHz	@ 10 MHz	@ 4 MHz
8 x 8 unsigned	Without hardware multiply	13	69	4.3 μs	5.7 μs	27.6 μs	69 μs
	Hardware multiply	1	1	62.5 ns	83.3 ns	400 ns	1 μs
8 x 8 signed	Without hardware multiply	33	91	5.6 μs	7.5 μs	36.4 μs	91 μs
	Hardware multiply	6	6	375 ns	500 ns	2.4 μs	6 μs
16 x 16 unsigned	Without hardware multiply	21	242	15.1 μs	20.1 μs	96.8 μs	242 μs
	Hardware multiply	28	28	1.7 μs	2.3 μs	11.2 μs	28 μs
16 x 16 signed	Without hardware multiply	52	254	15.8 μs	21.2 μs	101.6 μs	254 μs
	Hardware multiply	35	40	2.5 μs	3.3 μs	16.0 μs	40 μs

NOTES:

11.9 Parallel Slave Port

PORTD can function as an 8-bit-wide Parallel Slave Port (PSP), or microprocessor port, when control bit, PSPMODE (PSPCON<4>), is set. The port is asynchronously readable and writable by the external world through the $\overline{RD}$ control input pin (RE0/AN5/$\overline{RD}$) and $\overline{WR}$ control input pin (RE1/AN6/C1OUT/$\overline{WR}$).

Note:	The Parallel Slave Port is available only on 40/44-pin and 64-pin devices.

The PSP can directly interface to an 8-bit microprocessor data bus. The external microprocessor can read or write the PORTD latch as an eight-bit latch.

Setting bit, PSPMODE, enables port pin, RE0/AN5/$\overline{RD}$, to be the $\overline{RD}$ input, RE1/AN6/C1OUT/$\overline{WR}$ to be the $\overline{WR}$ input and RE2/AN7/C2OUT/$\overline{CS}$ to be the $\overline{CS}$ (Chip Select) input. For this functionality, the corresponding data direction bits of the TRISE register (TRISE<2:0>) must be configured as inputs (= 111).

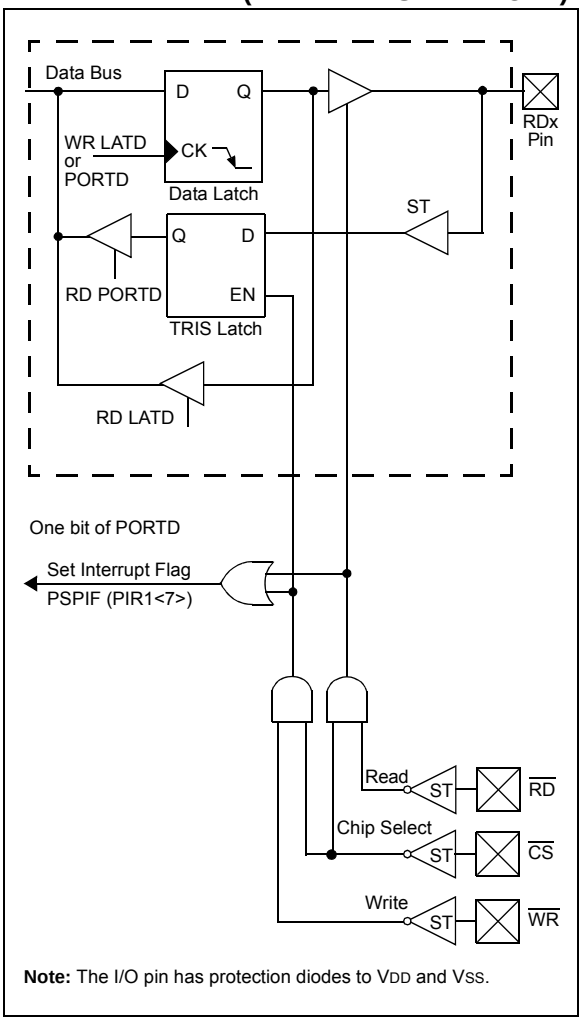
A write to the PSP occurs when both the $\overline{CS}$ and $\overline{WR}$ lines are first detected low and ends when either are detected high. The PSPIF and IBF flag bits (PIR1<7> and PSPCON<7>, respectively) are set when the write ends.

A read from the PSP occurs when both the $\overline{CS}$ and $\overline{RD}$ lines are first detected low. The data in PORTD is read out and the OBF bit (PSPCON<6>) is set. If the user writes new data to PORTD to set OBF, the data is immediately read out, but the OBF bit is not set.

When either the $\overline{CS}$ or $\overline{RD}$ line is detected high, the PORTD pins return to the input state and the PSPIF bit is set. User applications should wait for PSPIF to be set before servicing the PSP. When this happens, the IBF and OBF bits can be polled and the appropriate action taken.

The timing for the control signals in Write and Read modes is shown in Figure 11-4 and Figure 11-5, respectively.

FIGURE 11-3: PORTD AND PORTE BLOCK DIAGRAM (PARALLEL SLAVE PORT)



Note: The I/O pin has protection diodes to VDD and VSS.

13.1 Timer0 Operation

Timer0 can operate as either a timer or a counter. The mode is selected with the T0CS bit (T0CON<5>). In Timer mode (T0CS = 0), the module increments on every clock by default unless a different prescaler value is selected (see **Section 13.3 "Prescaler"**). If the TMR0 register is written to, the increment is inhibited for the following two instruction cycles. The user can work around this by writing an adjusted value to the TMR0 register.

The Counter mode is selected by setting the T0CS bit (= 1). In this mode, Timer0 increments either on every rising edge or falling edge of the T0CKI pin. The incrementing edge is determined by the Timer0 Source Edge Select bit, T0SE (T0CON<4>); clearing this bit selects the rising edge. Restrictions on the external clock input are discussed below.

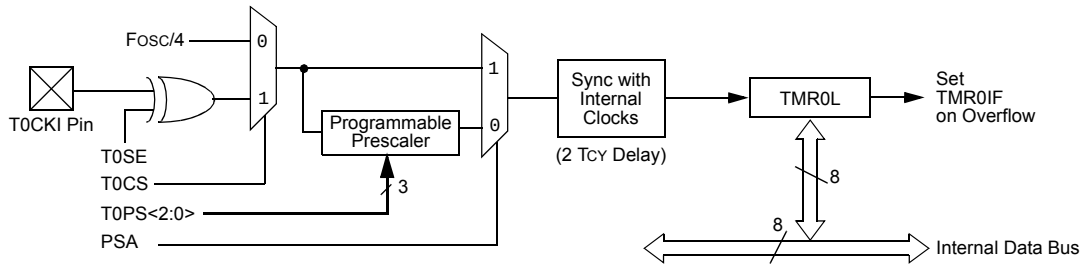
An external clock source can be used to drive Timer0; however, it must meet certain requirements to ensure that the external clock can be synchronized with the internal phase clock (TOSC). There is a delay between synchronization and the onset of incrementing the timer/counter.

13.2 Timer0 Reads and Writes in 16-Bit Mode

TMR0H is not the actual high byte of Timer0 in 16-bit mode. It is actually a buffered version of the real high byte of Timer0, which is not directly readable nor writable. (See Figure 13-2.) TMR0H is updated with the contents of the high byte of Timer0 during a read of TMR0L. This provides the ability to read all 16 bits of Timer0 without having to verify that the read of the high and low byte were valid, due to a rollover between successive reads of the high and low byte.

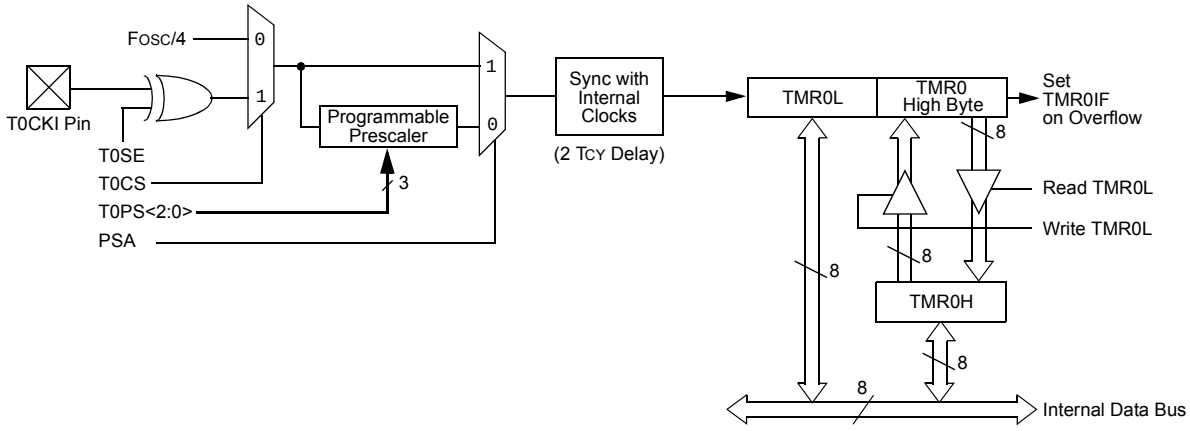
Similarly, a write to the high byte of Timer0 must also take place through the TMR0H Buffer register. The high byte is updated with the contents of TMR0H when a write occurs to TMR0L. This allows all 16 bits of Timer0 to be updated at once.

FIGURE 13-1: TIMER0 BLOCK DIAGRAM (8-BIT MODE)



Note: Upon Reset, Timer0 is enabled in 8-bit mode with clock input from T0CKI max. prescale.

FIGURE 13-2: TIMER0 BLOCK DIAGRAM (16-BIT MODE)



Note: Upon Reset, Timer0 is enabled in 8-bit mode with clock input from T0CKI max. prescale.

REGISTER 14-1: T1CON: TIMER1 CONTROL REGISTER (CONTINUED)

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.

18.2 CTMU Operation

The CTMU works by using a fixed current source to charge a circuit. The type of circuit depends on the type of measurement being made.

In the case of charge measurement, the current is fixed and the amount of time the current is applied to the circuit is fixed. The amount of voltage read by the A/D becomes a measurement of the circuit's capacitance.

In the case of time measurement, the current, as well as the capacitance of the circuit, is fixed. In this case, the voltage read by the A/D is representative of the amount of time elapsed from the time the current source starts and stops charging the circuit.

If the CTMU is being used as a time delay, both capacitance and current source are fixed, as well as the voltage supplied to the comparator circuit. The delay of a signal is determined by the amount of time it takes the voltage to charge to the comparator threshold voltage.

18.2.1 THEORY OF OPERATION

The operation of the CTMU is based on the equation for charge:

$$I = C \bullet \frac{dV}{dT}$$

More simply, the amount of charge measured in coulombs in a circuit is defined as current in amperes (I) multiplied by the amount of time in seconds that the current flows (t). Charge is also defined as the capacitance in farads (C) multiplied by the voltage of the circuit (V). It follows that:

$$I \bullet t = C \bullet V$$

The CTMU module provides a constant, known current source. The A/D Converter is used to measure (V) in the equation, leaving two unknowns: capacitance (C) and time (t). The above equation can be used to calculate capacitance or time, by either the relationship using the known fixed capacitance of the circuit:

$$t = (C \bullet V)/I$$

or by:

$$C = (I \bullet t)/V$$

using a fixed time that the current source is applied to the circuit.

18.2.2 CURRENT SOURCE

At the heart of the CTMU is a precision current source, designed to provide a constant reference for measurements. The level of current is user-selectable across three ranges, or a total of two orders of magnitude, with the ability to trim the output in ±2% increments (nominal). The current range is selected by the IRNG<1:0> bits (CTMUICON<1:0>), with a value of '01' representing the lowest range.

Current trim is provided by the ITRIM<5:0> bits (CTMUICON<7:2>). These six bits allow trimming of the current source in steps of approximately 2% per step. Half of the range adjusts the current source positively and the other half reduces the current source. A value of '000000' is the neutral position (no change). A value of '100001' is the maximum negative adjustment (approximately -62%) and '011111' is the maximum positive adjustment (approximately +62%).

18.2.3 EDGE SELECTION AND CONTROL

CTMU measurements are controlled by edge events occurring on the module's two input channels. Each channel, referred to as Edge 1 and Edge 2, can be configured to receive input pulses from one of the edge input pins (CTED1 and CTED2) or CCPx Special Event Triggers (ECCP1 and CCP2). The input channels are level-sensitive, responding to the instantaneous level on the channel rather than a transition between levels. The inputs are selected using the EDG1SEL and EDG2SEL bit pairs (CTMUCONL<3:2>, 6:5>).

In addition to source, each channel can be configured for event polarity using the EDGE2POL and EDGE1POL bits (CTMUCONL<7,4>). The input channels can also be filtered for an edge event sequence (Edge 1 occurring before Edge 2) by setting the EDGSEQEN bit (CTMUCONH<2>).

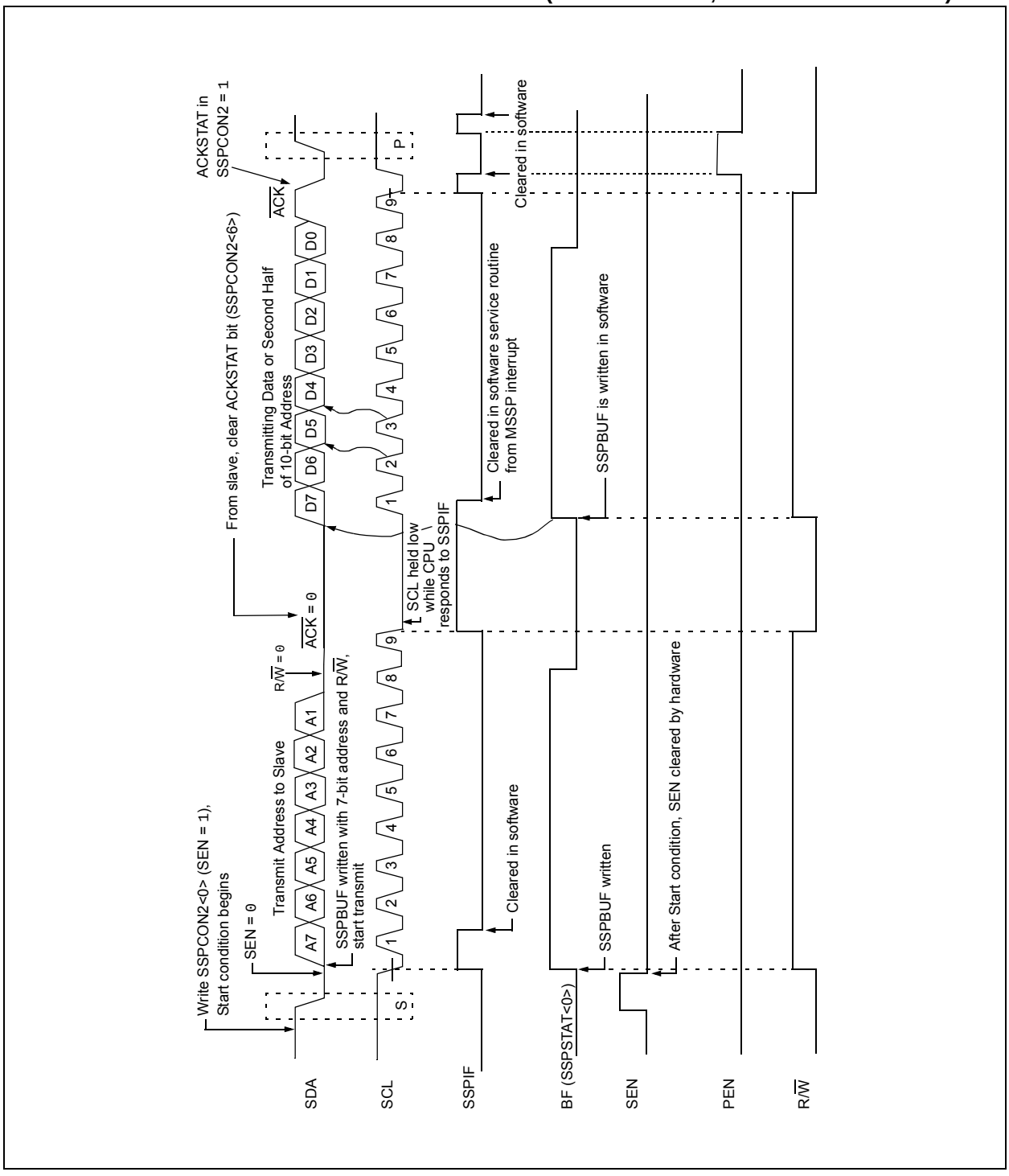
18.2.4 EDGE STATUS

The CTMUCONL register also contains two status bits, EDG2STAT and EDG1STAT (CTMUCONL<1:0>). Their primary function is to show if an edge response has occurred on the corresponding channel. The CTMU automatically sets a particular bit when an edge response is detected on its channel. The level-sensitive nature of the input channels also means that the status bits become set immediately if the channel's configuration is changed and matches the channel's current state.

The module uses the edge status bits to control the current source output to external analog modules (such as the A/D Converter). Current is only supplied to external modules when only one (not both) of the status bits is set. Current is shut off when both bits are either set or cleared. This allows the CTMU to measure current only during the interval between edges. After both status bits are set, it is necessary to clear them before another measurement is taken. Both bits should be cleared simultaneously, if possible, to avoid re-enabling the CTMU current source.

In addition to being set by the CTMU hardware, the edge status bits can also be set by software. This permits a user application to manually enable or disable the current source. Setting either (but not both) of the bits enables the current source. Setting or clearing both bits at once disables the source.

FIGURE 21-23: I²C™ MASTER MODE WAVEFORM (TRANSMISSION, 7 OR 10-BIT ADDRESS)



27.2 CAN Module Registers

Note:	Not all CAN registers are available in the Access Bank.

There are many control and data registers associated with the CAN module. For convenience, their descriptions have been grouped into the following sections:

- Control and Status Registers
- Dedicated Transmit Buffer Registers
- Dedicated Receive Buffer Registers
- Programmable TX/RX and Auto RTR Buffers
- Baud Rate Control Registers
- I/O Control Register
- Interrupt Status and Control Registers

Detailed descriptions of each register and their usage are described in the following sections.

27.2.1 CAN CONTROL AND STATUS REGISTERS

The registers described in this section control the overall operation of the CAN module and show its operational status.

REGISTER 27-47: RXFBCONn: RECEIVE FILTER BUFFER CONTROL REGISTER 'n'(1)

RXFBCON0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	F1BP_3	F1BP_2	F1BP_1	F1BP_0	F0BP_3	F0BP_2	F0BP_1	F0BP_0
RXFBCON1	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0	R/W-1
	F3BP_3	F3BP_2	F3BP_1	F3BP_0	F2BP_3	F2BP_2	F2BP_1	F2BP_0
RXFBCON2	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0	R/W-1
	F5BP_3	F5BP_2	F5BP_1	F5BP_0	F4BP_3	F4BP_2	F4BP_1	F4BP_0
RXFBCON3	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	F7BP_3	F7BP_2	F7BP_1	F7BP_0	F6BP_3	F6BP_2	F6BP_1	F6BP_0
RXFBCON4	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	F9BP_3	F9BP_2	F9BP_1	F9BP_0	F8BP_3	F8BP_2	F8BP_1	F8BP_0
RXFBCON5	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	F11BP_3	F11BP_2	F11BP_1	F11BP_0	F10BP_3	F10BP_2	F10BP_1	F10BP_0
RXFBCON6	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	F13BP_3	F13BP_2	F13BP_1	F13BP_0	F12BP_3	F12BP_2	F12BP_1	F12BP_0
RXFBCON7	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	F15BP_3	F15BP_2	F15BP_1	F15BP_0	F14BP_3	F14BP_2	F14BP_1	F14BP_0
	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 **F<15:2>BP_<3:0>:** Filter n Buffer Pointer Nibble bits

0000 = Filter n is associated with RXB0
0001 = Filter n is associated with RXB1
0010 = Filter n is associated with B0
0011 = Filter n is associated with B1
...
0111 = Filter n is associated with B5
1111-1000 = Reserved

Note 1: This register is available in Mode 1 and 2 only.

REGISTER 27-51: MSEL3: MASK SELECT REGISTER 3(1)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
FIL15_1	FIL15_0	FIL14_1	FIL14_0	FIL13_1	FIL13_0	FIL12_1	FIL12_0
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 **FIL15_<1:0>:** Filter 15 Select bits 1 and 0
- 11 = No mask
- 10 = Filter 15
- 01 = Acceptance Mask 1
- 00 = Acceptance Mask 0

bit 5-4 **FIL14_<1:0>:** Filter 14 Select bits 1 and 0
- 11 = No mask
- 10 = Filter 15
- 01 = Acceptance Mask 1
- 00 = Acceptance Mask 0

bit 3-2 **FIL13_<1:0>:** Filter 13 Select bits 1 and 0
- 11 = No mask
- 10 = Filter 15
- 01 = Acceptance Mask 1
- 00 = Acceptance Mask 0

bit 1-0 **FIL12_<1:0>:** Filter 12 Select bits 1 and 0
- 11 = No mask
- 10 = Filter 15
- 01 = Acceptance Mask 1
- 00 = Acceptance Mask 0

Note 1: This register is available in Mode 1 and 2 only.

REGISTER 28-13: DEVID1: DEVICE ID REGISTER 1 FOR THE PIC18F66K80 FAMILY

R	R	R	R	R	R	R	R
DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0
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 **DEV<2:0>:** Device ID bits

These bits are used with the DEV<10:3> bits in the Device ID Register 2 to identify the part number:
000 = PIC18F46K80, PIC18LF26K80
001 = PIC18F26K80, PIC18LF65K80
010 = PIC18F65K80, PIC18LF45K80
011 = PIC18F45K80, PIC18LF25K80
100 = PIC18F25K80
110 = PIC18LF66K80
111 = PIC18F66K80, PIC18LF46K80

bit 4-0 **REV<4:0>:** Revision ID bits

These bits are used to indicate the device revision.

REGISTER 28-14: DEVID2: DEVICE ID REGISTER 2 FOR THE PIC18F66K80 FAMILY

R	R	R	R	R	R	R	R
DEV10(1)	DEV9(1)	DEV8(1)	DEV7(1)	DEV6(1)	DEV5(1)	DEV4(1)	DEV3(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 7-0 **DEV<10:3>:** Device ID bits(1)

These bits are used with the DEV<2:0> bits in the Device ID Register 1 to identify the part number.

Note 1: These values for DEV<10:3> may be shared with other devices. The specific device is always identified by using the entire DEV<10:0> bit sequence.

RCALL **Relative Call**

Syntax: RCALL n

Operands: $-1024 \leq n \leq 1023$

Operation: (PC) + 2 → TOS,
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1101	1nnn	nnnn	nnnn

Description: Subroutine call with a jump up to 1K from the current location. First, return address (PC + 2) is pushed onto the stack. Then, add the 2's complement number '2n' to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is a two-cycle instruction.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'n' PUSH PC to stack	Process Data	Write to PC
No operation	No operation	No operation	No operation

Example: HERE RCALL Jump

Before Instruction
PC = Address (HERE)
After Instruction
PC = Address (Jump)
TOS = Address (HERE + 2)

RESET **Reset**

Syntax: RESET

Operands: None

Operation: Reset all registers and flags that are affected by a $\overline{\text{MCLR}}$ Reset.

Status Affected: All

Encoding:

0000	0000	1111	1111

Description: This instruction provides a way to execute a $\overline{\text{MCLR}}$ Reset in software.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Start reset	No operation	No operation

Example: RESET

After Instruction
Registers = Reset Value
Flags* = Reset Value

RETURN **Return from Subroutine**

Syntax: RETURN {s}

Operands: s ∈ [0,1]

Operation: (TOS) → PC;
if s = 1,
(WS) → W,
(STATUSS) → STATUS,
(BSRS) → BSR,
PCLATU, PCLATH are unchanged

Status Affected: None

Encoding:

0000	0000	0001	001s

Description: Return from subroutine. The stack is popped and the top of the stack (TOS) is loaded into the Program Counter. If 's' = 1, the contents of the shadow registers WS, STATUSS and BSRS are loaded into their corresponding registers W, STATUS and BSR. If 's' = 0, no update of these registers occurs (default).

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	Process Data	POP PC from stack
No operation	No operation	No operation	No operation

Example: RETURN

After Instruction:
PC = TOS

RLCF **Rotate Left f through Carry**

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

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

Operation: (f<n>) → dest<n + 1>,
(f<7>) → C,
(C) → dest<0>

Status Affected: C, N, Z

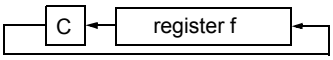
Encoding:

0011	01da	ffff	ffff

Description: The contents of register 'f' are rotated one bit to the left through the Carry flag. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is stored back in register 'f' (default).

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

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.



Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: RLCF REG, 0, 0

Before Instruction
REG = 1110 0110
C = 0
After Instruction
REG = 1110 0110
W = 1100 1100
C = 1

30.2 MPLAB C Compilers for Various Device Families

The MPLAB C Compiler code development systems are complete ANSI C compilers for Microchip's PIC18, PIC24 and PIC32 families of microcontrollers and the dsPIC30 and dsPIC33 families of digital signal controllers. These compilers provide powerful integration capabilities, superior code optimization and ease of use.

For easy source level debugging, the compilers provide symbol information that is optimized to the MPLAB IDE debugger.

30.3 HI-TECH C for Various Device Families

The HI-TECH C Compiler code development systems are complete ANSI C compilers for Microchip's PIC family of microcontrollers and the dsPIC family of digital signal controllers. These compilers provide powerful integration capabilities, omniscient code generation and ease of use.

For easy source level debugging, the compilers provide symbol information that is optimized to the MPLAB IDE debugger.

The compilers include a macro assembler, linker, preprocessor, and one-step driver, and can run on multiple platforms.

30.4 MPASM Assembler

The MPASM Assembler is a full-featured, universal macro assembler for PIC10/12/16/18 MCUs.

The MPASM Assembler generates relocatable object files for the MPLINK Object Linker, Intel® standard HEX files, MAP files to detail memory usage and symbol reference, absolute LST files that contain source lines and generated machine code and COFF files for debugging.

The MPASM Assembler features include:

- Integration into MPLAB IDE projects
- User-defined macros to streamline assembly code
- Conditional assembly for multi-purpose source files
- Directives that allow complete control over the assembly process

30.5 MPLINK Object Linker/ MPLIB Object Librarian

The MPLINK Object Linker combines relocatable objects created by the MPASM Assembler and the MPLAB C18 C Compiler. It can link relocatable objects from precompiled libraries, using directives from a linker script.

The MPLIB Object Librarian manages the creation and modification of library files of precompiled code. When a routine from a library is called from a source file, only the modules that contain that routine will be linked in with the application. This allows large libraries to be used efficiently in many different applications.

The object linker/library features include:

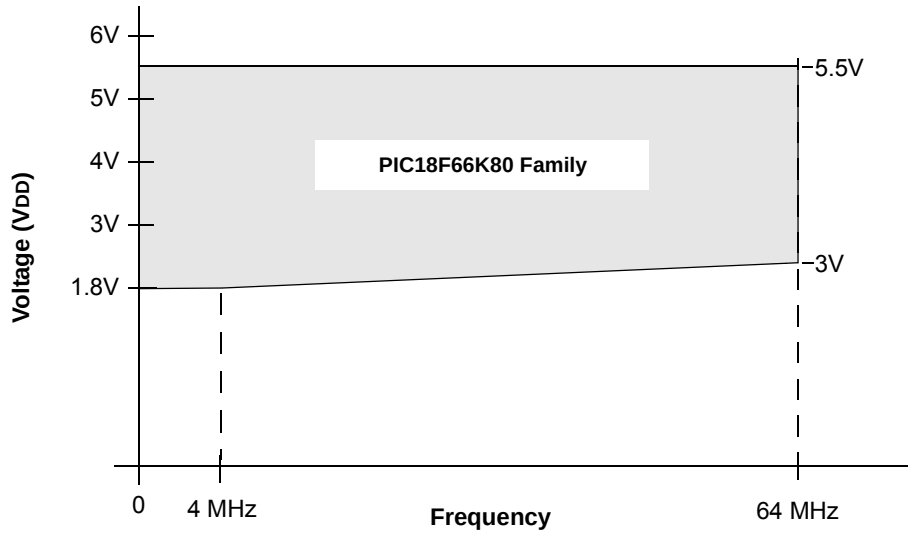
- Efficient linking of single libraries instead of many smaller files
- Enhanced code maintainability by grouping related modules together
- Flexible creation of libraries with easy module listing, replacement, deletion and extraction

30.6 MPLAB Assembler, Linker and Librarian for Various Device Families

MPLAB Assembler produces relocatable machine code from symbolic assembly language for PIC24, PIC32 and dsPIC devices. MPLAB C Compiler uses the assembler to produce its object file. The assembler generates relocatable object files that can then be archived or linked with other relocatable object files and archives to create an executable file. Notable features of the assembler include:

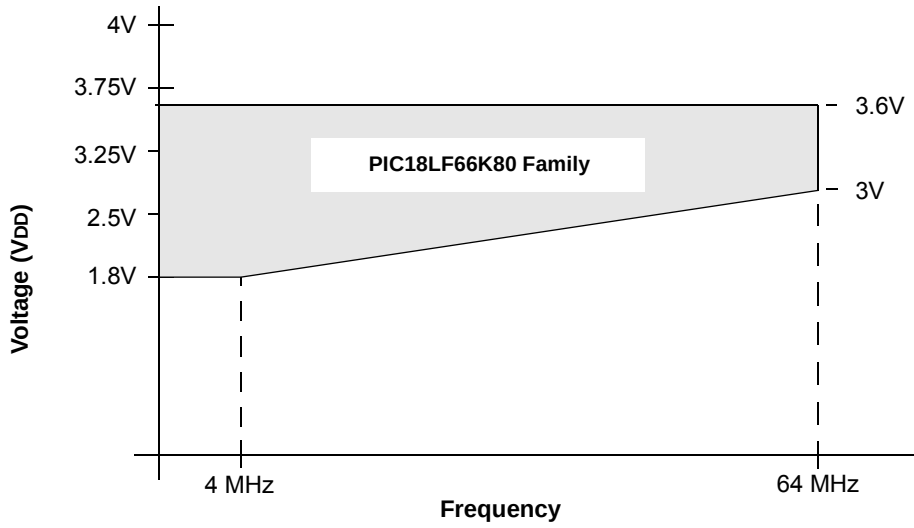
- Support for the entire device instruction set
- Support for fixed-point and floating-point data
- Command line interface
- Rich directive set
- Flexible macro language
- MPLAB IDE compatibility

FIGURE 31-1: VOLTAGE-FREQUENCY GRAPH, REGULATOR ENABLED (INDUSTRIAL/EXTENDED)(1)



Note 1: For VDD values 1.8V to 3V, FMAX = (VDD – 1.72)/0.02 MHz.

FIGURE 31-2: VOLTAGE-FREQUENCY GRAPH, REGULATOR DISABLED (INDUSTRIAL/EXTENDED)(1,2)



Note 1: When the on-chip voltage regulator is disabled, VDD must be maintained so that VDD ≤ 3.6V.
2: For VDD values 1.8V to 3V, FMAX = (VDD – 1.72)/0.02 MHz.

31.2 DC Characteristics: Power-Down and Supply Current PIC18F66K80 Family (Industrial/Extended) (Continued)

PIC18F66K80 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)						
	PIC18**LF**XXK80	20	70	µA	-40°C	VDD = 1.8V(4) Regulator disabled	FOSC = 1 MHz (**PRI_IDLE** mode, EC oscillator)
		20	70	µA	+25°C		
		20	70	µA	+60°C		
		25	80	µA	+85°C		
		30	100	µA	+125°C		
	PIC18**LF**XXK80	37	120	µA	-40°C	VDD = 3.3V(4) Regulator disabled	
		37	120	µA	+25°C		
		37	120	µA	+60°C		
		40	130	µA	+85°C		
		45	150	µA	+125°C		
	PIC18**F**XXK80	85	140	µA	-40°C	VDD = 3.3V(5) Regulator enabled	
		100	140	µA	+25°C		
		105	140	µA	+60°C		
		110	150	µA	+85°C		
		120	170	µA	+125°C		
	PIC18**F**XXK80	110	225	µA	-40°C	VDD = 5V(5) Regulator enabled	
		110	225	µA	+25°C		
		110	225	µA	+60°C		
		120	230	µA	+85°C		
		130	250	µA	+125°C		

Legend: Shading of rows is to assist in readability of the table.

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;
$\overline{MCLR}$ = 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: For LF devices, $\overline{RETEN}$ (CONFIG1L<0>) = 1.

5: For F devices, SRETEN (WDTCON<4>) = 1 and $\overline{RETEN}$ (CONFIG1L<0>) = 0.

S

T