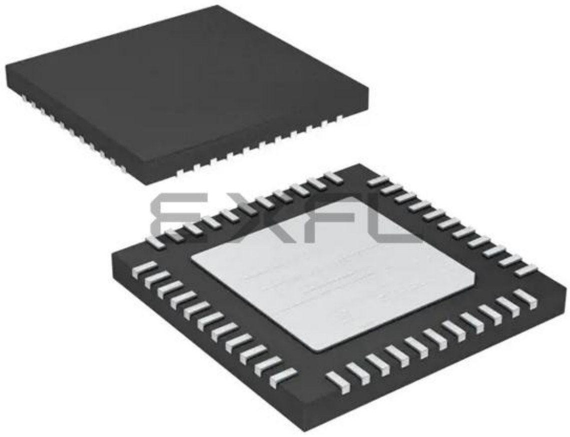


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

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

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	64MHz
Connectivity	ECANbus, I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	35
Program Memory Size	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 11x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-VQFN Exposed Pad
Supplier Device Package	44-QFN (8x8)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18lf45k80t-i-ml

2.0 GUIDELINES FOR GETTING STARTED WITH PIC18FXXKXX MICROCONTROLLERS

2.1 Basic Connection Requirements

Getting started with the PIC18F66K80 family family of 8-bit microcontrollers requires attention to a minimal set of device pin connections before proceeding with development.

The following pins must always be connected:

- All VDD and VSS pins (see **Section 2.2 "Power Supply Pins"**)
- All AVDD and AVSS pins, regardless of whether or not the analog device features are used (see **Section 2.2 "Power Supply Pins"**)
- MCLR pin (see **Section 2.3 "Master Clear (MCLR) Pin"**)
- ENVREG (if implemented) and VCAP/VDDCORE pins (see **Section 2.4 "Voltage Regulator Pins (ENVREG and VCAP/VDDCORE)"**)

These pins must also be connected if they are being used in the end application:

- PGC/PGD pins used for In-Circuit Serial Programming™ (ICSP™) and debugging purposes (see **Section 2.5 "ICSP Pins"**)
- OSCI and OSCO pins when an external oscillator source is used (see **Section 2.6 "External Oscillator Pins"**)

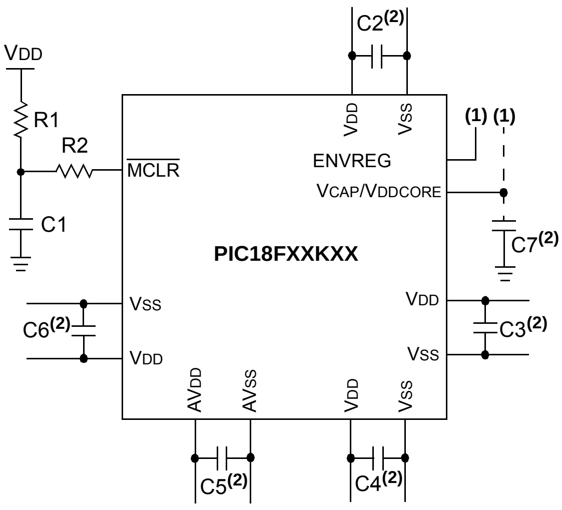
Additionally, the following pins may be required:

- VREF+/VREF- pins are used when external voltage reference for analog modules is implemented

Note:	The AVDD and AVSS pins must always be connected, regardless of whether any of the analog modules are being used.

The minimum mandatory connections are shown in Figure 2-1.

FIGURE 2-1: RECOMMENDED MINIMUM CONNECTIONS



Key (all values are recommendations):

C1 through C6: 0.1 µF, 20V ceramic

R1: 10 kΩ

R2: 100Ω to 470Ω

Note 1: See **Section 2.4 "Voltage Regulator Pins (ENVREG and VCAP/VDDCORE)"** for explanation of ENVREG pin connections.

2: The example shown is for a PIC18F device with five VDD/VSS and AVDD/AVSS pairs. Other devices may have more or less pairs; adjust the number of decoupling capacitors appropriately.

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

Register	Applicable Devices			Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets, WDT Reset, RESET Instruction, Stack Resets	Wake-up via WDT or Interrupt
RXF12SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF11EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF11EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF11SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXF11SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF10EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF10EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF10SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXF10SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF9EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF9EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF9SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXF9SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF8EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF8EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF8SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXF8SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF7EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF7EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF7SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXF7SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF6EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF6EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF6SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXF6SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXFCON0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
RXFCON1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
BRGCON3	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	00-- -000	00-- -000	uu-- -uuu
BRGCON2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
BRGCON1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
TXERRCNT	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
RXERRCNT	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).

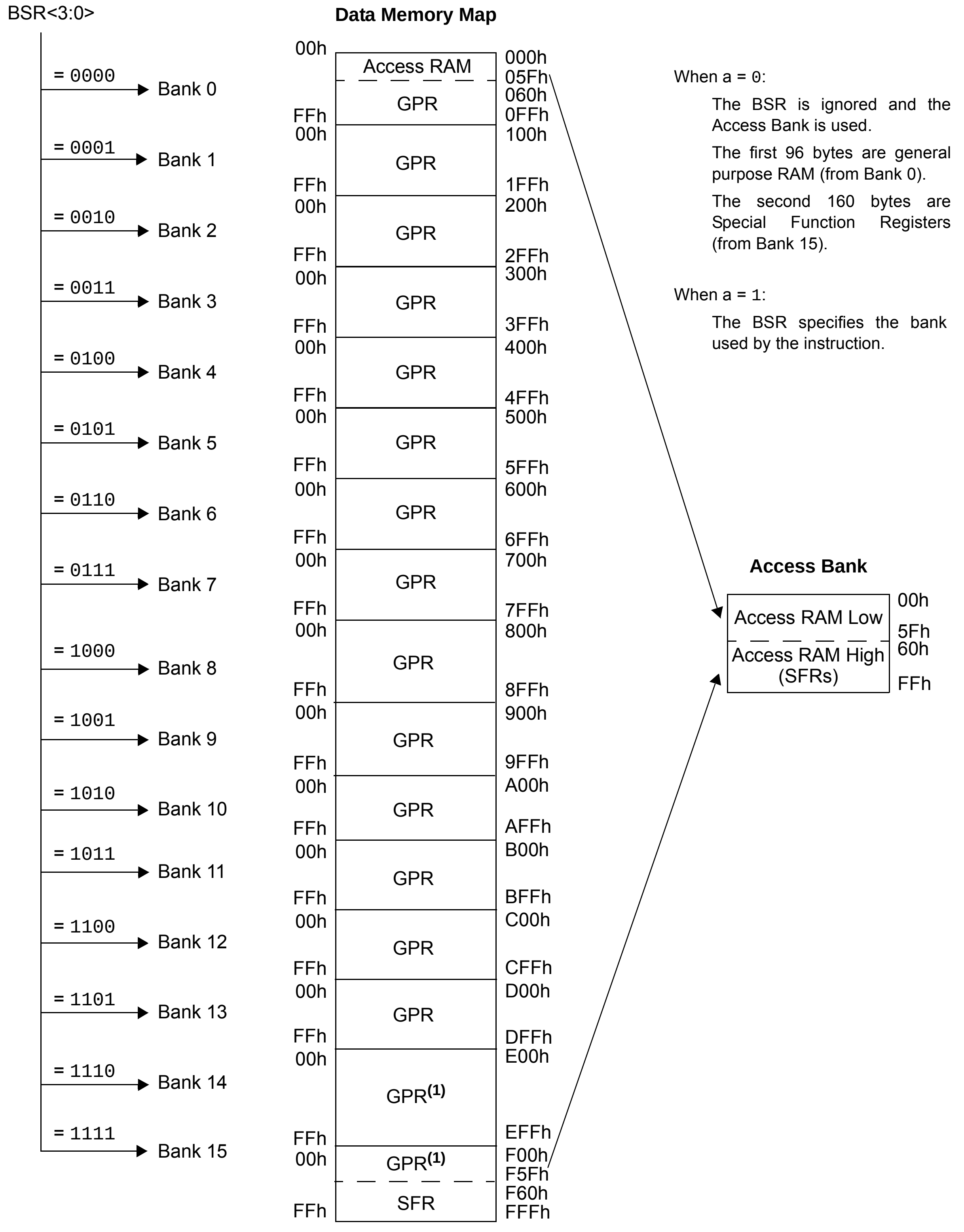
2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

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'.

FIGURE 6-6: DATA MEMORY MAP FOR PIC18FX5K80 AND PIC18FX6K80 DEVICES



When a = 0:

The BSR is ignored and the Access Bank is used.

The first 96 bytes are general purpose RAM (from Bank 0).

The second 160 bytes are Special Function Registers (from Bank 15).

When a = 1:

The BSR specifies the bank used by the instruction.

Note 1: Addresses, E41h through F5Fh, are also used by SFRs, but are not part of the Access RAM. Users must always use the complete address, or load the proper BSR value, to access these registers.

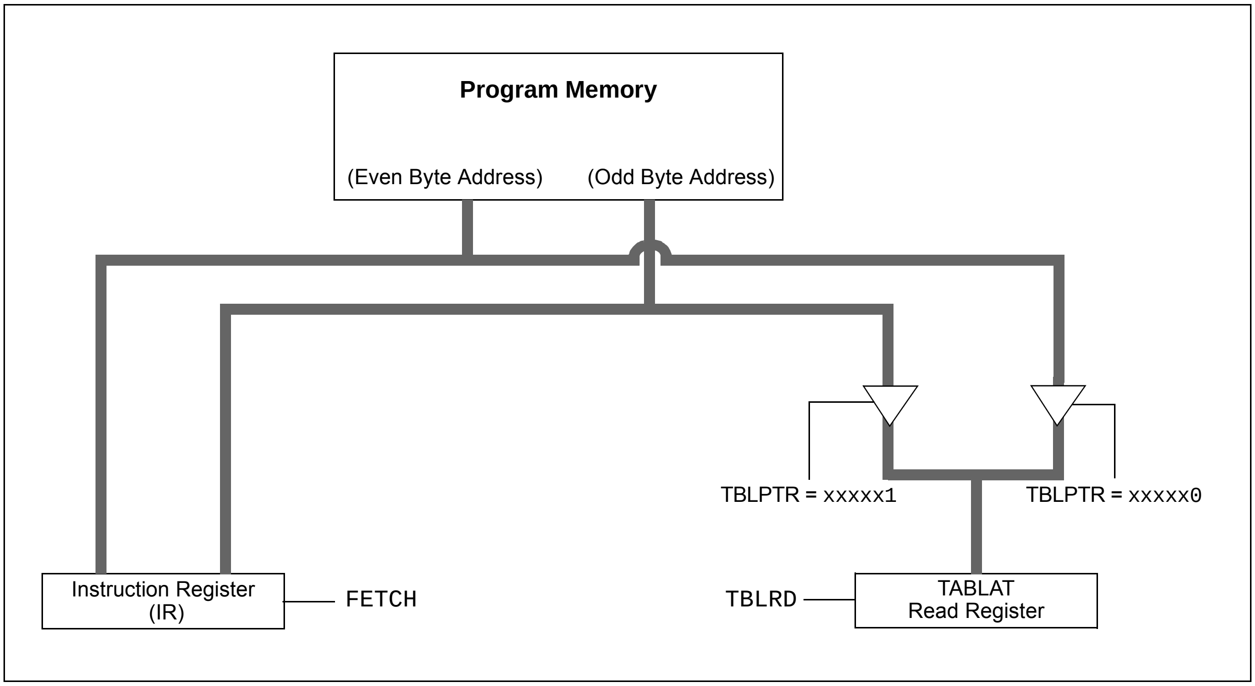
7.3 Reading the Flash Program Memory

The TBLRD instruction is used to retrieve data from program memory and places it into data RAM. Table reads from program memory are performed one byte at a time.

TBLPTR points to a byte address in program space. Executing TBLRD places the byte pointed to into TABLAT. In addition, TBLPTR can be modified automatically for the next table read operation.

The internal program memory is typically organized by words. The Least Significant bit of the address selects between the high and low bytes of the word. Figure 7-4 shows the interface between the internal program memory and the TABLAT.

FIGURE 7-4: READS FROM FLASH PROGRAM MEMORY



EXAMPLE 7-1: READING A FLASH PROGRAM MEMORY WORD

```
          MOVLW   CODE_ADDR_UPPER       ; Load TBLPTR with the base
          MOVWF   TBLPTRU               ; address of the word
          MOVLW   CODE_ADDR_HIGH
          MOVWF   TBLPTRH
          MOVLW   CODE_ADDR_LOW
          MOVWF   TBLPTRL
READ_WORD
          TBLRD*+                       ; read into TABLAT and increment
          MOVF    TABLAT, W             ; get data
          MOVWF   WORD_EVEN
          TBLRD*+                       ; read into TABLAT and increment
          MOVF    TABLAT, W             ; get data
          MOVF    WORD_ODD
```

FIGURE 12-4: CARRIER LOW SYNCHRONIZATION (MDCHSYNC = 0, MDCLSYNC = 1)

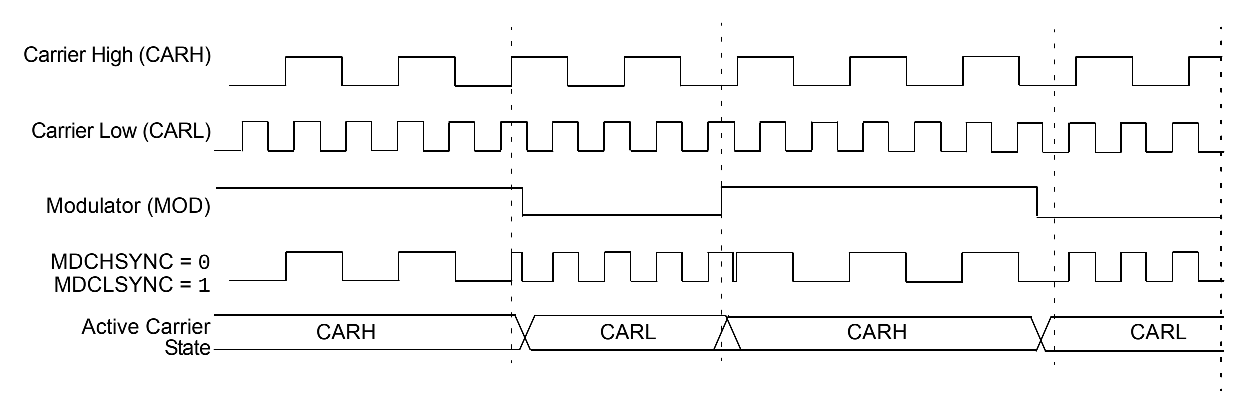
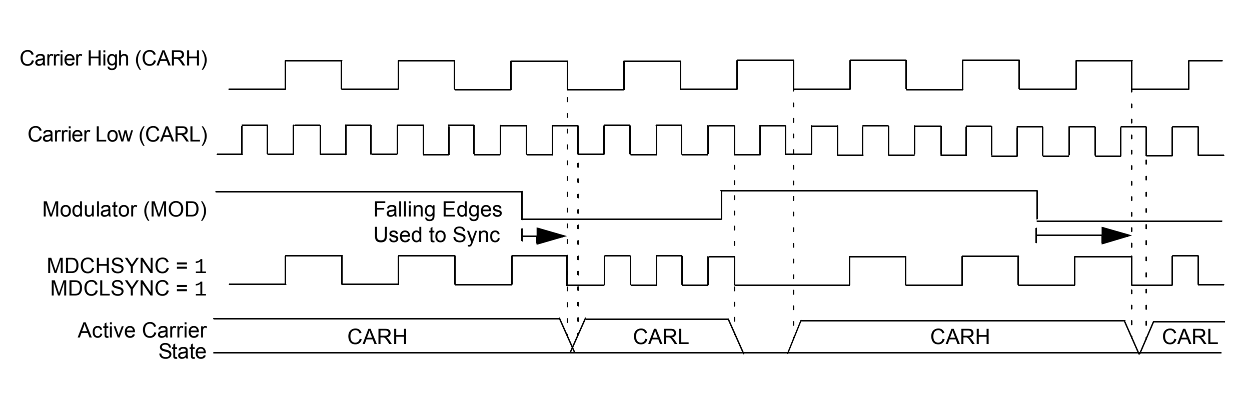


FIGURE 12-5: FULL SYNCHRONIZATION (MDCHSYNC = 1, MDCLSYNC = 1)



14.7 Resetting Timer1 Using the ECCP Special Event Trigger

If ECCP modules are configured to use Timer1 and to generate a Special Event Trigger in Compare mode (CCP1M<3:0> = 1011), this signal will reset Timer1. The trigger from ECCP will also start an A/D conversion if the A/D module is enabled. (For more information, see **Section 20.3.4 "Special Event Trigger"**.)

To take advantage of this feature, the module must be configured as either a timer or a synchronous counter. When used this way, the CCPR1H:CCPR1L register pair effectively becomes a Period register for Timer1.

If Timer1 is running in Asynchronous Counter mode, this Reset operation may not work.

In the event that a write to Timer1 coincides with a Special Event Trigger, the write operation will take precedence.

Note: The Special Event Trigger from the ECCP module will only clear the TMR1 register's content, but not set the TMR1IF interrupt flag bit (PIR1<0>).

14.8 Timer1 Gate

Timer1 can be configured to count freely or the count can be enabled and disabled using the Timer1 gate circuitry. This is also referred to as Timer1 gate count enable.

Timer1 gate can also be driven by multiple selectable sources.

14.8.1 TIMER1 GATE COUNT ENABLE

The Timer1 Gate Enable mode is enabled by setting the TMR1GE bit of the T1GCON register. The polarity of the Timer1 Gate Enable mode is configured using the T1GPOL bit (T1GCON<6>).

When Timer1 Gate Enable mode is enabled, Timer1 will increment on the rising edge of the Timer1 clock source. When Timer1 Gate Enable mode is disabled, no incrementing will occur and Timer1 will hold the current count. See Figure 14-4 for timing details.

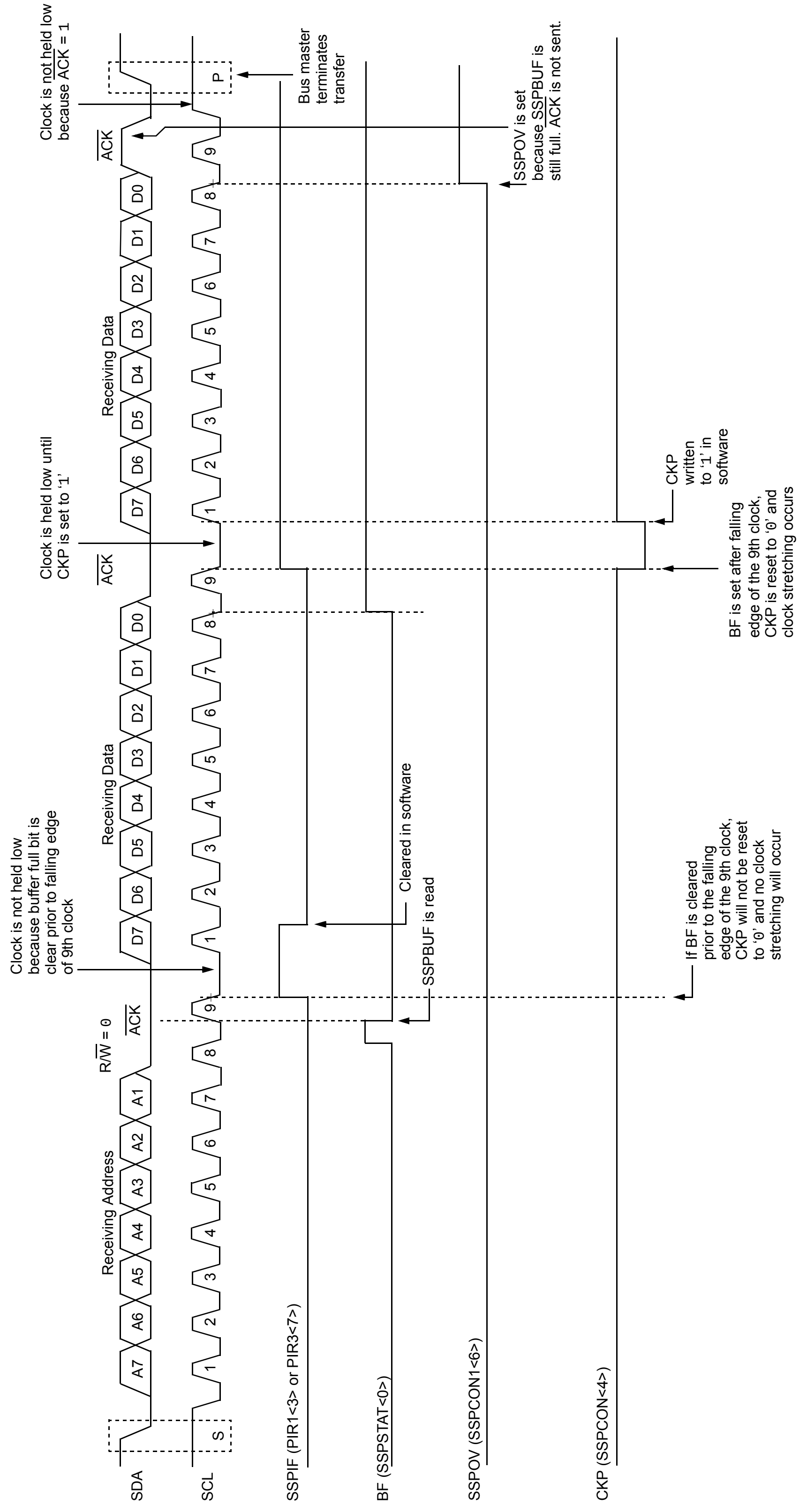
TABLE 14-3: TIMER1 GATE ENABLE SELECTIONS

T1CLK[†]	T1GPOL (T1GCON<6>)	T1G Pin	Timer1 Operation
↑	0	0	Counts
↑	0	1	Holds Count
↑	1	0	Holds Count
↑	1	1	Counts

† The clock on which TMR1 is running. For more information, see Figure 14-1.

Note: The CCP and ECCP modules use Timers, 1 through 4, for some modes. The assignment of a particular timer to a CCP/ECCP module is determined by the Timer to CCP enable bits in the CCPTMRS register. For more details, see Register 20-2 and Register 19-2.

FIGURE 21-15: I^{2}C™ SLAVE MODE TIMING WITH SEN = 1 (RECEPTION, 7-BIT ADDRESS)


SDA
SCL
SSPIF (PIR1<3> or PIR3<7>)
BF (SSPSTAT<0>)
SSPOV (SSPCON1<6>)
CKP (SSPCON<4>)
S
A7
A6
A5
A4
A3
A2
A1
Receiving Address
R/$\overline{W}$ = 0
$\overline{ACK}$
1
2
3
4
5
6
7
8
9
Clock is not held low because buffer full bit is clear prior to falling edge of 9th clock
D7
D6
D5
D4
D3
D2
D1
D0
Receiving Data
Cleared in software
SSPBUF is read
If BF is cleared prior to the falling edge of the 9th clock, CKP will not be reset to '0' and no clock stretching will occur
Clock is held low until CKP is set to '1'
BF is set after falling edge of the 9th clock, CKP is reset to '0' and clock stretching occurs
CKP written to '1' in software
SSPOV is set because SSPBUF is still full. ACK is not sent.
Clock is not held low because $\overline{ACK}$ = 1
Bus master terminates transfer
P

21.4.7 BAUD RATE

In I^{2}C Master mode, the Baud Rate Generator (BRG) reload value is placed in the 8 bits of the SSPADD register (Figure 21-19). When a write occurs to SSPBUF, the Baud Rate Generator will automatically begin counting. The BRG counts down to 0 and stops until another reload has taken place. The BRG count is decremented twice per instruction cycle (TCY) on the Q2 and Q4 clocks. In I^{2}C Master mode, the BRG is reloaded automatically.

Once the given operation is complete (i.e., transmission of the last data bit is followed by $\overline{\text{ACK}}$), the internal clock will automatically stop counting and the SCL pin will remain in its last state.

Table 21-3 demonstrates clock rates based on instruction cycles and the BRG value loaded into SSPADD. The SSPADD BRG value of 00h is not supported.

FIGURE 21-19: BAUD RATE GENERATOR BLOCK DIAGRAM

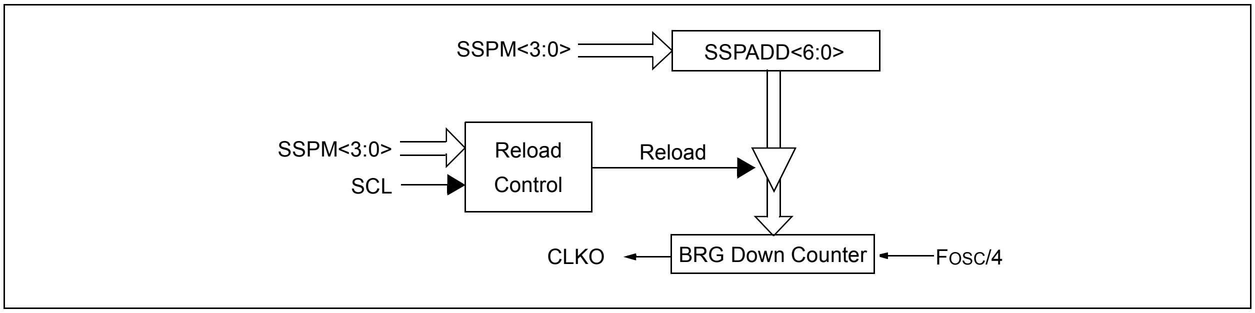


TABLE 21-3: I^{2}C™ CLOCK RATE w/BRG

FOSC	FCY	FCY * 2	BRG Value	FSCL (2 Rollovers of BRG)
40 MHz	10 MHz	20 MHz	18h	400 kHz(1)
40 MHz	10 MHz	20 MHz	1Fh	312.5 kHz
40 MHz	10 MHz	20 MHz	63h	100 kHz
16 MHz	4 MHz	8 MHz	09h	400 kHz(1)
16 MHz	4 MHz	8 MHz	0Ch	308 kHz
16 MHz	4 MHz	8 MHz	27h	100 kHz
4 MHz	1 MHz	2 MHz	02h	333 kHz(1)
4 MHz	1 MHz	2 MHz	09h	100 kHz
16 MHz(2)	4 MHz	8 MHz	03h	1 MHz(1)

Note 1: The I^{2}C interface does not conform to the 400 kHz I^{2}C specification (which applies to rates greater than 100 kHz) in all details, but may be used with care where higher rates are required by the application.

2: A minimum 16-MHz FOSC is required for 1 MHz I^{2}C.

21.4.12 ACKNOWLEDGE SEQUENCE TIMING

An Acknowledge sequence is enabled by setting the Acknowledge Sequence Enable bit, ACKEN (SSPCON2<4>). When this bit is set, the SCL pin is pulled low and the contents of the Acknowledge data bit are presented on the SDA pin. If the user wishes to generate an Acknowledge, then the ACKDT bit should be cleared. If not, the user should set the ACKDT bit before starting an Acknowledge sequence. The Baud Rate Generator then counts for one rollover period (TBRG) and the SCL pin is deasserted (pulled high). When the SCL pin is sampled high (clock arbitration), the Baud Rate Generator counts for TBRG; the SCL pin is then pulled low. Following this, the ACKEN bit is automatically cleared, the Baud Rate Generator is turned off and the MSSP module then goes into an inactive state (Figure 21-25).

21.4.12.1 WCOL Status Flag

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

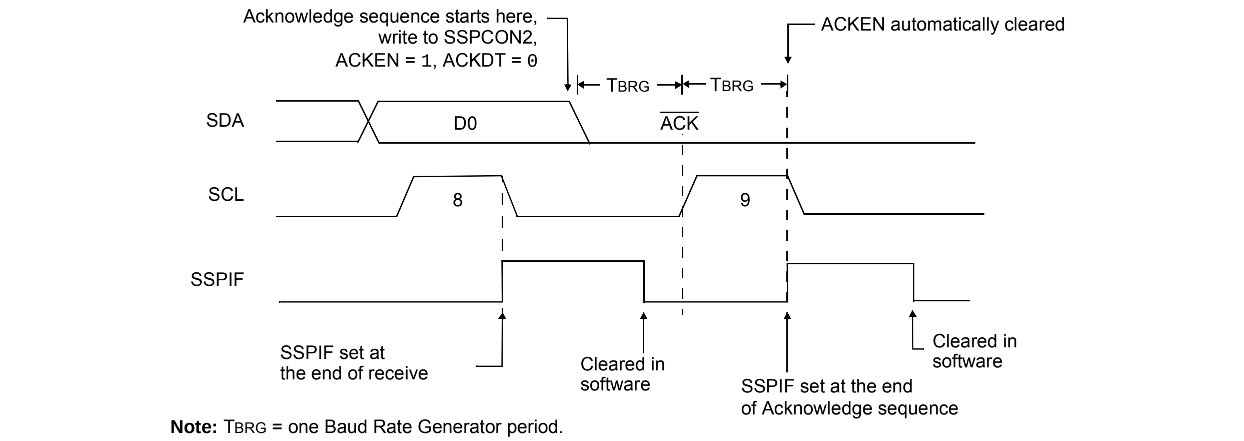
21.4.13 STOP CONDITION TIMING

A Stop bit is asserted on the SDA pin at the end of a receive/transmit by setting the Stop Sequence Enable bit, PEN (SSPCON2<2>). At the end of a receive/transmit, the SCL line is held low after the falling edge of the ninth clock. When the PEN bit is set, the master will assert the SDA line low. When the SDA line is sampled low, the Baud Rate Generator is reloaded and counts down to 0. When the Baud Rate Generator times out, the SCL pin will be brought high and one TBRG (Baud Rate Generator rollover count) later, the SDA pin will be deasserted. When the SDA pin is sampled high while SCL is high, the P bit (SSPSTAT<4>) is set. A TBRG later, the PEN bit is cleared and the SSPIF bit is set (Figure 21-26).

21.4.13.1 WCOL Status Flag

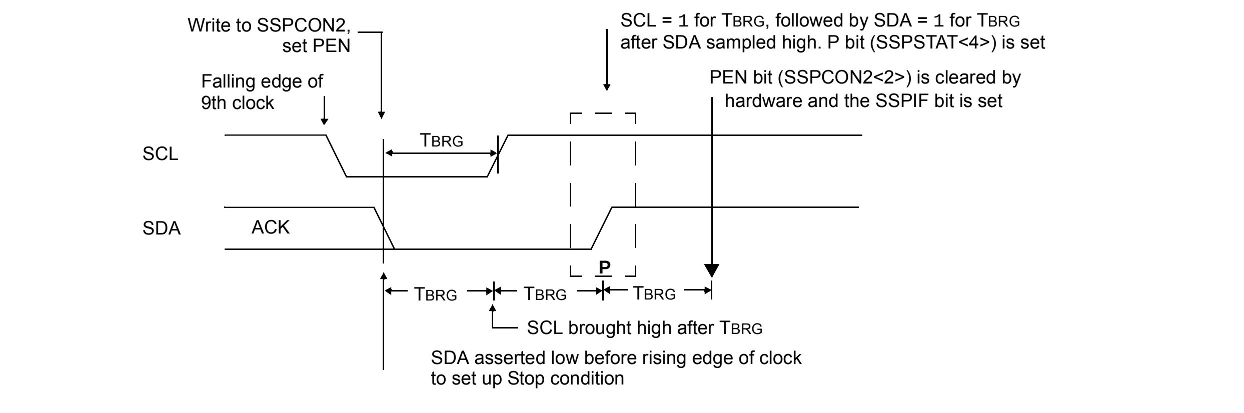
If the user writes the SSPBUF when a Stop sequence is in progress, then the WCOL bit is set and the contents of the buffer are unchanged (the write doesn't occur).

FIGURE 21-25: ACKNOWLEDGE SEQUENCE WAVEFORM



Note: TBRG = one Baud Rate Generator period.

FIGURE 21-26: STOP CONDITION RECEIVE OR TRANSMIT MODE


Write to SSPCON2,
set PEN
Falling edge of
9th clock
SCL = 1 for TBRG, followed by SDA = 1 for TBRG
after SDA sampled high. P bit (SSPSTAT<4>) is set
PEN bit (SSPCON2<2>) is cleared by
hardware and the SSPIF bit is set
TBRG
SCL
SDA
ACK
P
TBRG
TBRG
TBRG
SCL brought high after TBRG
SDA asserted low before rising edge of clock
to set up Stop condition

Note: TBRG = one Baud Rate Generator period.

22.5 EUSARTx Synchronous Slave Mode

Synchronous Slave mode is entered by clearing bit, CSRC (TXSTAx<7>). This mode differs from the Synchronous Master mode in that the shift clock is supplied externally at the CKx pin (instead of being supplied internally in Master mode). This allows the device to transfer or receive data while in any low-power mode.

22.5.1 EUSARTx SYNCHRONOUS SLAVE TRANSMISSION

The operation of the Synchronous Master and Slave modes is identical, except in the case of Sleep mode.

If two words are written to the TXREGx and then the SLEEP instruction is executed, the following will occur:

a) The first word will immediately transfer to the TSR register and transmit.

b) The second word will remain in the TXREGx register.

c) Flag bit, TXxIF, will not be set.

d) When the first word has been shifted out of TSR, the TXREGx register will transfer the second word to the TSR and flag bit, TXxIF, will now be set.

e) If enable bit, TXxIE, is set, the interrupt will wake the chip from Sleep. If the global interrupt is enabled, the program will branch to the interrupt vector.

To set up a Synchronous Slave Transmission:

1. Enable the synchronous slave serial port by setting bits, SYNC and SPEN, and clearing bit, CSRC.
2. Clear bits, CREN and SREN.
3. If interrupts are desired, set enable bit, TXxIE.
4. If 9-bit transmission is desired, set bit, TX9.
5. Enable the transmission by setting enable bit, TXEN.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit, TX9D.
7. Start transmission by loading data to the TXREGx register.
8. If using interrupts, ensure that the GIE and PEIE bits (INTCON<7:6>) are set.

TABLE 22-10: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE TRANSMISSION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	TMR1GIF	TMR2IF	TMR1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	TMR1GIE	TMR2IE	TMR1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	TMR1GIP	TMR2IP	TMR1IP
PIR3	—	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	—
PIE3	—	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	—
IPR3	—	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	—
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
TXREG1	EUSART1 Transmit Register							
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
BAUDCON1	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN
SPBRGH1	EUSART1 Baud Rate Generator Register High Byte							
SPBRG1	EUSART1 Baud Rate Generator Register Low Byte							
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
TXREG2	EUSART2 Transmit Register							
TXSTA2	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
BAUDCON2	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN
SPBRGH2	EUSART2 Baud Rate Generator Register High Byte							
SPBRG2	EUSART2 Baud Rate Generator Register Low Byte							
PMD0	CCP5MD	CCP4MD	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSPMD
ODCON	SSPOD	CCP5OD	CCP4OD	CCP3OD	CCP2OD	CCP1OD	U2OD	U1OD

Legend: — = unimplemented, read as ‘0’. Shaded cells are not used for synchronous slave transmission.

23.2 A/D Registers

23.2.1 A/D CONTROL REGISTERS

REGISTER 23-1: ADCON0: A/D CONTROL REGISTER 0

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	CHS4	CHS3	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	ADON
bit 7							bit 0

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

bit 7 **Unimplemented:** Read as ‘0’

bit 6-2 **CHS<4:0>:** Analog Channel Select bits

00000 = Channel 00 (AN0)
00001 = Channel 01 (AN1)
00010 = Channel 02 (AN2)
00011 = Channel 03 (AN3)
00100 = Channel 04 (AN4)
00101 = Channel 05 (AN5)[1,2]
00110 = Channel 06 (AN6)[1,2]
00111 = Channel 07 (AN7)[1,2]
01000 = Channel 08 (AN8)
01001 = Channel 09 (AN9)
01010 = Channel 10 (AN10)
01011 = (Reserved)[2]
01100 = (Reserved)[2)]
01101 = (Reserved)[2)]
01110 = (Reserved)[2)]
01111 = (Reserved)[2]

10000 = (Reserved)[2]
10001 = (Reserved)[2]
10010 = (Reserved)[2]
10011 = (Reserved)[2]
10100 = (Reserved)[2]
10101 = (Reserved)[2]
10110 = (Reserved)[2]
10111 = (Reserved)[2]
11000 = (Reserved)[2]
11001 = (Reserved)[2]
11010 = (Reserved)[2]
11011 = (Reserved)[2]
11100 = (MUX disconnect)[3]
11101 = Channel 29 (temperature diode)
11110 = Channel 30 (VDDCORE)
11111 = Channel 31 (1.024V band gap)

bit 1 **GO/$\overline{\text{DONE}}$:** A/D Conversion Status bit

1 = A/D cycle is in progress. Setting this bit starts an A/D conversion cycle. The bit is cleared automatically by hardware when the A/D conversion is completed.
0 = A/D conversion has completed or is not in progress

bit 0 **ADON:** A/D On bit

1 = A/D Converter is operating
0 = A/D conversion module is shut off and consuming no operating current

Note 1: These channels are not implemented on 28-pin devices.

2: Performing a conversion on unimplemented channels will return random values.

3: Channel 28 turns off analog MUX switches to allow for minimum capacitive loading of the A/D input, for finer resolution CTMU time measurements.

REGISTER 23-3: ADCON2: A/D CONTROL REGISTER 2

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0
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 **ADFM:** A/D Result Format Select bit

1 = Right justified
0 = Left justified

bit 6 **Unimplemented:** Read as ‘0’

bit 5-3 **ACQT<2:0>:** A/D Acquisition Time Select bits

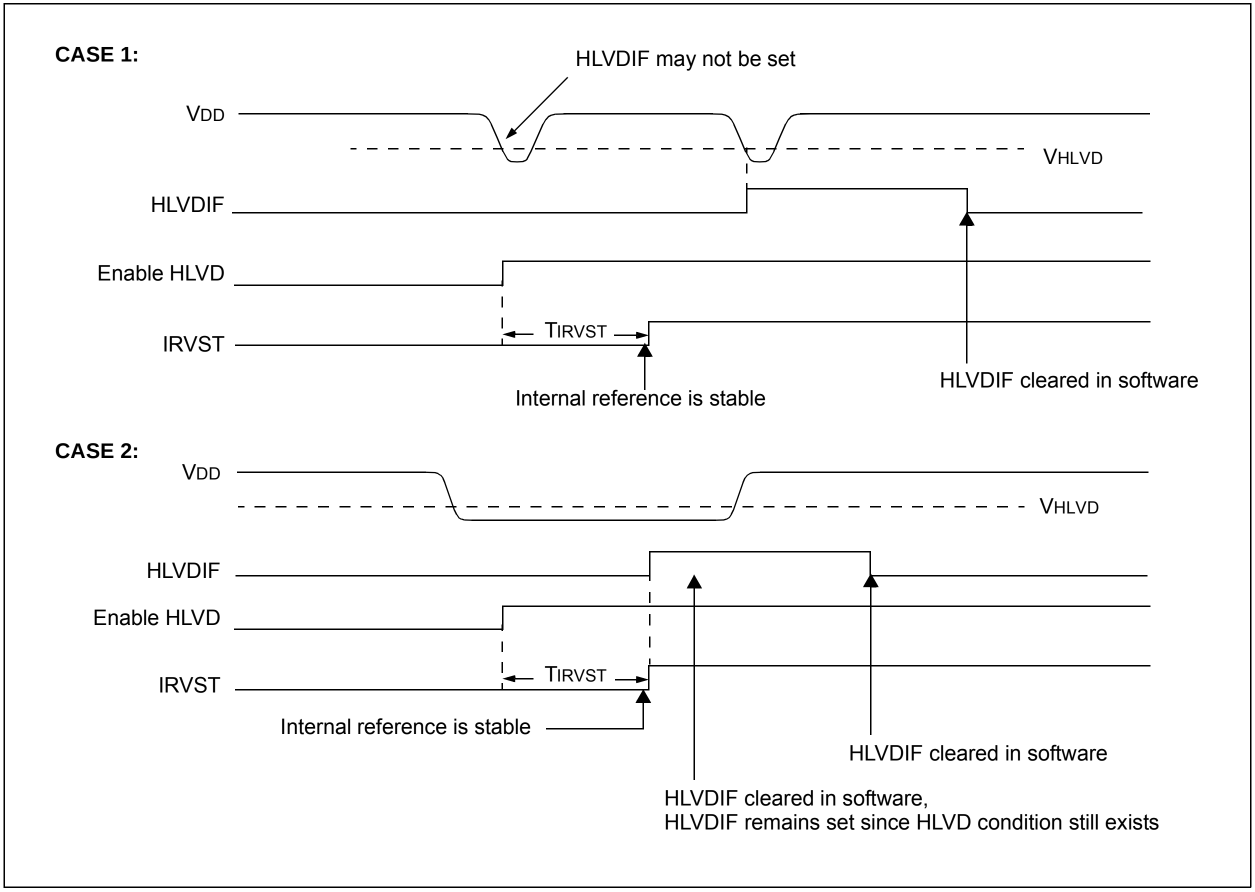
111 = 20 TAD
110 = 16 TAD
101 = 12 TAD
100 = 8 TAD
011 = 6 TAD
010 = 4 TAD
001 = 2 TAD
000 = 0 TAD(1)

bit 2-0 **ADCS<2:0>:** A/D Conversion Clock Select bits

111 = FRC (clock derived from A/D RC oscillator)(1)
110 = FOSC/64
101 = FOSC/16
100 = FOSC/4
011 = FRC (clock derived from A/D RC oscillator)(1)
010 = FOSC/32
001 = FOSC/8
000 = FOSC/2

Note 1: If the A/D FRC clock source is selected, a delay of one TCY (instruction cycle) is added before the A/D clock starts. This allows the SLEEP instruction to be executed before starting a conversion.

FIGURE 26-2: LOW-VOLTAGE DETECT OPERATION (VDIRMAG = 0)


CASE 1:
HLVDIF may not be set
VDD
VHLVD
HLVDIF
Enable HLVD
TIRVST
IRVST
HLVDIF cleared in software
Internal reference is stable
CASE 2:
VDD
VHLVD
HLVDIF
Enable HLVD
TIRVST
IRVST
Internal reference is stable
HLVDIF cleared in software
HLVDIF cleared in software,
HLVDIF remains set since HLVD condition still exists

REGISTER 27-14: RXB1CON: RECEIVE BUFFER 1 CONTROL REGISTER

Mode 0	R/C-0	R/W-0	R/W-0	U-0	R-0	R/W-0	R-0	R-0
	RXFUL[1]	RXM1	RXM0	—	RXRTRRO	FILHIT2	FILHIT1	FILHIT0

Mode 1,2	R/C-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
	RXFUL[1]	RXM1	RTRRO	FILHIT4	FILHIT3	FILHIT2	FILHIT1	FILHIT0
	bit 7							bit 0

Legend:	C = Clearable bit		
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 **RXFUL:** Receive Full Status bit[1]

1 = Receive buffer contains a received message
0 = Receive buffer is open to receive a new message

bit 6-5, 6 Mode 0:
RXM<1:0>: Receive Buffer Mode bit 1 (combines with RXM0 to form RXM<1:0> bits, see bit 5)

11 = Receive all messages (including those with errors); filter criteria is ignored
10 = Receive only valid messages with extended identifier; EXIDEN in RXFnSIDL must be ‘1’
01 = Receive only valid messages with standard identifier, EXIDEN in RXFnSIDL must be ‘0’
00 = Receive all valid messages as per EXIDEN bit in RXFnSIDL register

Mode 1, 2:
RXM1: Receive Buffer Mode bit

1 = Receive all messages (including those with errors); acceptance filters are ignored
0 = Receive all valid messages as per acceptance filters

bit 5 Mode 0:
RXM<1:0>: Receive Buffer Mode bit 0 (combines with RXM1 to form RXM<1:0> bits, see bit 6)

Mode 1, 2:
RTRRO: Remote Transmission Request bit for Received Message (read-only)

1 = A remote transmission request is received
0 = A remote transmission request is not received

bit 4 Mode 0:
FILHIT2**4:** Filter Hit bit 4

Mode 1, 2:
FILHIT<4:0>: Filter Hit bit 4
This bit combines with other bits to form the filter acceptance bits<4:0>.

bit 3 Mode 0:
RXRTRRO: Remote Transmission Request bit for Received Message (read-only)

1 = A remote transmission request is received
0 = A remote transmission request is not received

Mode 1, 2:
FILHIT<4:0>: Filter Hit bit 3
This bit combines with other bits to form the filter acceptance bits<4:0>.

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 27-28: BnEIDH: TX/RX BUFFER ‘n’ EXTENDED IDENTIFIER REGISTERS, HIGH BYTE IN RECEIVE MODE [0 ≤ n ≤ 5, TXnEN (BSEL0<n>) = 0][1]

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8
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 **EID<15:8>:** Extended Identifier bits

Note 1: These registers are available in Mode 1 and 2 only.

REGISTER 27-29: BnEIDH: TX/RX BUFFER ‘n’ EXTENDED IDENTIFIER REGISTERS, HIGH BYTE IN TRANSMIT MODE [0 ≤ n ≤ 5, TXnEN (BSEL0<n>) = 1][1]

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8
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 **EID<15:8>:** Extended Identifier bits

Note 1: These registers are available in Mode 1 and 2 only.

REGISTER 27-30: BnEIDL: TX/RX BUFFER ‘n’ EXTENDED IDENTIFIER REGISTERS, LOW BYTE IN RECEIVE MODE [0 ≤ n ≤ 5, TXnEN (BSEL<n>) = 0][1]

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0
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 **EID<7:0>:** Extended Identifier bits

Note 1: These registers are available in Mode 1 and 2 only.

REGISTER 27-48: MSEL0: MASK SELECT REGISTER 0[1]

R/W-0	R/W-1	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0
FIL3_1	FIL3_0	FIL2_1	FIL2_0	FIL1_1	FIL1_0	FIL0_1	FIL0_0
bit 7							bit 0

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

bit 7-6 **FIL3_<1:0>:** Filter 3 Select bits 1 and 0

11 = No mask
10 = Filter 15
01 = Acceptance Mask 1
00 = Acceptance Mask 0

bit 5-4 **FIL2_<1:0>:** Filter 2 Select bits 1 and 0

11 = No mask
10 = Filter 15
01 = Acceptance Mask 1
00 = Acceptance Mask 0

bit 3-2 **FIL1_<1:0>:** Filter 1 Select bits 1 and 0

11 = No mask
10 = Filter 15
01 = Acceptance Mask 1
00 = Acceptance Mask 0

bit 1-0 **FIL0_<1:0>:** Filter 0 Select bits 1 and 0

11 = No mask
10 = Filter 15
01 = Acceptance Mask 1
00 = Acceptance Mask 0

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

BZ Branch if Zero

Syntax:	BZ n
Operands:	$-128 \le n \le 127$
Operation:	if Zero bit is '1', (PC) + 2 + 2n → PC
Status Affected:	None

Encoding:	1110	0000	nnnn	nnnn

Description: If the Zero bit is '1', then the program will branch.

The 2's complement number, '2n', is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.

Words:	1
Cycles:	1(2)

Q Cycle Activity:
If Jump:

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

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example:
```
Example:          HERE        BZ   Jump

    Before Instruction
        PC           =    address (HERE)
    After Instruction
        If Zero      =    1;
            PC       =    address (Jump)
        If Zero      =    0;
            PC       =    address (HERE + 2)
```

CALL Subroutine Call

Syntax:	CALL k {,s}
Operands:	$0 \le k \le 1048575$ $s \in [0,1]$
Operation:	(PC) + 4 → TOS, k → PC<20:1>; if s = 1 (W) → WS, (STATUS) → STATUSS, (BSR) → BSRS
Status Affected:	None

Encoding:				
1st word (k<7:0>)	1110	110s	$k_7$kkk	kkkk$_0$
2nd word(k<19:8>)	1111	k_{19}kkk	kkkk	kkkk$_8$

Description: Subroutine call of entire 2-Mbyte memory range. First, return address (PC+ 4) is pushed onto the return stack. If 's' = 1, the W, STATUS and BSR registers are also pushed into their respective shadow registers, WS, STATUSS and BSRS. If 's' = 0, no update occurs (default). Then, the 20-bit value 'k' is loaded into PC<20:1>. CALL is a two-cycle instruction.

Words:	2
Cycles:	2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'<7:0>,	Push PC to stack	Read literal 'k'<19:8>, Write to PC
No operation	No operation	No operation	No operation

```
Example:          HERE       CALL   THERE,1

    Before Instruction
        PC        =    address (HERE)
    After Instruction
        PC        =    address (THERE)
        TOS       =    address (HERE + 4)
        WS        =    W
        BSRS      =    BSR
        STATUSS=       STATUS
```

LFSR **Load FSR**

Syntax: LFSR f, k

Operands: $0 \leq f \leq 2$
$0 \leq k \leq 4095$

Operation: k → FSRf

Status Affected: None

Encoding:

1110	1110	00ff	k_{11}kkk
1111	0000	$k_7$kkk	kkkk

Description: The 12-bit literal ‘k’ is loaded into the file select register pointed to by ‘f’.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal ‘k’ MSB	Process Data	Write literal ‘k’ MSB to FSRfH
Decode	Read literal ‘k’ LSB	Process Data	Write literal ‘k’ to FSRfL

Example: `LFSR 2, 3ABh`

After Instruction

FSR2H	=	03h
FSR2L	=	ABh

MOVF **Move f**

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

Operands: $0 \leq f \leq 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: f → dest

Status Affected: N, Z

Encoding:

0101	00da	ffff	ffff

Description: The contents of register ‘f’ are moved to a destination dependent upon the status of ‘d’. If ‘d’ is ‘0’, the result is placed in W. If ‘d’ is ‘1’, the result is placed back in register ‘f’ (default). Location ‘f’ can be anywhere in the 256-byte bank.

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

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	Write W

Example: `MOVF REG, 0, 0`

Before Instruction

REG	=	22h
W	=	FFh

After Instruction

REG	=	22h
W	=	22h

TSTFSZ — Test f, Skip if 0

Syntax: TSTFSZ f {,a}

Operands: 0 ≤ f ≤ 255
a ∈ [0,1]

Operation: skip if f = 0

Status Affected: None

Encoding:

0110	011a	ffff	ffff

Description: If ‘f’ = 0, the next instruction fetched during the current instruction execution is discarded and a NOP is executed, 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 ≤ 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    TSTFSZ  CNT, 1
NZERO   :
ZERO    :
```

Before Instruction
PC = Address (HERE)
After Instruction
If CNT = 00h,
PC = Address (ZERO)
If CNT ≠ 00h,
PC = Address (NZERO)

XORLW — Exclusive OR Literal with W

Syntax: XORLW k

Operands: 0 ≤ k ≤ 255

Operation: (W) .XOR. k → W

Status Affected: N, Z

Encoding:

0000	1010	kkkk	kkkk

Description: The contents of W are XORed with the 8-bit literal ‘k’. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal ‘k’	Process Data	Write to W

Example:

```
XORLW   0AFh
```

Before Instruction
W = B5h
After Instruction
W = 1Ah

APPENDIX B: MIGRATION TO PIC18F66K80 FAMILY

Devices in the PIC18F66K80, PIC18F4580, PIC18F4680 and 18F8680 families are similar in their functions and features. Code can be migrated from the other families to the PIC18F66K80 without many changes. The differences between the device families are listed in Table B-1 and Table B-2. For more details on migrating to the PIC18F66K80, refer to *"PIC18FXX80 to PIC18FXXK80 Migration Guide"* (DS39982).

TABLE B-1: NOTABLE DIFFERENCES BETWEEN 28, 40 AND 44-PIN DEVICES – PIC18F66K80, PIC18F4580 AND PIC18F4680 FAMILIES

Characteristic	PIC18F66K80 Family	PIC18F4680 Family	PIC18F4580 Family
Max Operating Frequency	64 MHz	40 MHz	40 MHz
Max Program Memory	64 Kbytes	64 Kbytes	32 Kbytes
Data Memory (bytes)	3,648	3,328	1,536
CTMU	Yes	No	No
SOSC Oscillator Options	Low-power oscillator option for SOSC	No options	No options
T1CKI Clock	T1CKI can be used as a clock without enabling the SOSC oscillator	No	No
INTOSC	Up to 16 MHz	Up to 8 MHz	Up to 8 MHz
Timers	Two 8-bit, three 16-bit	One 8-bit, three 16-bit	One 8-bit, three 16-bit
ECCP	One for all devices	40 and 44-pin devices – One 28-pin devices – None	40 and 44-pin devices – One 28-pin devices – None
CCP	Four	One	One
Data EEPROM (bytes)	1,024	1,024	256
WDT Prescale Options	22	16	16
5V Operation	18FXXK80 parts – 5V operation 18LFXXK80 parts – 3.3V operation	Yes	Yes
nanoWatt XLP	Yes	No	No
Regulator	18FXXK80 parts – Yes 18LFXXK80 parts – No	No	No
Low-Power BOR	Yes	No	No
A/D Converter	12-bit signed differential	10-bit	10-bit
A/D Channels	28-pin devices – 8 Channels 40 and 44-pin devices – 11 Channels	8 Channels for 28-pin devices/ 11 Channels for 40 and 44-pin devices	8 Channels for 28-pin devices/ 11 Channels for 40 and 44-pin devices
Internal Temp Sensor	Yes	No	No
EUSART	Two	One	One
Comparators	Two	28-pin devices – None 40 and 44-pin devices – Two	28-pin devices – None 40 and 44-pin devices – Two
Oscillator Options	14	Nine	Nine
Ultra Low-Power Wake-up (ULPW)	Yes	No	No
Adjustable Slew Rate for I/O	Yes	No	No
PLL	Available for all oscillator options	Available only for high-speed crystal and internal oscillator	Available only for high-speed crystal and internal oscillator
TXM Modulator	No	No	No