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	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 ~ 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/pic18lf26k80t-i-ss

TABLE 1-5: PIC18F4XK80 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	PDIP	QFN/ TQFP			
					PORTB is a bidirectional I/O port.
RB0/AN10/FLT0/INT0	33	8			
RB0			I/O	ST/ CMOS	Digital I/O.
AN10			I	Analog	Analog Input 10.
FLT0			I	ST	Enhanced PWM Fault input for ECCP1.
INT0			I	ST	External Interrupt 0.
RB1/AN8/CTDIN/INT1	34	9			
RB1			I/O	ST/ CMOS	Digital I/O.
AN8			I	Analog	Analog Input 8.
CTDIN			I	ST	CTMU pulse delay input.
INT1			I	ST	External Interrupt 1.
RB2/CANTX/CTED1/ INT2	35	10			
RB2			I/O	ST/ CMOS	Digital I/O.
CANTX			O	CMOS	CAN bus TX.
CTED1			I	ST	CTMU Edge 1 input.
INT2			I	ST	External Interrupt 2.
RB3/CANRX/CTED2/ INT3	36	11			
RB3			I/O	ST/ CMOS	Digital I/O.
CANRX			I	ST	CAN bus RX.
CTED2			I	ST	CTMU Edge 2 input.
INT3			I	ST	External Interrupt 3.
RB4/AN9/CTPLS/KBI0	37	14			
RB4			I/O	ST/ CMOS	Digital I/O.
AN9			I	Analog	Analog Input 9.
CTPLS			O	ST	CTMU pulse generator output.
KBI0			I	ST	Interrupt-on-change pin.
RB5/T0CKI/T3CKI/CCP5/ KBI1	38	15			
RB5			I/O	ST/ CMOS	Digital I/O.
T0CKI			I	ST	Timer0 external clock input.
T3CKI			I	ST	Timer3 external clock input.
CCP5			I/O	ST	Capture 5 input/Compare 5 output/PWM5 output.
KBI1			I	ST	Interrupt-on-change pin.

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

TABLE 1-6: PIC18F6XK80 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Num	Pin Type	Buffer Type	Description
				PORTB is a bidirectional I/O port.
RB0/AN10/FLT0/INT0	13			
RB0		I/O	ST/ CMOS	Digital I/O.
AN10		I	Analog	Analog Input 10.
FLT0		I	ST	Enhanced PWM Fault input for ECCP1.
INT0		I	ST	External Interrupt 0.
RB1/AN8/CTDIN/INT1	14			
RB1		I/O	ST/ CMOS	Digital I/O.
AN8		I	Analog	Analog Input 8.
CTDIN		I	ST	CTMU pulse delay input.
INT1		I	ST	External Interrupt 1.
RB2/CANTX/CTED1/ INT2	15			
RB2		I/O	ST/ CMOS	Digital I/O.
CANTX		O	CMOS	CAN bus TX.
CTED1		I	ST	CTMU Edge 1 input.
INT2		I	ST	External Interrupt 2.
RB3/CANRX/CTED2/ INT3	16			
RB3		I/O	ST/ CMOS	Digital I/O.
CANRX		I	ST	CAN bus RX.
CTED2		I	ST	CTMU Edge 2 input.
INT3		I	ST	External Interrupt 3.
RB4/AN9/CTPLS/KBI0	20			
RB4		I/O	ST/ CMOS	Digital I/O.
AN9		I	Analog	Analog Input 9.
CTPLS		O	ST	CTMU pulse generator output.
KBI0		I	ST	Interrupt-on-change pin.
RB5/T0CKI/T3CKI/CCP5/ KBI1	21			
RB5		I/O	ST/ CMOS	Digital I/O.
T0CKI		I	ST	Timer0 external clock input.
T3CKI		I	ST	Timer3 external clock input.
CCP5		I/O	ST/ CMOS	Capture 5 input/Compare 5 output/PWM5 output.
KBI1		I	ST	Interrupt-on-change pin.

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

4.0 POWER-MANAGED MODES

The PIC18F66K80 family of devices offers a total of seven operating modes for more efficient power management. These modes provide a variety of options for selective power conservation in applications where resources may be limited (such as battery-powered devices).

There are three categories of power-managed mode:

- Run modes
- Idle modes
- Sleep mode

There is an Ultra Low-Power Wake-up (ULPWU) for waking from Sleep mode.

These categories define which portions of the device are clocked, and sometimes, at what speed. The Run and Idle modes may use any of the three available clock sources (primary, secondary or internal oscillator block). The Sleep mode does not use a clock source.

The ULPWU mode, on the RA0 pin, enables a slow falling voltage to generate a wake-up, even from Sleep, without excess current consumption. (See **Section 4.7 "Ultra Low-Power Wake-up"**.)

The power-managed modes include several power-saving features offered on previous PIC® devices. One is the clock switching feature, offered in other PIC18 devices. This feature allows the controller to use the SOSC oscillator instead of the primary one. Another power-saving feature is Sleep mode, offered by all PIC devices, where all device clocks are stopped.

4.1 Selecting Power-Managed Modes

Selecting a power-managed mode requires two decisions:

- Will the CPU be clocked or not
- What will be the clock source

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

4.1.1 CLOCK SOURCES

The SCS<1:0> bits select one of three clock sources for power-managed modes. Those sources are:

- The primary clock as defined by the FOSC<3:0> Configuration bits
- The Secondary Clock (the SOSC oscillator)
- The Internal Oscillator block (for LF-INTOSC modes)

4.1.2 ENTERING POWER-MANAGED MODES

Switching from one power-managed mode to another begins by loading the OSCCON register. The SCS<1:0> bits select the clock source and determine which Run or Idle mode is used. Changing these bits causes an immediate switch to the new clock source, assuming that it is running. The switch may also be subject to clock transition delays. These considerations are discussed in **Section 4.1.3 "Clock Transitions and Status Indicators"** and subsequent sections.

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

Depending on the current and impending mode, a change to a power-managed mode does not always require setting all of the previously discussed bits. Many transitions can be done by changing the oscillator select bits, or changing the IDLEN bit, prior to issuing a SLEEP instruction. If the IDLEN bit is already configured as desired, it may only be necessary to perform a SLEEP instruction to switch to the desired mode.

TABLE 4-1: POWER-MANAGED MODES

Mode	OSCCON Bits		Module Clocking		Available Clock and Oscillator Source
	IDLEN<7>(1)	SCS<1:0>	CPU	Peripherals	
Sleep	0	N/A	Off	Off	None – All clocks are disabled
PRI_RUN	N/A	00	Clocked	Clocked	Primary – XT, LP, HS, EC, RC and PLL modes. This is the normal, full-power execution mode.
SEC_RUN	N/A	01	Clocked	Clocked	Secondary – SOSC Oscillator
RC_RUN	N/A	1x	Clocked	Clocked	Internal oscillator block(2)
PRI_IDLE	1	00	Off	Clocked	Primary – LP, XT, HS, RC, EC
SEC_IDLE	1	01	Off	Clocked	Secondary – SOSC oscillator
RC_IDLE	1	1x	Off	Clocked	Internal oscillator block(2)

Note 1: IDLEN reflects its value when the SLEEP instruction is executed.

2: Includes INTOSC (HF-INTOSC and MG-INTOSC) and INTOSC postscaler, as well as the LF-INTOSC source.

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

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
F09h	TXB2D3	TXB2D37	TXB2D36	TXB2D35	TXB2D34	TXB2D33	TXB2D32	TXB2D31	TXB2D30	95
F08h	TXB2D2	TXB2D27	TXB2D26	TXB2D25	TXB2D24	TXB2D23	TXB2D22	TXB2D21	TXB2D20	95
F07h	TXB2D1	TXB2D17	TXB2D16	TXB2D15	TXB2D14	TXB2D13	TXB2D12	TXB2D11	TXB2D10	95
F06h	TXB2D0	TXB2D07	TXB2D06	TXB2D05	TXB2D04	TXB2D03	TXB2D02	TXB2D01	TXB2D00	95
F05h	TXB2DLC	—	TXRTR	—	—	DLC3	DLC2	DLC1	DLC0	95
F04h	TXB2EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	95
F03h	TXB2EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	95
F02h	TXB2SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	95
F01h	TXB2SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	95
F00h	TXB2CON	TXBIF	TXABT	TXLARB	TXERR	TXREQ	—	TXPRI1	TXPRI0	95
EFFh	RXM1EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	95
EFEh	RXM1EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	95
EFDh	RXM1SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	95
EFCh	RXM1SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	95
EFBh	RXM0EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	95
EFAh	RXM0EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	95
EF9h	RXM0SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	95
EF8h	RXM0SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	95
EF7h	RXF5EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	95
EF6h	RXF5EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	95
EF5h	RXF5SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	95
EF4h	RXF5SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	95
EF3h	RXF4EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	95
EF2h	RXF4EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	95
EF1h	RXF4SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	95
EF0h	RXF4SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	95
EEFh	RXF3EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	95
EEEh	RXF3EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	95
EEDh	RXF3SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	95
EECh	RXF3SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	95
EEBh	RXF2EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	95
EEAh	RXF2EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	95
EE9h	RXF2SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	95
EE8h	RXF2SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	95
EE7h	RXF1EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	95
EE6h	RXF1EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	95
EE5h	RXF1SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	95
EE4h	RXF1SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	95
EE3h	RXF0EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	95
EE2h	RXF0EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	95
EE1h	RXF0SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	95
EE0h	RXF0SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	95
EDFh	CANCON_RO4	CANCON_RO4								95
EDEh	CANSTAT_RO4	CANSTAT_RO4								95
EDDh	B5D7	B5D77	B5D76	B5D75	B5D74	B5D73	B5D72	B5D71	B5D70	95
EDCh	B5D6	B5D67	B5D66	B5D65	B5D64	B5D63	B5D62	B5D61	B5D60	95
EDBh	B5D5	B5D57	B5D56	B5D55	B5D54	B5D53	B5D52	B5D51	B5D50	95
EDAh	B5D4	B5D47	B5D46	B5D45	B5D44	B5D43	B5D42	B5D41	B5D40	95
ED9h	B5D3	B5D37	B5D36	B5D35	B5D34	B5D33	B5D32	B5D31	B5D30	95
ED8h	B5D2	B5D27	B5D26	B5D25	B5D24	B5D23	B5D22	B5D21	B5D20	95
ED7h	B5D1	B5D17	B5D16	B5D15	B5D14	B5D13	B5D12	B5D11	B5D10	95

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”**.

REGISTER 10-18: IPR5: PERIPHERAL INTERRUPT PRIORITY REGISTER 5

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
IRXIP	WAKIP	ERRIP	TXB2IP	TXB1IP	TXB0IP	RXB1IP	RXB0IP/ FIFOFIE
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 **IRXIP:** Invalid Message Received Interrupt Priority bits
1 = High priority
0 = Low priority

bit 6 **WAKIP:** Bus Wake-up Activity Interrupt Priority bit
1 = High priority
0 = Low priority

bit 5 **ERRIP:** CAN Bus Error Interrupt Priority bit
1 = High priority
0 = Low priority

bit 4 **TXB2IP:** Transmit Buffer 2 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **TXB1IP:** Transmit Buffer 1 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 2 **TXB0IP:** Transmit Buffer 0 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **RXB1IP:** Receive Buffer 1 Interrupt Priority bit
Mode 0:
1 = High priority for Receive Buffer 1
0 = Low priority for Receive Buffer 1
Modes 1 and 2:
1 = High priority for received messages
0 = Low priority for received messages

bit 0 **RXB0IP/FIFOFIP:** Receive Buffer 0 Interrupt Priority bit
Mode 0:
1 = High priority for Receive Buffer 0
0 = Low priority for Receive Buffer 0
Mode 1:
Unimplemented: Read as '0'
Mode 2:
FIFOFIE: FIFO Full Interrupt Flag bit
1 = High priority
0 = Low priority

REGISTER 18-2: CTMUCONL: CTMU CONTROL LOW 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
EDG2POL	EDG2SEL1	EDG2SEL0	EDG1POL	EDG1SEL1	EDG1SEL0	EDG2STAT	EDG1STAT
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 **EDG2POL:** Edge 2 Polarity Select bit
1 = Edge 2 is programmed for a positive edge response
0 = Edge 2 is programmed for a negative edge response

bit 6-5 **EDG2SEL<1:0>:** Edge 2 Source Select bits
11 = CTED1 pin
10 = CTED2 pin
01 = ECCP1 Special Event Trigger
00 = CCP2 Special Event Trigger

bit 4 **EDG1POL:** Edge 1 Polarity Select bit
1 = Edge 1 is programmed for a positive edge response
0 = Edge 1 is programmed for a negative edge response

bit 3-2 **EDG1SEL<1:0>:** Edge 1 Source Select bits
11 = CTED1 pin
10 = CTED2 pin
01 = ECCP1 Special Event Trigger
00 = CCP2 Special Event Trigger

bit 1 **EDG2STAT:** Edge 2 Status bit
1 = Edge 2 event has occurred
0 = Edge 2 event has not occurred

bit 0 **EDG1STAT:** Edge 1 Status bit
1 = Edge 1 event has occurred
0 = Edge 1 event has not occurred

REGISTER 21-2: SSPCON1: MSSP CONTROL REGISTER 1 (SPI MODE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV(1)	SSPEN(2)	CKP	SSPM3(3)	SSPM2(3)	SSPM1(3)	SSPM0(3)
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 **WCOL:** Write Collision Detect bit
- 1 = The SSPBUF register is written while it is still transmitting the previous word (must be cleared in software)
- 0 = No collision

bit 6 **SSPOV:** Receive Overflow Indicator bit(1)

SPI Slave mode:
- 1 = A new byte is received while the SSPBUF register is still holding the previous data. In case of overflow, the data in SSPSR is lost. Overflow can only occur in Slave mode. The user must read the SSPBUF, even if only transmitting data, to avoid setting overflow (must be cleared in software).
- 0 = No overflow

bit 5 **SSPEN:** Master Synchronous Serial Port Enable bit(2)
- 1 = Enables the serial port and configures SCK, SDO, SDI and $\overline{SS}$ as serial port pins
- 0 = Disables the serial port and configures these pins as I/O port pins

bit 4 **CKP:** Clock Polarity Select bit
- 1 = Idle state for clock is a high level
- 0 = Idle state for clock is a low level

bit 3-0 **SSPM<3:0>:** Master Synchronous Serial Port Mode Select bits(3)
- 1010 = SPI Master mode: clock = FOSC/8
- 0101 = SPI Slave mode: clock = SCK pin; $\overline{SS}$ pin control disabled; $\overline{SS}$ can be used as I/O pin
- 0100 = SPI Slave mode: clock = SCK pin; $\overline{SS}$ pin control enabled
- 0011 = SPI Master mode: clock = TMR2 output/2
- 0010 = SPI Master mode: clock = FOSC/64
- 0001 = SPI Master mode: clock = FOSC/16
- 0000 = SPI Master mode: clock = FOSC/4

Note 1: In Master mode, the overflow bit is not set since each new reception (and transmission) is initiated by writing to the SSPBUF register.

2: When enabled, these pins must be properly configured as inputs or outputs.

3: Bit combinations not specifically listed here are either reserved or implemented in I^2C mode only.

21.4.3.4 7-Bit Address Masking Mode

Unlike 5-bit masking, 7-Bit Address Masking mode uses a mask of up to 8 bits (in 10-bit addressing) to define a range of addresses that can be Acknowledged, using the lowest bits of the incoming address. This allows the module to Acknowledge up to 127 different addresses with 7-bit addressing, or 255 with 10-bit addressing (see Example 21-3). This mode is the default configuration of the module, which is selected when MSSPMSK is unprogrammed ('1').

The address mask for 7-Bit Address Masking mode is stored in the SSPMSK register, instead of the SSPCON2 register. SSPMSK is a separate hardware register within the module, but it is not directly addressable. Instead, it shares an address in the SFR space with the SSPADD register. To access the SSPMSK register, it is necessary to select MSSP mode, '1001' (SSPCON1<3:0> = 1001) and then read or write to the location of SSPADD.

To use 7-Bit Address Masking mode, it is necessary to initialize SSPMSK with a value before selecting the I^2C Slave Addressing mode. Thus, the required sequence of events is:

1. Select SSPMSK Access mode (SSPCON2<3:0> = 1001).
2. Write the mask value to the appropriate SSPADD register address (FC8h).
3. Set the appropriate I^2C Slave mode (SSPCON2<3:0> = 0111 for 10-bit addressing, 0110 for 7-bit addressing).

Setting or clearing mask bits in SSPMSK behaves in the opposite manner of the ADMSKx bits in 5-Bit Address Masking mode. That is, clearing a bit in SSPMSK causes the corresponding address bit to be masked; setting the bit requires a match in that position. SSPMSK resets to all '1's upon any Reset condition and, therefore, has no effect on the standard MSSP operation until written with a mask value.

With 7-bit addressing, SSPMSK<7:1> bits mask the corresponding address bits in the SSPADD register. For any SSPMSK bits that are active (SSPMSK<n> = 0), the corresponding SSPADD address bit is ignored (SSPADD<n> = x). For the module to issue an address Acknowledge, it is sufficient to match only on addresses that do not have an active address mask.

With 10-bit addressing, SSPMSK<7:0> bits mask the corresponding address bits in the SSPADD register. For any SSPMSK bits that are active (= 0), the corresponding SSPADD address bit is ignored (SSPADD<n> = x).

Note:	The two Most Significant bits of the address are not affected by address masking.

EXAMPLE 21-3: ADDRESS MASKING EXAMPLES IN 7-BIT MASKING MODE

7-Bit Addressing:

SSPADD<7:1> = 1010 000

SSPMSK<7:1> = 1111 001

Addresses Acknowledged = ACh, A8h, A4h, A0h

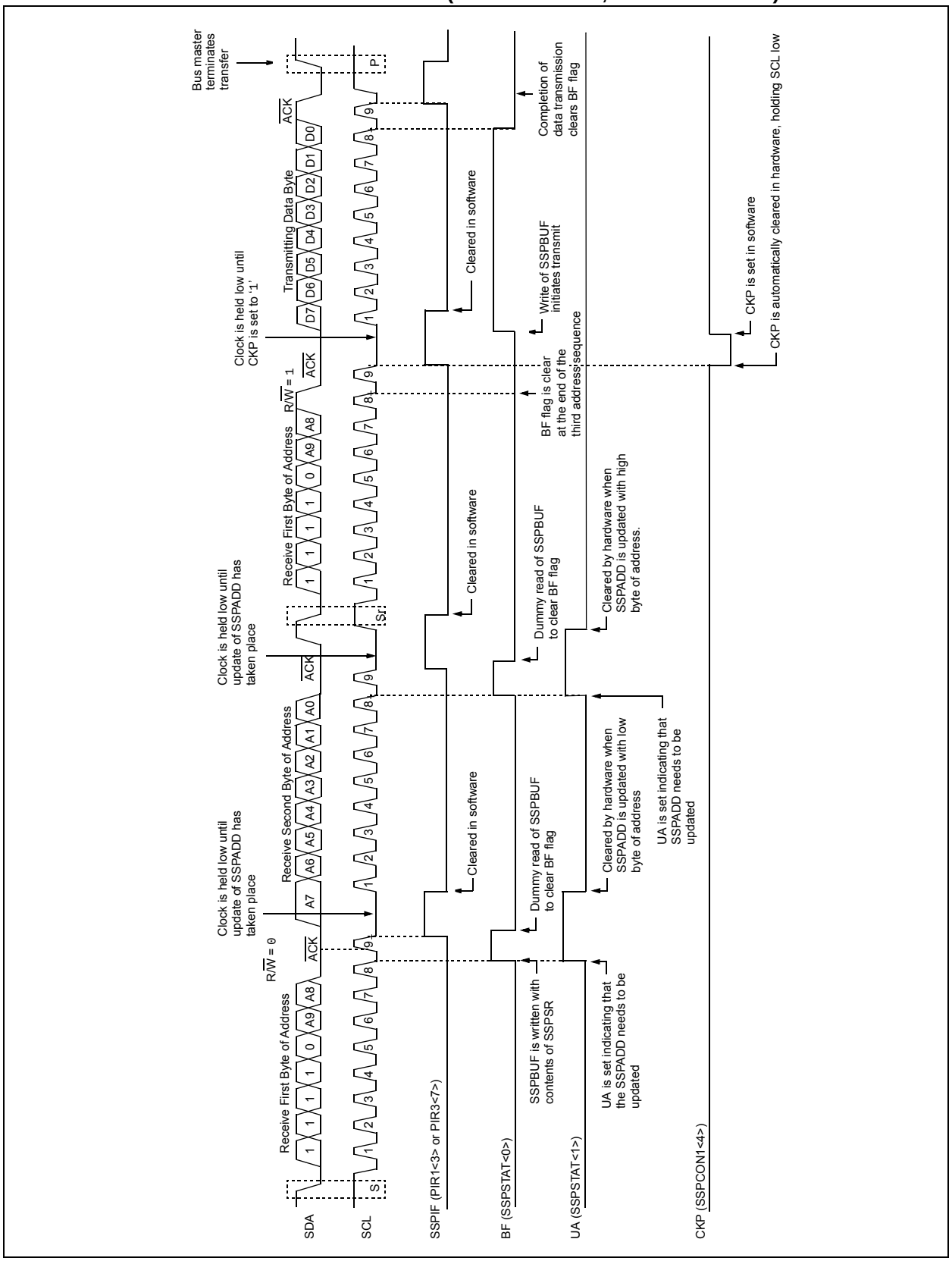
10-Bit Addressing:

SSPADD<7:0> = 1010 0000 (The two MSb are ignored in this example since they are not affected)

SSPMSK<5:1> = 1111 0011

Addresses Acknowledged = ACh, A8h, A4h, A0h

FIGURE 21-13: I²C™ SLAVE MODE TIMING (TRANSMISSION, 10-BIT ADDRESS)



22.3.4.2 Special Considerations Using the WUE Bit

The timing of WUE and RCxIF events may cause some confusion when it comes to determining the validity of received data. As noted, setting the WUE bit places the EUSARTx in an Idle mode. The wake-up event causes a receive interrupt by setting the RCxIF bit. The WUE bit is cleared after this when a rising edge is seen on RXx/DTx. The interrupt condition is then cleared by reading the RCREGx register. Ordinarily, the data in RCREGx will be dummy data and should be discarded.

The fact that the WUE bit has been cleared (or is still set) and the RCxIF flag is set should not be used as an indicator of the integrity of the data in RCREGx. Users should consider implementing a parallel method in firmware to verify received data integrity.

To assure that no actual data is lost, check the RCIDL bit to verify that a receive operation is not in process. If a receive operation is not occurring, the WUE bit may then be set just prior to entering the Sleep mode.

FIGURE 22-8: AUTO-WAKE-UP BIT (WUE) TIMINGS DURING NORMAL OPERATION

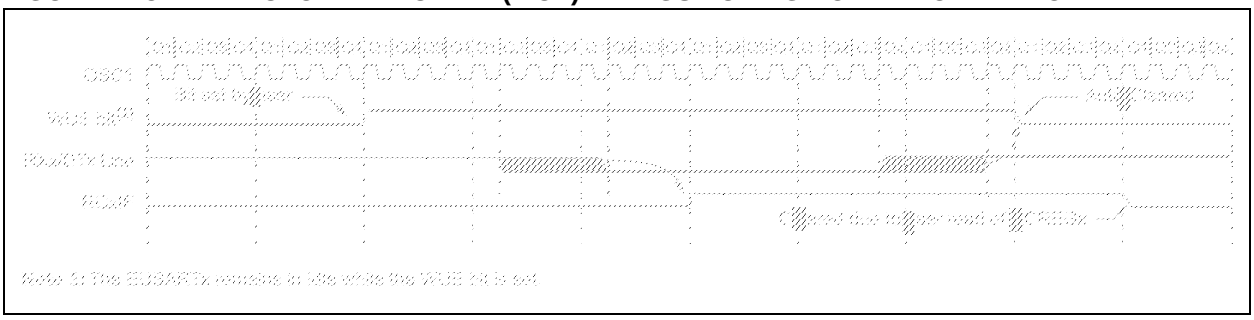
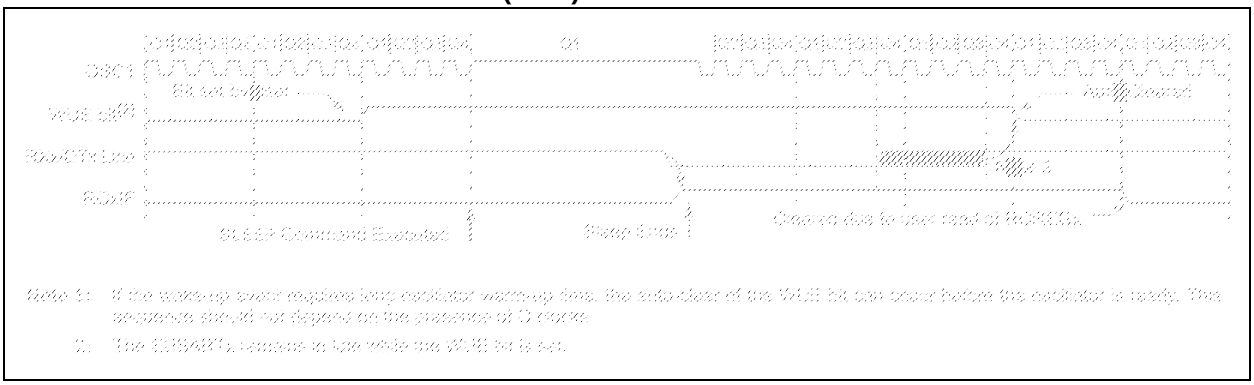


FIGURE 22-9: AUTO-WAKE-UP BIT (WUE) TIMINGS DURING SLEEP



REGISTER 27-14: RXB1CON: RECEIVE BUFFER 1 CONTROL REGISTER (CONTINUED)

bit 2-0 Mode 0:

FILHIT<2:0>: Filter Hit bits

These bits indicate which acceptance filter enabled the last message reception into Receive Buffer 1.

111 = Reserved
110 = Reserved
101 = Acceptance Filter 5 (RXF5)
100 = Acceptance Filter 4 (RXF4)
011 = Acceptance Filter 3 (RXF3)
010 = Acceptance Filter 2 (RXF2)
001 = Acceptance Filter 1 (RXF1), only possible when RXB0DBEN bit is set
000 = Acceptance Filter 0 (RXF0), only possible when RXB0DBEN bit is set

Mode 1, 2:

FILHIT<4:0>: Filter Hit bits<2:0>

These bits, in combination with FILHIT<4:3>, indicate which acceptance filter enabled the message reception into this receive buffer.

01111 = Acceptance Filter 15 (RXF15)
01110 = Acceptance Filter 14 (RXF14)
...
00000 = Acceptance Filter 0 (RXF0)

Note 1: This bit is set by the CAN module upon receiving a message and must be cleared by software after the buffer is read. As long as RXFUL is set, no new message will be loaded and the buffer will be considered full.

REGISTER 27-15: RXBnSIDH: RECEIVE BUFFER 'n' STANDARD IDENTIFIER REGISTERS, HIGH BYTE [0 ≤ n ≤ 1]

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3
bit 7							bit 0

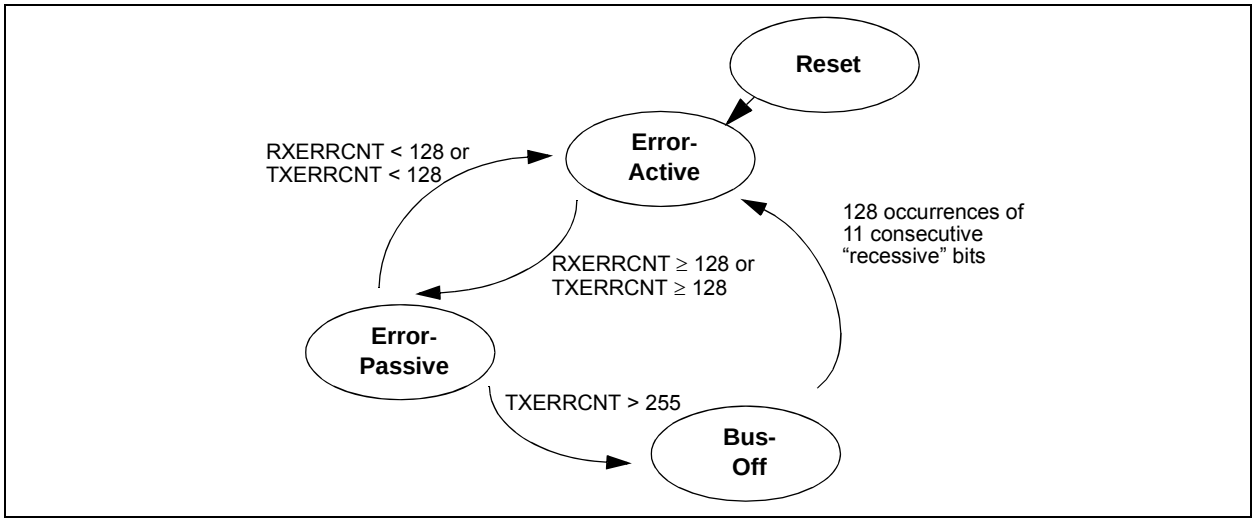
Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-0 **SID<10:3>:** Standard Identifier bits (if EXID (RXBnSIDL<3>) = 0)
Extended Identifier bits, EID<28:21> (if EXID = 1).

The PIC18F66K80 family devices are error-active if both error counters are below the error-passive limit of 128. They are error-passive if at least one of the error counters equals or exceeds 128. They go to bus-off if the transmit error counter equals or exceeds the bus-off limit of 256. The devices remain in this state until the bus-off recovery sequence is finished. The bus-off recovery sequence consists of 128 occurrences of 11 consecutive recessive bits (see Figure 27-8). Note that the CAN module, after going bus-off, will recover back to error-active without any intervention by the MCU if the bus remains Idle for 128 x 11 bit times. If this is not desired, the error Interrupt Service Routine should address this. The current Error mode of the CAN module can be read by the MCU via the COMSTAT register.

Additionally, there is an Error State Warning flag bit, EWARN, which is set if at least one of the error counters equals or exceeds the error warning limit of 96. EWARN is reset if both error counters are less than the error warning limit.

FIGURE 27-8: ERROR MODES STATE DIAGRAM



27.15 CAN Interrupts

The module has several sources of interrupts. Each of these interrupts can be individually enabled or disabled. The PIR5 register contains interrupt flags. The PIE5 register contains the enables for the 8 main interrupts. A special set of read-only bits in the CANSTAT register, the ICODE bits, can be used in combination with a jump table for efficient handling of interrupts.

All interrupts have one source, with the exception of the error interrupt and buffer interrupts in Mode 1 and 2. Any of the error interrupt sources can set the error interrupt flag. The source of the error interrupt can be determined by reading the Communication Status register, COMSTAT. In Mode 1 and 2, there are two interrupt enable/disable and flag bits – one for all transmit buffers and the other for all receive buffers.

The interrupts can be broken up into two categories: receive and transmit interrupts.

The receive related interrupts are:

- Receive Interrupts
- Wake-up Interrupt
- Receiver Overrun Interrupt
- Receiver Warning Interrupt
- Receiver Error-Passive Interrupt

The transmit related interrupts are:

- Transmit Interrupts
- Transmitter Warning Interrupt
- Transmitter Error-Passive Interrupt
- Bus-Off Interrupt

28.6 Program Verification and Code Protection

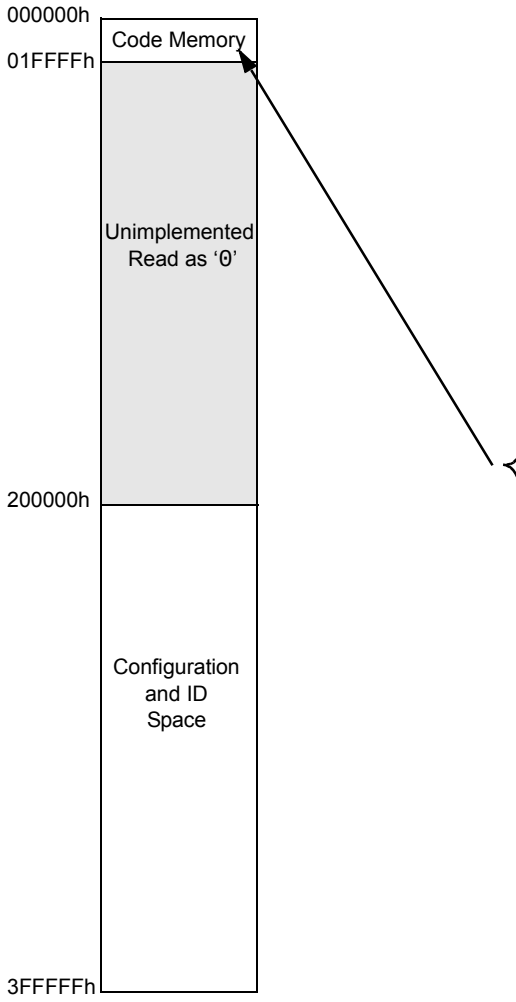
The user program memory is divided into four blocks. One of these is a boot block of 1 or 2 Kbytes. The remainder of the memory is divided into blocks on binary boundaries.

Each of the blocks has three code protection bits associated with them. They are:

- Code-Protect bit (CPx)
- Write-Protect bit (WRTx)
- External Block Table Read bit (EBTRx)

Figure 28-6 shows the program memory organization for 48, 64, 96 and 128 Kbyte devices and the specific code protection bit associated with each block. The actual locations of the bits are summarized in Table 28-4.

FIGURE 28-6: CODE-PROTECTED PROGRAM MEMORY FOR THE PIC18F66K80 FAMILY



Device/Memory Size(1)

PIC18FX6K80		PIC18FX5K80		
BBSIZ = 1	**BBSIZ = 0**	**BBSIZ = 1**	**BBSIZ = 0**	**Address**
Boot Block 2 kW	Boot Block	Boot Block 2 kW	Boot Block	0000h
	Block 0 7 kW		Block 0 3 kW	0800h
Block 0 6 kW		Block 0 2 kW		1000h 1FFFh
		Block 1 4 kW	Block 1 4 kW	2000h 3FFFh
Block 1 8 kW	Block 1 8 kW	Block 2 4 kW	Block 2 4 kW	4000h 5FFFh
		Block 3 4 kW	Block 3 4 kW	6000h 7FFFh
Block 2 8 kW	Block 2 8 kW			8000h BFFFh
Block 3 8 kW	Block 3 8 kW			C000h FFFFh
				10000h 13FFFh
				14000h 17FFFh
				18000h 1BFFFh
				1C000h 1FFFFh

Note 1: Sizes of memory areas are not to scale.

2: Boot block size is determined by the BBSIZ0 bit (CONFIG4L<4>).

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

Syntax: BTFSC f, b {,a}

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

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

Status Affected: None

Encoding:

1011	bbba	ffff	ffff

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

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 $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

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

If skip:

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

If skip and followed by 2-word instruction:

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

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

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

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

Syntax: BTFSS f, b {,a}

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

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

Status Affected: None

Encoding:

1010	bbba	ffff	ffff

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

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 $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

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

If skip:

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

If skip and followed by 2-word instruction:

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

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

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

29.2.3 BYTE-ORIENTED AND BIT-ORIENTED INSTRUCTIONS IN INDEXED LITERAL OFFSET MODE

Note: Enabling the PIC18 instruction set extension may cause legacy applications to behave erratically or fail entirely.

In addition to eight new commands in the extended set, enabling the extended instruction set also enables Indexed Literal Offset Addressing (**Section 6.6.1 "Indexed Addressing with Literal Offset"**). This has a significant impact on the way that many commands of the standard PIC18 instruction set are interpreted.

When the extended set is disabled, addresses embedded in opcodes are treated as literal memory locations: either as a location in the Access Bank (a = 0) or in a GPR bank designated by the BSR (a = 1). When the extended instruction set is enabled and a = 0, however, a file register argument of 5Fh or less is interpreted as an offset from the pointer value in FSR2 and not as a literal address. For practical purposes, this means that all instructions that use the Access RAM bit as an argument – that is, all byte-oriented and bit-oriented instructions, or almost half of the core PIC18 instructions – may behave differently when the extended instruction set is enabled.

When the content of FSR2 is 00h, the boundaries of the Access RAM are essentially remapped to their original values. This may be useful in creating backward-compatible code. If this technique is used, it may be necessary to save the value of FSR2 and restore it when moving back and forth between C and assembly routines in order to preserve the Stack Pointer. Users must also keep in mind the syntax requirements of the extended instruction set (see **Section 29.2.3.1 "Extended Instruction Syntax with Standard PIC18 Commands"**).

Although the Indexed Literal Offset mode can be very useful for dynamic stack and pointer manipulation, it can also be very annoying if a simple arithmetic operation is carried out on the wrong register. Users who are accustomed to the PIC18 programming must keep in mind that, when the extended instruction set is enabled, register addresses of 5Fh or less are used for Indexed Literal Offset Addressing.

Representative examples of typical byte-oriented and bit-oriented instructions in the Indexed Literal Offset mode are provided on the following page to show how execution is affected. The operand conditions shown in the examples are applicable to all instructions of these types.

29.2.3.1 Extended Instruction Syntax with Standard PIC18 Commands

When the extended instruction set is enabled, the file register argument 'f' in the standard byte-oriented and bit-oriented commands is replaced with the literal offset value 'k'. As already noted, this occurs only when 'f' is less than or equal to 5Fh. When an offset value is used, it must be indicated by square brackets ("[ ]"). As with the extended instructions, the use of brackets indicates to the compiler that the value is to be interpreted as an index or an offset. Omitting the brackets, or using a value greater than 5Fh within the brackets, will generate an error in the MPASM™ Assembler.

If the index argument is properly bracketed for Indexed Literal Offset Addressing, the Access RAM argument is never specified; it will automatically be assumed to be '0'. This is in contrast to standard operation (extended instruction set disabled), when 'a' is set on the basis of the target address. Declaring the Access RAM bit in this mode will also generate an error in the MPASM Assembler.

The destination argument 'd' functions as before.

In the latest versions of the MPASM Assembler, language support for the extended instruction set must be explicitly invoked. This is done with either the command line option, /y, or the PE directive in the source listing.

29.2.4 CONSIDERATIONS WHEN ENABLING THE EXTENDED INSTRUCTION SET

It is important to note that the extensions to the instruction set may not be beneficial to all users. In particular, users who are not writing code that uses a software stack may not benefit from using the extensions to the instruction set.

Additionally, the Indexed Literal Offset Addressing mode may create issues with legacy applications written to the PIC18 assembler. This is because instructions in the legacy code may attempt to address registers in the Access Bank below 5Fh. Since these addresses are interpreted as literal offsets to FSR2 when the instruction set extension is enabled, the application may read or write to the wrong data addresses.

When porting an application to the PIC18F66K80 family, it is very important to consider the type of code. A large, re-entrant application that is written in C and would benefit from efficient compilation will do well when using the instruction set extensions. Legacy applications that heavily use the Access Bank will most likely not benefit from using the extended instruction set.

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	330	480	µA	-40°C	VDD = 1.8V(4) Regulator Disabled	FOSC = 4 MHz (**RC_IDLE** mode, Internal HF-INTOSC)
		330	480	µA	+25°C		
		330	480	µA	+60°C		
		340	500	µA	+85°C		
		350	540	µA	+125°C		
	PIC18**LF**XXK80	522	720	µA	-40°C	VDD = 3.3V(4) Regulator Disabled	
		522	720	µA	+25°C		
		522	720	µA	+60°C		
		540	740	µA	+85°C		
		550	780	µA	+125°C		
	PIC18**F**XXK80	540	760	µA	-40°C	VDD = 3.3V(5) Regulator Enabled	
		540	760	µA	+25°C		
		540	760	µA	+60°C		
		560	780	µA	+85°C		
		580	810	µA	+125°C		
	PIC18**F**XXK80	600	1250	µA	-40°C	VDD = 5V(5) Regulator Enabled	
		600	1250	µA	+25°C		
		600	1250	µA	+60°C		
		610	1300	µA	+85°C		
		620	1340	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.

31.3 DC Characteristics: PIC18F66K80 Family (Industrial) (Continued)

DC CHARACTERISTICS			Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended			
Param No.	**Symbol**	**Characteristic**	**Min**	**Max**	**Units**	**Conditions**
	VOH	**Output High Voltage(1)**				
D090		I/O Ports:			V	
		PORTA, PORTB, PORTC	VDD – 0.7	—	V	IOH = -3 mA, VDD = 5.5V, -40°C to +125°C
		PORTD, PORTE, PORTF, PORTG	VDD – 0.7	—	V	IOH = -2 mA, VDD = 5.5V, -40°C to +125°C
D092		OSC2/CLKO (INTOSC, EC modes)	VDD – 0.7	—	V	IOH = -1 mA, VDD = 5.5V, -40°C to +125°C
		Capacitive Loading Specs on Output Pins				
D100(4)	COSC2	OSC2 Pin	—	20	pF	In HS mode when external clock is used to drive OSC1
D101	CIO	All I/O Pins and OSC2	—	50	pF	To meet the AC Timing Specifications
D102	CB	SCL, SDA	—	400	pF	I^2C™ Specification

Note 1: Negative current is defined as current sourced by the pin.

S

T