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	24
Program Memory Size	32KB (16K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	3.6K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	A/D 8x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-SSOP (0.209", 5.30mm Width)
Supplier Device Package	28-SSOP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18lf25k80t-i-ss

1.1.5 EASY MIGRATION

Regardless of the memory size, all devices share the same rich set of peripherals, allowing for a smooth migration path as applications grow and evolve.

The consistent pinout scheme used throughout the entire family also aids in migrating to the next larger device. This is true when moving between the 28-pin, 40-pin, 44-pin and 64-pin members, or even jumping from smaller to larger memory devices.

The PIC18F66K80 family is also largely pin compatible with other PIC18 families, such as the PIC18F4580, PIC18F4680 and PIC18F8680 families of microcontrollers with an ECAN module. This allows a new dimension to the evolution of applications, allowing developers to select different price points within Microchip's PIC18 portfolio, while maintaining a similar feature set.

1.2 Other Special Features

- **Communications:** The PIC18F66K80 family incorporates a range of serial communication peripherals, including two Enhanced USARTs that support LIN/J2602, one Master SSP module capable of both SPI and I^2C™ (Master and Slave) modes of operation and an Enhanced CAN module.
- **CCP Modules:** PIC18F66K80 family devices incorporate four Capture/Compare/PWM (CCP) modules. Up to four different time bases can be used to perform several different operations at once.
- **ECCP Modules:** The PIC18F66K80 family has one Enhanced CCP (ECCP) module to maximize flexibility in control applications:
  - Up to four different time bases for performing several different operations at once
  - Up to four PWM outputs
  - Other beneficial features, such as polarity selection, programmable dead time, auto-shutdown and restart, and Half-Bridge and Full-Bridge Output modes
- **12-Bit A/D Converter:** The PIC18F66K80 family has a differential A/D. It incorporates programmable acquisition time, allowing for a channel to be selected and a conversion to be initiated without waiting for a sampling period, and thus, reducing code overhead.
- **Charge Time Measurement Unit (CTMU):** The CTMU is a flexible analog module that provides accurate differential time measurement between pulse sources, as well as asynchronous pulse generation.

  Together with other on-chip analog modules, the CTMU can precisely measure time, measure capacitance or relative changes in capacitance, or generate output pulses that are independent of the system clock.
- **LP Watchdog Timer (WDT):** This enhanced version incorporates a 22-bit prescaler, allowing an extended time-out range that is stable across operating voltage and temperature. See **Section 31.0 "Electrical Characteristics"** for time-out periods.

1.3 Details on Individual Family Members

Devices in the PIC18F66K80 family are available in 28-pin, 40/44-pin and 64-pin packages. Block diagrams for each package are shown in Figure 1-1, Figure 1-2 and Figure 1-3, respectively.

The devices are differentiated from each other in these ways:

- Flash Program Memory:
  - PIC18FX5K80 (PIC18F25K80, PIC18F45K80 and PIC18F45K80) – 32 Kbytes
  - PIC18FX6K80 (PIC18F26K80, PIC18F46K80 and PIC18F66K80) – 64 Kbytes
- I/O Ports:
  - PIC18F2XK80 (28-pin devices) – Three bidirectional ports
  - PIC18F4XK80 (40/44-pin devices) – Five bidirectional ports
  - PIC18F6XK80 (64-pin devices) – Seven bidirectional ports

All other features for devices in this family are identical. These are summarized in Table 1-1, Table 1-2 and Table 1-3.

The pinouts for all devices are listed in Table 1-4, Table 1-5 and Table 1-6.

4.1.3 CLOCK TRANSITIONS AND STATUS INDICATORS

The length of the transition between clock sources is the sum of two cycles of the old clock source and three to four cycles of the new clock source. This formula assumes that the new clock source is stable. The HF-INTOSC and MF-INTOSC are termed as INTOSC in this chapter.

Three bits indicate the current clock source and its status, as shown in Table 4-2. The three bits are:

- OSTS (OSCCON<3>)
- HFIOFS (OSCCON<2>)
- SOSCRUN (OSCCON2<6>)

TABLE 4-2: SYSTEM CLOCK INDICATOR

Main Clock Source	OSTS	HFIOFS or MFIOFS	SOSCRUN
Primary Oscillator	1	0	0
INTOSC (HF-INTOSC or MF-INTOSC)	0	1	0
Secondary Oscillator	0	0	1
MF-INTOSC or HF-INTOSC as Primary Clock Source	1	1	0
LF-INTOSC is Running or INTOSC is Not Yet Stable	0	0	0

When the OSTS bit is set, the primary clock is providing the device clock. When the HFIOFS or MFIOFS bit is set, the INTOSC output is providing a stable clock source to a divider that actually drives the device clock. When the SOSCRUN bit is set, the SOSC oscillator is providing the clock. If none of these bits are set, either the LF-INTOSC clock source is clocking the device or the INTOSC source is not yet stable.

If the internal oscillator block is configured as the primary clock source by the FOSC<3:0> Configuration bits (CONFIG1H<3:0>). Then, the OSTS and HFIOFS or MFIOFS bits can be set when in PRI_RUN or PRI_IDLE mode. This indicates that the primary clock (INTOSC output) is generating a stable output. Entering another INTOSC power-managed mode at the same frequency would clear the OSTS bit.

Note 1: Caution should be used when modifying a single IRCF bit. At a lower VDD, it is possible to select a higher clock speed than is supportable by that VDD. Improper device operation may result if the VDD/FOSC specifications are violated.

2: Executing a SLEEP instruction does not necessarily place the device into Sleep mode. It acts as the trigger to place the controller into either the Sleep mode, or one of the Idle modes, depending on the setting of the IDLEN bit.

4.1.4 MULTIPLE SLEEP COMMANDS

The power-managed mode that is invoked with the SLEEP instruction is determined by the setting of the IDLEN bit at the time the instruction is executed. If another SLEEP instruction is executed, the device will enter the power-managed mode specified by IDLEN at that time. If IDLEN has changed, the device will enter the new power-managed mode specified by the new setting.

4.2 Run Modes

In the Run modes, clocks to both the core and peripherals are active. The difference between these modes is the clock source.

4.2.1 PRI_RUN MODE

The PRI_RUN mode is the normal, full-power execution mode of the microcontroller. This is also the default mode upon a device Reset, unless Two-Speed Start-up is enabled. (For details, see **Section 28.4 “Two-Speed Start-up”**.) In this mode, the OSTS bit is set. The HFIOFS or MFIOFS bit may be set if the internal oscillator block is the primary clock source. (See **Section 3.2 “Control Registers”**.)

4.2.2 SEC_RUN MODE

The SEC_RUN mode is the compatible mode to the “clock-switching” feature offered in other PIC18 devices. In this mode, the CPU and peripherals are clocked from the SOSC oscillator. This enables lower power consumption while retaining a high-accuracy clock source.

SEC_RUN mode is entered by setting the SCS<1:0> bits to ‘01’. The device clock source is switched to the SOSC oscillator (see Figure 4-1), the primary oscillator is shut down, the SOSCRUN bit (OSCCON2<6>) is set and the OSTS bit is cleared.

Note: The SOSC oscillator can be enabled by setting the SOSCGO bit (OSCCON2<3>). If this bit is set, the clock switch to the SEC_RUN mode can switch immediately once SCS<1:0> are set to ‘01’.

On transitions from SEC_RUN mode to PRI_RUN mode, the peripherals and CPU continue to be clocked from the SOSC oscillator while the primary clock is started. When the primary clock becomes ready, a clock switch back to the primary clock occurs (see Figure 4-2). When the clock switch is complete, the SOSCRUN bit is cleared, the OSTS bit is set and the primary clock is providing the clock. The IDLEN and SCSx bits are not affected by the wake-up and the SOSC oscillator continues to run.

TABLE 5-4: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices			Power-on Reset, Brown-out Reset	MCLR Resets, WDT Reset, RESET Instruction, Stack Resets	Wake-up via WDT or Interrupt
BAUDCON2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	01x0 0-00	01x0 0-00	uuuu u-uu
IPR4	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1111 -111	1111 -111	uuuu -uuu
PIR4	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 -000	0000 -000	uuuu -uuu
PIE4	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 -000	0000 -000	uuuu -uuu
CVRCON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
CMSTAT	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xx-- ----	xx-- ----	uu-- ----
TMR3H	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR3L	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
T3CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
T3GCON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0x00	0000 0x00	uuuu u-uu
SPBRG1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
RCREG1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
TXREG1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
TXSTA1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0010	0000 0010	uuuu uuuu
RCSTA1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 000x	0000 000x	uuuu uuuu
T1GCON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0x00	0000 0x00	uuuu u-uu
PR4	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1111 1111	1111 1111	uuuu uuuu
HLVDCON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
BAUDCON1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	01x0 0-00	01x0 0-00	uuuu u-uu
RCSTA2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 000x	0000 000x	uuuu uuuu
IPR3	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	--11 111-	--11 111-	--uu uuu-
PIR3	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	--00 000-	--x0 xxx-	--uu uuu-
PIE3	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	--00 000-	0000 0000	uuuu uuuu
IPR2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1--- 1111	1--- 111x	u--- uuuu
PIR2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0--- 0000	0--- 000x	u--- uuuu(1)
PIE2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0--- 0000	0--- 0000	u--- uuuu
IPR1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1111 1111	1111 1111	uuuu uuuu
	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	-111 1111	-111 1111	-uuu uuuu
PIR1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu(1)
	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	-000 0000	-000 0000	-uuu uuuu
PIE1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	-000 0000	-000 0000	-uuu uuuu
PSTR1CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	00-0 0001	xx-x xxxx	—
OSCTUNE	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
REFOCON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0-00 0000	0-00 0000	u-uu uuuu

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

Note 1: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

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

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

4: See Table 5-3 for Reset value for specific conditions.

5: Bits 6 and 7 of PORTA, LATA and TRISA are enabled, depending on the oscillator mode selected. When not enabled as PORTA pins, they are disabled and read as '0'.

TABLE 5-4: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices			Power-on Reset, Brown-out Reset	MCLR Resets, WDT Reset, RESET Instruction, Stack Resets	Wake-up via WDT or Interrupt
CCPTMRS	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	---0 0000	---x xxxx	---u uuuu
TRISG	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	---1 1111	---1 1111	---u uuuu
TRISF	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1111 1111	1111 1111	uuuu uuuu
TRISE	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1111 -111	1111 -111	uuuu -uuu
TRISD	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1111 1111	1111 1111	uuuu uuuu
TRISC	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1111 1111	1111 1111	uuuu uuuu
TRISB	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1111 1111	1111 1111	uuuu uuuu
TRISA(5)	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	111- 1111(5)	111- 1111(5)	uuu- uuuu(5)
ODCON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
SLRCON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	-000 0000	-111 1111	-111 1111
LATG	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	---x xxxx	---x xxxx	---u uuuu
LATF	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx -xxx	uuuu -uuu
LATE	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx -xxx	xxxx xxxx	uuuu uuuu
LATD	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
LATC	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
LATB	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
LATA(5)	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- xxxx(5)	xxx- xxxx(5)	uuu- uuuu(5)
T4CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	-000 0000	-000 0000	-uuu uuuu
TMR4	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
PORTG	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	---x xxxx	---x xxxx	---u uuuu
PORTF	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
PORTE	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
PORTD	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
PORTC	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
PORTB	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
PORTA(5)	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- xxxx(5)	xxx- xxxx(5)	uuu- uuuu(5)
EECON1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xx-0 x000	uu-0 u000	uu-u uuuu
EECON2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
SPBRGH1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
SPBRGH2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
SPBRG2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
RCREG2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
TXREG2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
IPR5	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1111 1111	1111 1111	uuuu uuuu
PIR5	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu

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

Note 1: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

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

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

4: See Table 5-3 for Reset value for specific conditions.

5: Bits 6 and 7 of PORTA, LATA and TRISA are enabled, depending on the oscillator mode selected. When not enabled as PORTA pins, they are disabled and read as '0'.

TABLE 6-2: PIC18F66K80 FAMILY REGISTER FILE SUMMARY (CONTINUED)

Addr.	File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR on page
F6Dh	RXB0D7	RXB0D77	RXB0D76	RXB0D75	RXB0D74	RXB0D73	RXB0D72	RXB0D71	RXB0D70	91
F6Ch	RXB0D6	RXB0D67	RXB0D66	RXB0D65	RXB0D64	RXB0D63	RXB0D62	RXB0D61	RXB0D60	91
F6Bh	RXB0D5	RXB0D57	RXB0D56	RXB0D55	RXB0D54	RXB0D53	RXB0D52	RXB0D51	RXB0D50	91
F6Ah	RXB0D4	RXB0D47	RXB0D46	RXB0D45	RXB0D44	RXB0D43	RXB0D42	RXB0D41	RXB0D40	91
F69h	RXB0D3	RXB0D37	RXB0D36	RXB0D35	RXB0D34	RXB0D33	RXB0D32	RXB0D31	RXB0D30	91
F68h	RXB0D2	RXB0D27	RXB0D26	RXB0D25	RXB0D24	RXB0D23	RXB0D22	RXB0D21	RXB0D20	91
F67h	RXB0D1	RXB0D17	RXB0D16	RXB0D15	RXB0D14	RXB0D13	RXB0D12	RXB0D11	RXB0D10	91
F66h	RXB0D0	RXB0D07	RXB0D06	RXB0D05	RXB0D04	RXB0D03	RXB0D02	RXB0D01	RXB0D00	91
F65h	RXB0DLC	—	RXRTR	RB1	RB0	DLC3	DLC2	DLC1	DLC0	91
F64h	RXB0EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	91
F63h	RXB0EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	91
F62h	RXB0SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	91
F61h	RXB0SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	91
F60h	RXB0CON	RXFUL	RXM1	RXM0	—	RXRTRRO	RXB0DBEN	JTOFF	FILHIT0	91
F60h	RXB0CON	RXFUL	RXM1	RTRRO	FILHIT4	FILHIT3	FILHIT2	FILHIT1	FILHIT0	91
F5Fh	CM1CON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0	91
F5Eh	CM2CON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0	91
F5Dh	ANCON0	ANSEL7	ANSEL6	ANSEL5	ANSEL4	ANSEL3	ANSEL2	ANSEL1	ANSEL0	91
F5Ch	ANCON1	—	ANSEL14	ANSEL13	ANSEL12	ANSEL11	ANSEL10	ANSEL9	ANSEL8	91
F5Bh	WPUB	WPUB7	WPUB6	WPUB5	WPUB4	WPUB3	WPUB2	WPUB1	WPUB0	91
F5Ah	IOCB	IOCB7	IOCB6	IOCB5	IOCB4	—	—	—	—	91
F59h	PMD0	CCP5MD	CCP4MD	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSPMD	91
F58h	PMD1	PSPMD	CTMUMD	ADCMD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	TMR0MD	93
F57h	PMD2	—	—	—	—	MODMD	ECANMD	CMP2MD	CMP1MD	93
F56h	PADCFG1	RDPU	REPU	RFPU	RGPU	—	—	—	CTMUDS	93
F55h	CTMUCONH	CTMUEN	—	CTMUSIDL	TGEN	EDGEN	EDGSEQEN	IDISSEN	CTTRIG	93
F54h	CTMUCONL	EDG2POL	EDG2SEL1	EDG2SEL0	EDG1POL	EDG1SEL1	EDG1SEL0	EDG2STAT	EDG1STAT	93
F53h	CTMUICON	ITRIM5	ITRIM4	ITRIM3	ITRIM2	ITRIM1	ITRIM0	IRNG1	IRNG0	93
F52h	CCPR2H	Capture/Compare/PWM Register 2 High Byte								93
F51h	CCPR2L	Capture/Compare/PWM Register 2 Low Byte								93
F50h	CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	93
F4Fh	CCPR3H	Capture/Compare/PWM Register 3 High Byte								93
F4Eh	CCPR3L	Capture/Compare/PWM Register 3 Low Byte								93
F4Dh	CCP3CON	—	—	DC3B1	D32B0	CCP3M3	CCP3M2	CCP3M1	CCP3M0	93
F4Ch	CCPR4H	Capture/Compare/PWM Register 4 High Byte								93
F4Bh	CCPR4L	Capture/Compare/PWM Register 4 Low Byte								93
F4Ah	CCP4CON	—	—	DC4B1	DC4B0	CCP4M3	CCP4M2	CCP4M1	CCP4M0	93
F49H	CCPR5H	Capture/Compare/PWM Register 5 High Byte								93
F48h	CCPR5L	Capture/Compare/PWM Register 5 Low Byte								93
F47h	CCP5CON	—	—	DC5B1	DC5B0	CCP5M3	CCP5M2	CCP5M1	CCP5M0	93
F46h	PSPCON	IBF	OBF	IBOV	PSPMODE	—	—	—	—	93
F45h	MDCON	MDEN	MDOE	MDSLR	MDOPOL	MDO	—	—	MDBIT	93
F44h	MDSRC	MDSODIS	—	—	—	MDSRC3	MDSRC2	MDSRC1	MDSRC0	93
F43h	MDCARH	MDCHODIS	MDCHPOL	MDCHSYNC	—	MDCH3	MDCH2	MDCH1	MDCH0	93
F42h	MDCARL	MDCLODIS	MDCLPOL	MDCLSYNC	—	MDCL3	MDCL2	MDCL1	MDCL0	93
F41h	Unimplemented									—
F40h	Unimplemented									—
F3Fh	CANCON_RO0	CANCON_RO0								93
F3Eh	CANSTAT_RO0	CANSTAT_RO0								93
F3Dh	RXB1D7	RXB1D77	RXB1D76	RXB1D75	RXB1D74	RXB1D73	RXB1D72	RXB1D71	RXB1D70	93
F3Ch	RXB1D6	RXB1D67	RXB1D66	RXB1D65	RXB1D64	RXB1D63	RXB1D62	RXB1D61	RXB1D60	93

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.

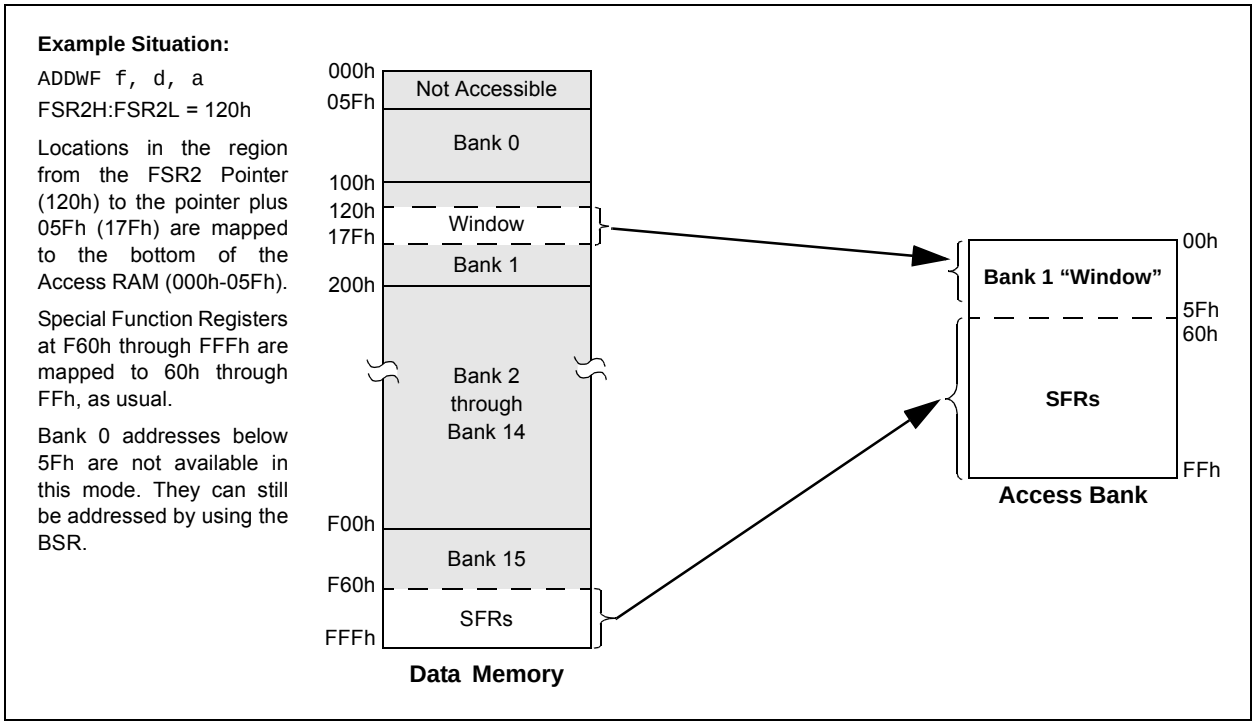
The value of FSR2 establishes the lower boundary of the addresses mapped into the window, while the upper boundary is defined by FSR2 plus 95 (5Fh). Addresses in the Access RAM above 5Fh are mapped as previously described. (See **Section 6.3.2 "Access Bank"**.) An example of Access Bank remapping in this addressing mode is shown in Figure 6-10.

Remapping the Access Bank applies *only* to operations using the Indexed Literal Offset mode. Operations that use the BSR (Access RAM bit = 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



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

Example 9-3 shows the sequence to do a 16 x 16 unsigned multiplication. Equation 9-1 shows the algorithm that is used. The 32-bit result is stored in four registers (RES3:RES0).

EQUATION 9-1: 16 x 16 UNSIGNED MULTIPLICATION ALGORITHM

$$
\begin{aligned}
RES3:RES0 &= ARG1H:ARG1L \bullet ARG2H:ARG2L \\
&= (ARG1H \bullet ARG2H \bullet 2^{16}) + \\
&\quad (ARG1H \bullet ARG2L \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2H \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2L)
\end{aligned}
$$

EXAMPLE 9-3: 16 x 16 UNSIGNED MULTIPLY ROUTINE

```
    MOVF    ARG1L, W
    MULWF   ARG2L           ; ARG1L * ARG2L->
                            ; PRODH:PRODL
    MOVFF   PRODH, RES1     ;
    MOVFF   PRODL, RES0     ;
;
    MOVF    ARG1H, W
    MULWF   ARG2H           ; ARG1H * ARG2H->
                            ; PRODH:PRODL
    MOVFF   PRODH, RES3     ;
    MOVFF   PRODL, RES2     ;
;
    MOVF    ARG1L, W
    MULWF   ARG2H           ; ARG1L * ARG2H->
                            ; PRODH:PRODL
    MOVF    PRODL, W        ;
    ADDWF   RES1, F         ; Add cross
    MOVF    PRODH, W        ; products
    ADDWFC  RES2, F         ;
    CLRF    WREG            ;
    ADDWFC  RES3, F         ;
;
    MOVF    ARG1H, W        ;
    MULWF   ARG2L           ; ARG1H * ARG2L->
                            ; PRODH:PRODL
    MOVF    PRODL, W        ;
    ADDWF   RES1, F         ; Add cross
    MOVF    PRODH, W        ; products
    ADDWFC  RES2, F         ;
    CLRF    WREG            ;
    ADDWFC  RES3, F         ;
```

Example 9-4 shows the sequence to do a 16 x 16 signed multiply. Equation 9-2 shows the algorithm used. The 32-bit result is stored in four registers (RES3:RES0). To account for the sign bits of the arguments, the MSb for each argument pair is tested and the appropriate subtractions are done.

EQUATION 9-2: 16 x 16 SIGNED MULTIPLICATION ALGORITHM

$$
\begin{aligned}
RES3:RES0 &= ARG1H:ARG1L \bullet ARG2H:ARG2L \\
&= (ARG1H \bullet ARG2H \bullet 2^{16}) + \\
&\quad (ARG1H \bullet ARG2L \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2H \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2L) + \\
&\quad (-1 \bullet ARG2H\langle 7\rangle \bullet ARG1H:ARG1L \bullet 2^{16}) + \\
&\quad (-1 \bullet ARG1H\langle 7\rangle \bullet ARG2H:ARG2L \bullet 2^{16})
\end{aligned}
$$

EXAMPLE 9-4: 16 x 16 SIGNED MULTIPLY ROUTINE

```
   MOVF    ARG1L, W
   MULWF   ARG2L        ; ARG1L * ARG2L ->
                        ; PRODH:PRODL
   MOVFF   PRODH, RES1 ;
   MOVFF   PRODL, RES0 ;
;
   MOVF    ARG1H, W
   MULWF   ARG2H        ; ARG1H * ARG2H ->
                        ; PRODH:PRODL
   MOVFF   PRODH, RES3 ;
   MOVFF   PRODL, RES2 ;
;
   MOVF    ARG1L, W
   MULWF   ARG2H        ; ARG1L * ARG2H ->
                        ; PRODH:PRODL
   MOVF    PRODL, W     ;
   ADDWF   RES1, F      ; Add cross
   MOVF    PRODH, W     ; products
   ADDWFC  RES2, F      ;
   CLRF    WREG         ;
   ADDWFC  RES3, F      ;
;
   MOVF    ARG1H, W     ;
   MULWF   ARG2L        ; ARG1H * ARG2L ->
                        ; PRODH:PRODL
   MOVF    PRODL, W     ;
   ADDWF   RES1, F      ; Add cross
   MOVF    PRODH, W     ; products
   ADDWFC  RES2, F      ;
   CLRF    WREG         ;
   ADDWFC  RES3, F      ;
;
   BTFSS   ARG2H, 7     ; ARG2H:ARG2L neg?
   BRA  SIGN_ARG1       ; no, check ARG1
   MOVF    ARG1L, W     ;
   SUBWF   RES2         ;
   MOVF    ARG1H, W     ;
   SUBWFB  RES3         ;
SIGN_ARG1
   BTFSS   ARG1H, 7     ; ARG1H:ARG1L neg?
   BRA     CONT_CODE    ; no, done
   MOVF    ARG2L, W     ;
   SUBWF   RES2         ;
   MOVF    ARG2H, W     ;
   SUBWFB  RES3
;
CONT_CODE
   :
```

11.4 PORTC, TRISC and LATC Registers

PORTC is an eight-bit wide, bidirectional port. The corresponding Data Direction and Output Latch registers are TRISC and LATC. Only PORTC pins, RC2 through RC7, are digital only pins.

PORTC is multiplexed with CCP, MSSP and EUSARTx peripheral functions (Table 11-5). The pins have Schmitt Trigger input buffers. The pins for CCP, SPI and EUSARTx are also configurable for open-drain output whenever these functions are active. Open-drain configuration is selected by setting the SSPOD, CCPxOD and U1OD control bits in the ODCON register.

RC1 is configurable for open-drain output when CCP2 is active on this pin. Open-drain configuration is selected by setting the CCP2OD control bit (ODCON<3>).

When enabling peripheral functions, use care in defining TRIS bits for each PORTC pin. Some peripherals can override the TRIS bit to make a pin an output or input. Consult the corresponding peripheral section for the correct TRIS bit settings.

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

The contents of the TRISC register are affected by peripheral overrides. Reading TRISC always returns the current contents, even though a peripheral device may be overriding one or more of the pins.

EXAMPLE 11-3: INITIALIZING PORTC

```
CLRF    PORTC   ; Initialize PORTC by
                ; clearing output
                ; data latches
CLRF    LATC    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0CFh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISC   ; Set RC<3:0> as inputs
                ; RC<5:4> as outputs
                ; RC<7:6> as inputs
```

13.0 TIMER0 MODULE

The Timer0 module incorporates the following features:

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

The T0CON register (Register 13-1) controls all aspects of the module's operation, including the prescale selection. It is both readable and writable.

Figure 13-1 provides a simplified block diagram of the Timer0 module in 8-bit mode. Figure 13-2 provides a simplified block diagram of the Timer0 module in 16-bit mode.

REGISTER 13-1: T0CON: TIMER0 CONTROL REGISTER

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0
bit 7							bit 0

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

bit 7 **TMR0ON:** Timer0 On/Off Control bit
1 = Enables Timer0
0 = Stops Timer0

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

bit 5 **T0CS**: Timer0 Clock Source Select bit
1 = Transitions on T0CKI pin
0 = Internal instruction cycle clock (CLKO)

bit 4 **T0SE**: Timer0 Source Edge Select bit
1 = Increments on high-to-low transition on T0CKI pin
0 = Increments on low-to-high transition on T0CKI pin

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

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

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.

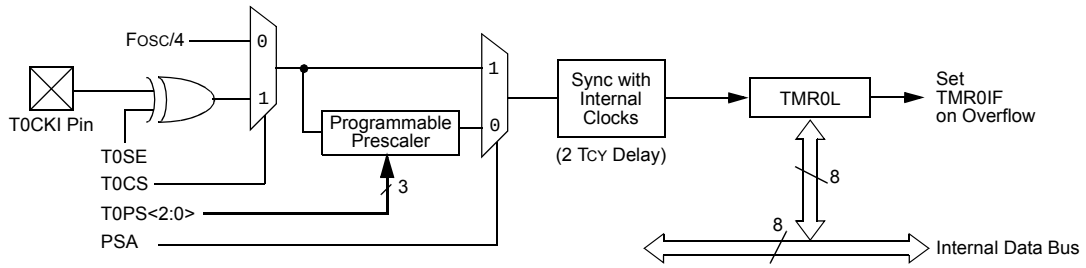
An external clock source can be used to drive Timer0; however, it must meet certain requirements to ensure that the external clock can be synchronized with the internal phase clock (TOSC). There is a delay between synchronization and the onset of incrementing the timer/counter.

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.

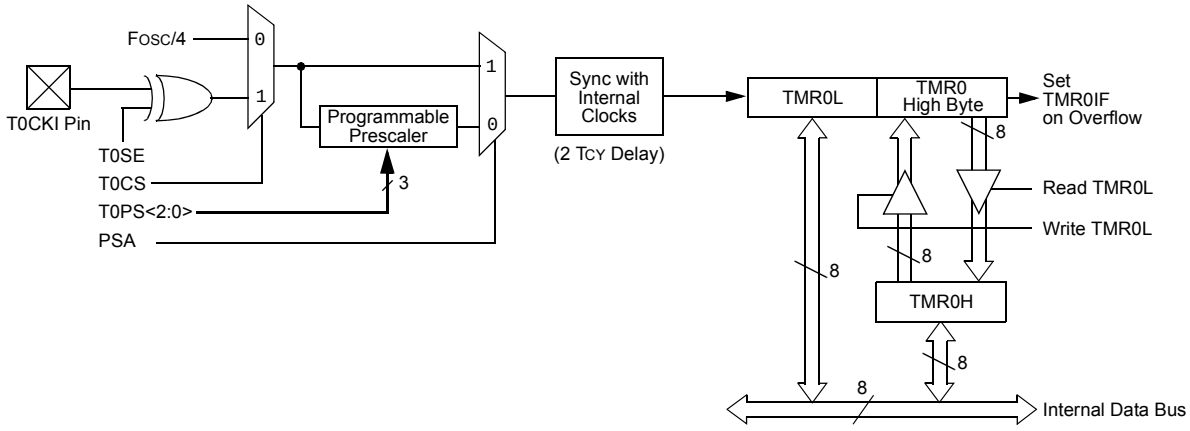
Similarly, a write to the high byte of Timer0 must also take place through the TMR0H Buffer register. The high byte is updated with the contents of TMR0H when a write occurs to TMR0L. This allows all 16 bits of Timer0 to be updated at once.

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.

16.5.2 TIMER3 GATE SOURCE SELECTION

The Timer3 gate source can be selected from one of four different sources. Source selection is controlled by the T3GSS<1:0> bits (T3GCON<1:0>). The polarity for each available source is also selectable and is controlled by the T3GPOL bit (T3GCON<6>).

TABLE 16-2: TIMER3 GATE SOURCES

T3GSS<1:0>	Timer3 Gate Source
00	Timerx Gate Pin
01	TMR4 to Match PR4 (TMR4 increments to match PR4)
10	Comparator 1 Output (comparator logic high output)
11	Comparator 2 Output (comparator logic high output)

16.5.2.1 T3G Pin Gate Operation

The T3G pin is one source for Timer3 gate control. It can be used to supply an external source to the Timerx gate circuitry.

16.5.2.2 Timer4 Match Gate Operation

The TMR4 register will increment until it matches the value in the PR4 register. On the very next increment cycle, TMR4 will be reset to 00h. When this Reset occurs, a low-to-high pulse will automatically be generated and internally supplied to the Timerx gate circuitry. The pulse will remain high for one instruction cycle and will return back to a low state until the next match.

Depending on T3GPOL, Timerx increments differently when TMR4 matches PR4. When T3GPOL = 1, Timer3 increments for a single instruction cycle following a TMR4 match with PR4. When T3GPOL = 0, Timer3 increments continuously, except for the cycle following the match, when the gate signal goes from low-to-high.

16.5.2.3 Comparator 1 Output Gate Operation

The output of Comparator 1 can be internally supplied to the Timer3 gate circuitry. After setting up Comparator 1 with the CM1CON register, Timer3 will increment depending on the transitions of the CMP1OUT (CMSTAT<6>) bit.

16.5.2.4 Comparator 2 Output Gate Operation

The output of Comparator 2 can be internally supplied to the Timer3 gate circuitry. After setting up Comparator 2 with the CM2CON register, Timer3 will increment depending on the transitions of the CMP2OUT (CMSTAT<7>) bit.

16.5.3 TIMER3 GATE TOGGLE MODE

When Timer3 Gate Toggle mode is enabled, it is possible to measure the full cycle length of a Timer3 gate signal, as opposed to the duration of a single level pulse.

The Timer3 gate source is routed through a flip-flop that changes state on every incrementing edge of the signal. (For timing details, see Figure 16-3.)

The T3GVAL bit will indicate when the Toggled mode is active and the timer is counting.

Timer3 Gate Toggle mode is enabled by setting the T3GTM bit (T3GCON<5>). When the T3GTM bit is cleared, the flip-flop is cleared and held clear. This is necessary in order to control which edge is measured.

FIGURE 16-3: TIMER3 GATE TOGGLE MODE

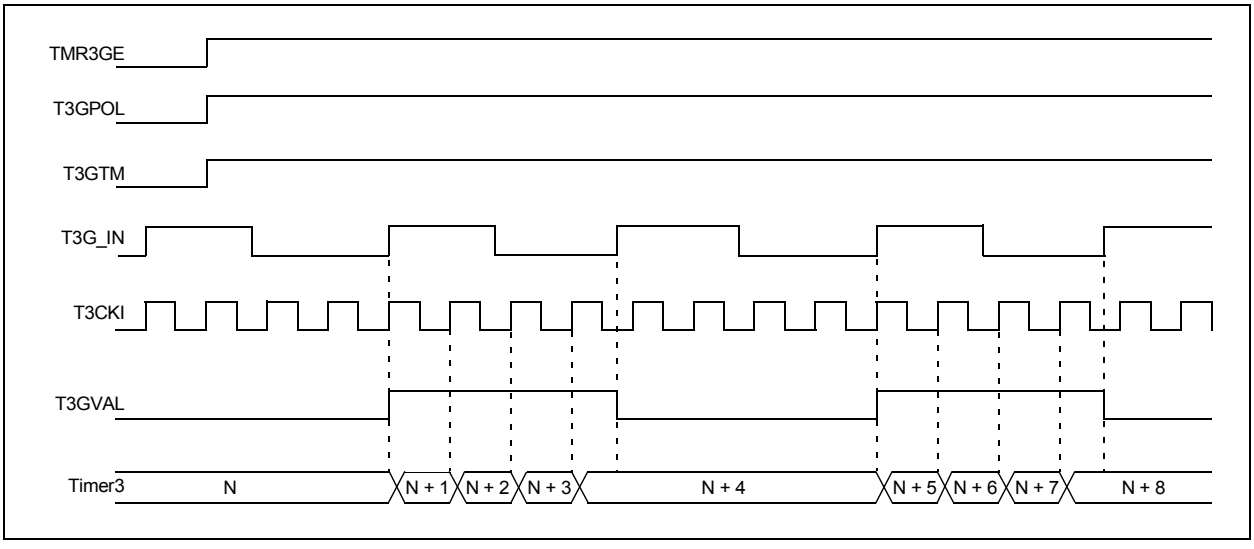
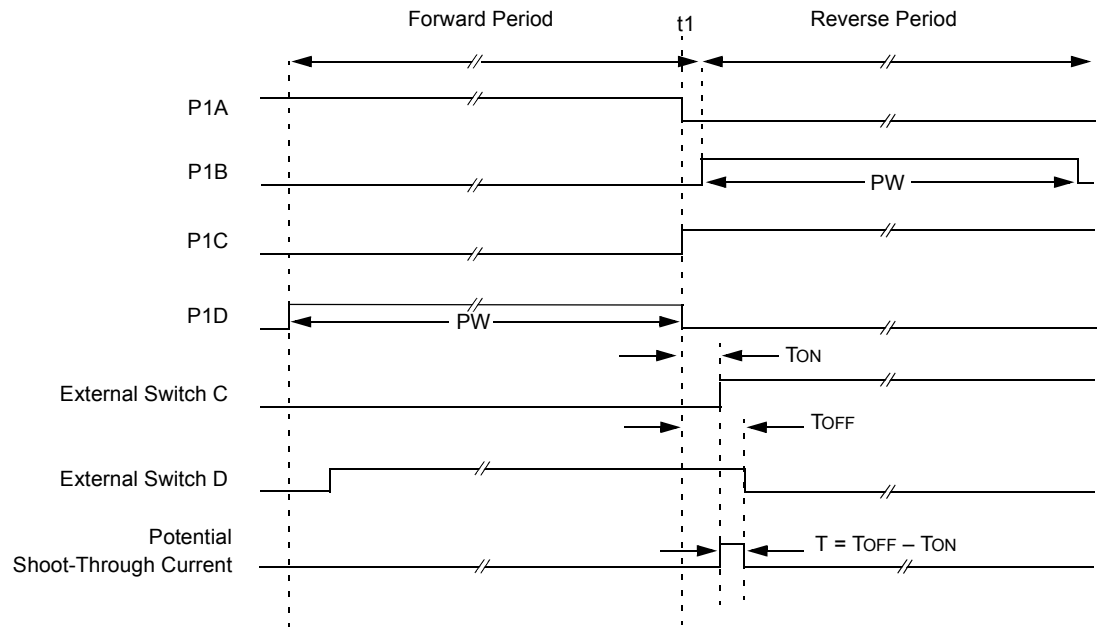


FIGURE 20-11: EXAMPLE OF PWM DIRECTION CHANGE AT NEAR 100% DUTY CYCLE



Note 1: All signals are shown as active-high.

2: TON is the turn-on delay of power switch, QC, and its driver.

3: TOFF is the turn-off delay of power switch, QD, and its driver.

20.4.3 START-UP CONSIDERATIONS

When any PWM mode is used, the application hardware must use the proper external pull-up and/or pull-down resistors on the PWM output pins.

Note:	When the microcontroller is released from Reset, all of the I/O pins are in the high-impedance state. The external circuits must keep the power switch devices in the OFF state until the microcontroller drives the I/O pins with the proper signal levels or activates the PWM output(s).

The CCP1M<1:0> bits of the CCP1CON register allow the user to choose whether the PWM output signals are active-high or active-low for each pair of PWM output pins (P1A/P1C and P1B/P1D). The PWM output polarities must be selected before the PWM pin output drivers are enabled. Changing the polarity configuration while the PWM pin output drivers are enabled is not recommended since it may result in damage to the application circuits.

The P1A, P1B, P1C and P1D output latches may not be in the proper states when the PWM module is initialized. Enabling the PWM pin output drivers at the same time as the Enhanced PWM modes may cause damage to the application circuit. The Enhanced PWM modes must be enabled in the proper Output mode and complete a full PWM cycle before enabling the PWM pin output drivers. The completion of a full PWM cycle is indicated by the TMR2IF or TMR4IF bit of the PIR1 or PIR4 register being set as the second PWM period begins.

20.4.4 ENHANCED PWM AUTO-SHUTDOWN MODE

The PWM mode supports an Auto-Shutdown mode that will disable the PWM outputs when an external shutdown event occurs. Auto-Shutdown mode places the PWM output pins into a predetermined state. This mode is used to help prevent the PWM from damaging the application.

The auto-shutdown sources are selected using the ECCP1AS<2:0> bits (ECCP1AS<6:4>). A shutdown event may be generated by:

- A logic '0' on the pin that is assigned the FLT0 input function
- Comparator C1
- Comparator C2
- Setting the ECCP1ASE bit in firmware

21.4.8 I²C™ MASTER MODE START CONDITION TIMING

To initiate a Start condition, the user sets the Start Enable bit, SEN (SSPCON2<0>). If the SDA and SCL pins are sampled high, the Baud Rate Generator is reloaded with the contents of SSPADD<6:0> and starts its count. If SCL and SDA are both sampled high when the Baud Rate Generator times out (TBRG), the SDA pin is driven low. The action of the SDA being driven low while SCL is high is the Start condition and causes the S bit (SSPSTAT<3>) to be set. Following this, the Baud Rate Generator is reloaded with the contents of SSPADD<6:0> and resumes its count. When the Baud Rate Generator times out (TBRG), the SEN bit (SSPCON2<0>) will be automatically cleared by hardware. The Baud Rate Generator is suspended, leaving the SDA line held low and the Start condition is complete.

Note:	If, at the beginning of the Start condition, the SDA and SCL pins are already sampled low or if during the Start condition, the SCL line is sampled low before the SDA line is driven low, a bus collision occurs, the Bus Collision Interrupt Flag, BCLIF, is set, the Start condition is aborted and the I²C module is reset into its Idle state.

21.4.8.1 WCOL Status Flag

If the user writes the SSPBUF when a Start sequence is in progress, the WCOL bit is set and the contents of the buffer are unchanged (the write doesn't occur).

Note:	Because queueing of events is not allowed, writing to the lower 5 bits of SSPCON2 is disabled until the Start condition is complete.

FIGURE 21-21: FIRST START BIT TIMING

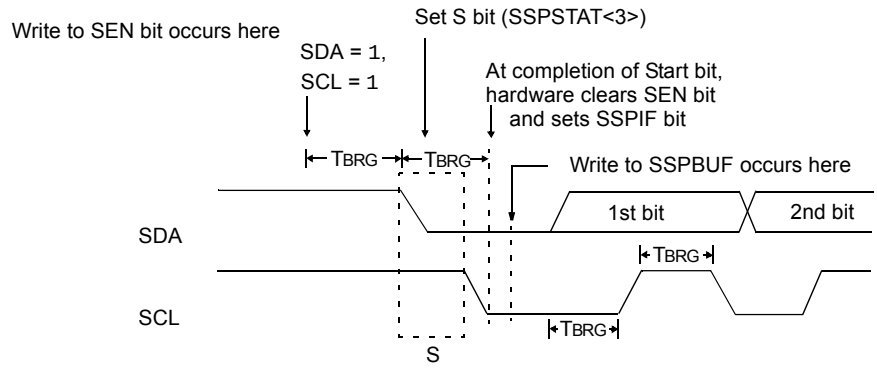
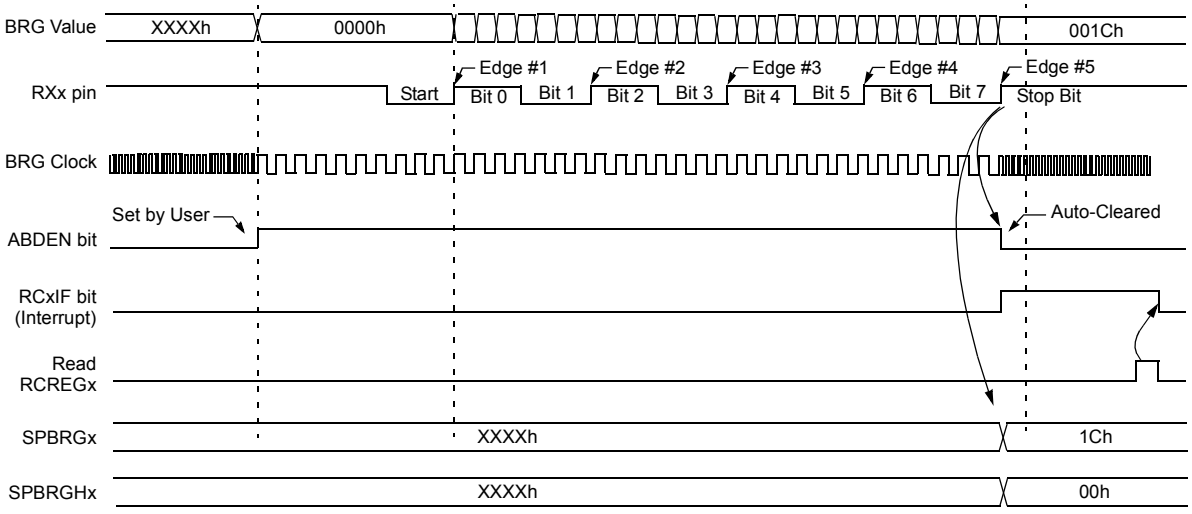
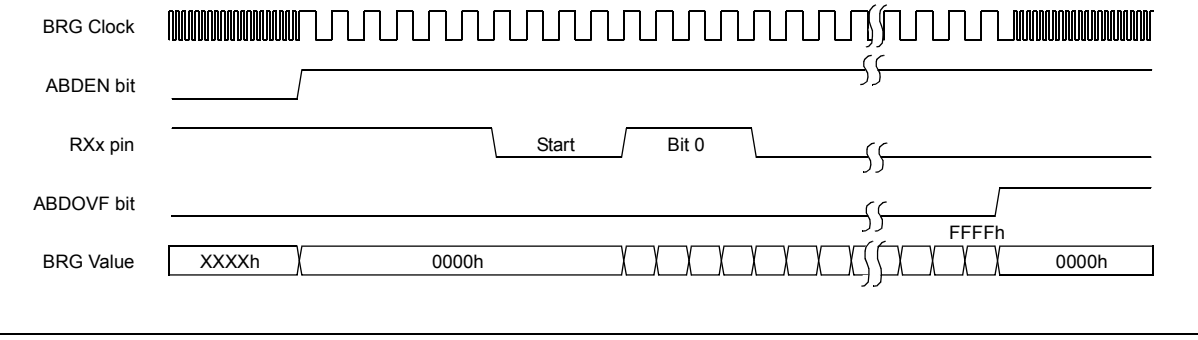


FIGURE 22-1: AUTOMATIC BAUD RATE CALCULATION



Note: The ABD sequence requires the EUSARTx module to be configured in Asynchronous mode and WUE = 0.

FIGURE 22-2: BRG OVERFLOW SEQUENCE



27.7 Message Reception

27.7.1 RECEIVING A MESSAGE

Of all receive buffers, the MAB is always committed to receiving the next message from the bus. The MCU can access one buffer while the other buffer is available for message reception or holding a previously received message.

Note:	The entire contents of the MAB are moved into the receive buffer once a message is accepted. This means that regardless of the type of identifier (standard or extended) and the number of data bytes received, the entire receive buffer is overwritten with the MAB contents. Therefore, the contents of all registers in the buffer must be assumed to have been modified when any message is received.

When a message is moved into either of the receive buffers, the associated RXFUL bit is set. This bit must be cleared by the MCU when it has completed processing the message in the buffer in order to allow a new message to be received into the buffer. This bit provides a positive lockout to ensure that the firmware has finished with the message before the module attempts to load a new message into the receive buffer. If the receive interrupt is enabled, an interrupt will be generated to indicate that a valid message has been received.

Once a message is loaded into any matching buffer, user firmware may determine exactly what filter caused this reception by checking the filter hit bits in the RXBnCON or BnCON registers. In Mode 0, FILHIT<2:0> of RXBnCON serve as filter hit bits. In Mode 1 and 2, FILHIT<4:0> bits of BnCON serve as filter hit bits. The same registers also indicate whether the current message is an RTR frame or not. A received message is considered a standard identifier message if the EXID/EXIDE bit in the RXBnSIDL or the BnSIDL register is cleared. Conversely, a set EXID bit indicates an extended identifier message. If the received message is a standard identifier message, user firmware needs to read the SIDL and SIDH registers. In the case of an extended identifier message, firmware should read the SIDL, SIDH, EIDL and EIDH registers. If the RXBnDLC or BnDLC register contain non-zero data count, user firmware should also read the corresponding number of data bytes by accessing the RXBnDm or the BnDm registers. When a received message is an RTR, and if the current buffer is not configured for automatic RTR handling, user firmware must take appropriate action and respond manually.

Each receive buffer contains RXM bits to set special Receive modes. In Mode 0, RXM<1:0> bits in RXBnCON define a total of four Receive modes. In Mode 1 and 2, RXM1 bit, in combination with the EXID mask and filter bit, define the same four receive modes. Normally, these bits are set to '00' to enable reception of all valid messages as determined by the appropriate acceptance filters. In this case, the determination of whether or not to receive standard or extended messages is determined by the EXIDE bit in the acceptance filter register. In Mode 0, if the RXM bits are set to '01' or '10', the receiver will accept only messages with standard or extended identifiers, respectively. If an acceptance filter has the EXIDE bit set, such that it does not correspond with the RXM mode, that acceptance filter is rendered useless. In Mode 1 and 2, setting EXID in the SIDL Mask register will ensure that only standard or extended identifiers are received. These two modes of RXM bits can be used in systems where it is known that only standard or extended messages will be on the bus. If the RXM bits are set to '11' (RXM1 = 1 in Mode 1 and 2), the buffer will receive all messages regardless of the values of the acceptance filters. Also, if a message has an error before the end of frame, that portion of the message assembled in the MAB before the error frame will be loaded into the buffer. This mode may serve as a valuable debugging tool for a given CAN network. It should not be used in an actual system environment as the actual system will always have some bus errors and all nodes on the bus are expected to ignore them.

In Mode 1 and 2, when a programmable buffer is configured as a transmit buffer and one or more acceptance filters are associated with it, all incoming messages matching this acceptance filter criteria will be discarded. To avoid this scenario, user firmware must make sure that there are no acceptance filters associated with a buffer configured as a transmit buffer.

27.7.2 RECEIVE PRIORITY

When in Mode 0, RXB0 is the higher priority buffer and has two message acceptance filters associated with it. RXB1 is the lower priority buffer and has four acceptance filters associated with it. The lower number of acceptance filters makes the match on RXB0 more restrictive and implies a higher priority for that buffer. Additionally, the RXB0CON register can be configured such that if RXB0 contains a valid message and another valid message is received, an overflow error will not occur and the new message will be moved into RXB1 regardless of the acceptance criteria of RXB1. There are also two programmable acceptance filter masks available, one for each receive buffer (see **Section 27.5 "CAN Message Buffers"**).

In Mode 1 and 2, there are a total of 16 acceptance filters available and each can be dynamically assigned to any of the receive buffers. A buffer with a lower number has higher priority. Given this, if an incoming message matches with two or more receive buffer acceptance criteria, the buffer with the lower number will be loaded with that message.

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	75	160	µA	-40°C	VDD = 1.8V(4) Regulator Disabled	FOSC = 4 MHz (**PRI_IDLE** mode, EC oscillator)
		75	160	µA	+25°C		
		75	160	µA	+60°C		
		76	170	µA	+85°C		
		82	180	µA	+125°C		
	PIC18**LF**XXK80	148	300	µA	-40°C	VDD = 3.3V(4) Regulator Disabled	
		148	300	µA	+25°C		
		148	300	µA	+60°C		
		150	400	µA	+85°C		
		157	460	µA	+125°C		
	PIC18**F**XXK80	187	320	µA	-40°C	VDD = 3.3V(5) Regulator Enabled	
		204	320	µA	+25°C		
		212	320	µA	+60°C		
		218	420	µA	+85°C		
		230	480	µA	+125°C		
	PIC18**F**XXK80	230	500	µA	-40°C	VDD = 5V(5) Regulator Enabled	
		230	500	µA	+25°C		
		230	500	µA	+60°C		
		240	600	µA	+85°C		
		250	700	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.

TABLE 31-22: MSSP I²C™ BUS DATA REQUIREMENTS

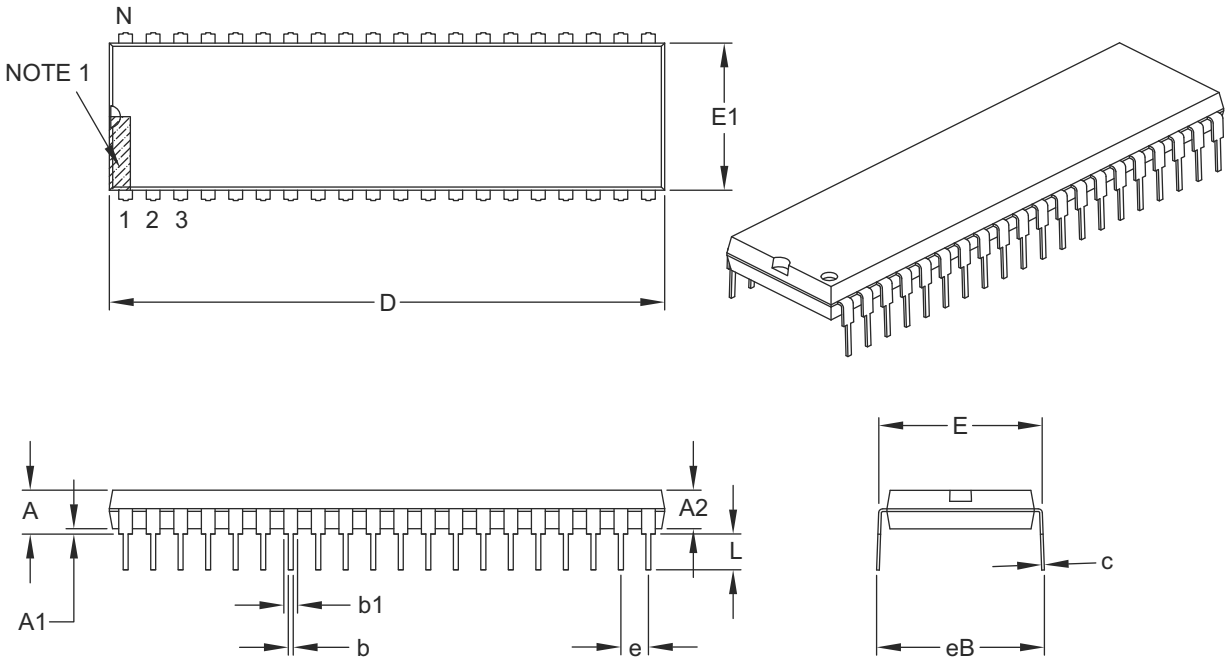
Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
100	THIGH	Clock High Time	100 kHz mode	2(TOSC)(BRG + 1)	—	—	
			400 kHz mode	2(TOSC)(BRG + 1)	—	—	
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—	—	
101	TLOW	Clock Low Time	100 kHz mode	2(TOSC)(BRG + 1)	—	—	
			400 kHz mode	2(TOSC)(BRG + 1)	—	—	
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—	—	
102	TR	SDA and SCL Rise Time	100 kHz mode	—	1000	ns	CB is specified to be from 10 to 400 pF
			400 kHz mode	20 + 0.1 CB	300	ns	
			1 MHz mode(1)	—	300	ns	
103	TF	SDA and SCL Fall Time	100 kHz mode	—	300	ns	CB is specified to be from 10 to 400 pF
			400 kHz mode	20 + 0.1 CB	300	ns	
			1 MHz mode(1)	—	100	ns	
90	TSU:STA	Start Condition Setup Time	100 kHz mode	2(TOSC)(BRG + 1)	—	—	Only relevant for Repeated Start condition
			400 kHz mode	2(TOSC)(BRG + 1)	—	—	
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—	—	
91	THD:STA	Start Condition Hold Time	100 kHz mode	2(TOSC)(BRG + 1)	—	—	After this period, the first clock pulse is generated
			400 kHz mode	2(TOSC)(BRG + 1)	—	—	
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—	—	
106	THD:DAT	Data Input Hold Time	100 kHz mode	0	—	—	
			400 kHz mode	0	0.9	μs	
			1 MHz mode(1)	—	μs	ns	
107	TSU:DAT	Data Input Setup Time	100 kHz mode	250	—	ns	**(Note 2)**
			400 kHz mode	100	—	ns	
			1 MHz mode(1)	—	—	ns	
92	TSU:STO	Stop Condition Setup Time	100 kHz mode	2(TOSC)(BRG + 1)	—	—	
			400 kHz mode	2(TOSC)(BRG + 1)	—	—	
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—	—	
109	TAA	Output Valid from Clock	100 kHz mode	—	3500	ns	
			400 kHz mode	—	1000	ns	
			1 MHz mode(1)	—	—	ns	
110	TBUF	Bus Free Time	100 kHz mode	4.7	—	μs	Time the bus must be free before a new transmission can start
			400 kHz mode	1.3	—	μs	
			1 MHz mode(1)	—	—	μs	
D102	CB	Bus Capacitive Loading		—	400	pF	

Note 1: Maximum pin capacitance = 10 pF for all I²C™ pins.

2: A Fast mode I²C bus device can be used in a Standard mode I²C bus system, but Parameter #107 ≥ 250 ns must then be met. This will automatically be the case if the device does not stretch the LOW period of the SCL signal. If such a device does stretch the LOW period of the SCL signal, it must output the next data bit to the SDA line, Parameter #102 + Parameter #107 = 1000 + 250 = 1250 ns (for 100 kHz mode), before the SCL line is released.

40-Lead Plastic Dual In-Line (P) – 600 mil Body [PDIP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Units		INCHES		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		40	
Pitch	e		.100 BSC	
Top to Seating Plane	A	–	–	.250
Molded Package Thickness	A2	.125	–	.195
Base to Seating Plane	A1	.015	–	–
Shoulder to Shoulder Width	E	.590	–	.625
Molded Package Width	E1	.485	–	.580
Overall Length	D	1.980	–	2.095
Tip to Seating Plane	L	.115	–	.200
Lead Thickness	c	.008	–	.015
Upper Lead Width	b1	.030	–	.070
Lower Lead Width	b	.014	–	.023
Overall Row Spacing §	eB	–	–	.700

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-016B

TABLE B-2: NOTABLE DIFFERENCES BETWEEN 64-PIN DEVICES – PIC18F66K80 AND PIC18F8680 FAMILIES

Characteristic	PIC18F66K80 Family	PIC18F8680 Family
Max Operating Frequency	64 MHz	40 MHz
Max Program Memory	64K	64K
Data Memory (bytes)	3,648	3,328
CTMU	Yes	No
SOSC Oscillator Options	Low-power oscillator option for SOSC	No options
T1CKI Clock	T1CKI can be used as a clock without enabling the SOSC oscillator	No
INTOSC	Up to 16 MHz	No Internal Oscillator
SPI/I^2C™	1 Module	1 Module
Timers	Two 8-bit, Three 16-bit	Two 8-bit, Three 16-bit
ECCP	1	1
CCP	4	1
Data EEPROM (bytes)	1,024	1,024
WDT Prescale Options	22	16
5V Operation	18FXXK80 parts – 5V operation 18LFXXK80 parts – 3.3V operation	Yes
nanoWatt XLP	Yes	No
On-Chip 3.3V Regulator	18FXXK80 parts – Yes 18LFXXK80 parts – No	No
Low-Power BOR	Yes	No
A/D Converter	12-bit signed differential	10-bit
A/D Channels	15 Channels	12 Channels
Internal Temp Sensor	Yes	No
EUSART	Two	One
Comparators	Two	Two
Oscillator Options	14	Seven
Ultra Low-Power Wake-up (ULPW)	Yes	No
Adjustable Slew Rate for I/O	Yes	No
PLL	Available for all oscillator options	Available for only high-speed crystal and external oscillator
Data Signal Modulator	Yes	No