

Welcome to [E-XFL.COM](https://www.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	20MHz
Connectivity	-
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	12
Program Memory Size	3.5KB (2K x 14)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	128 x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	14-SOIC (0.154", 3.90mm Width)
Supplier Device Package	14-SOIC
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16f684-i-sl

PIC16F684

2.2.1 GENERAL PURPOSE REGISTER FILE

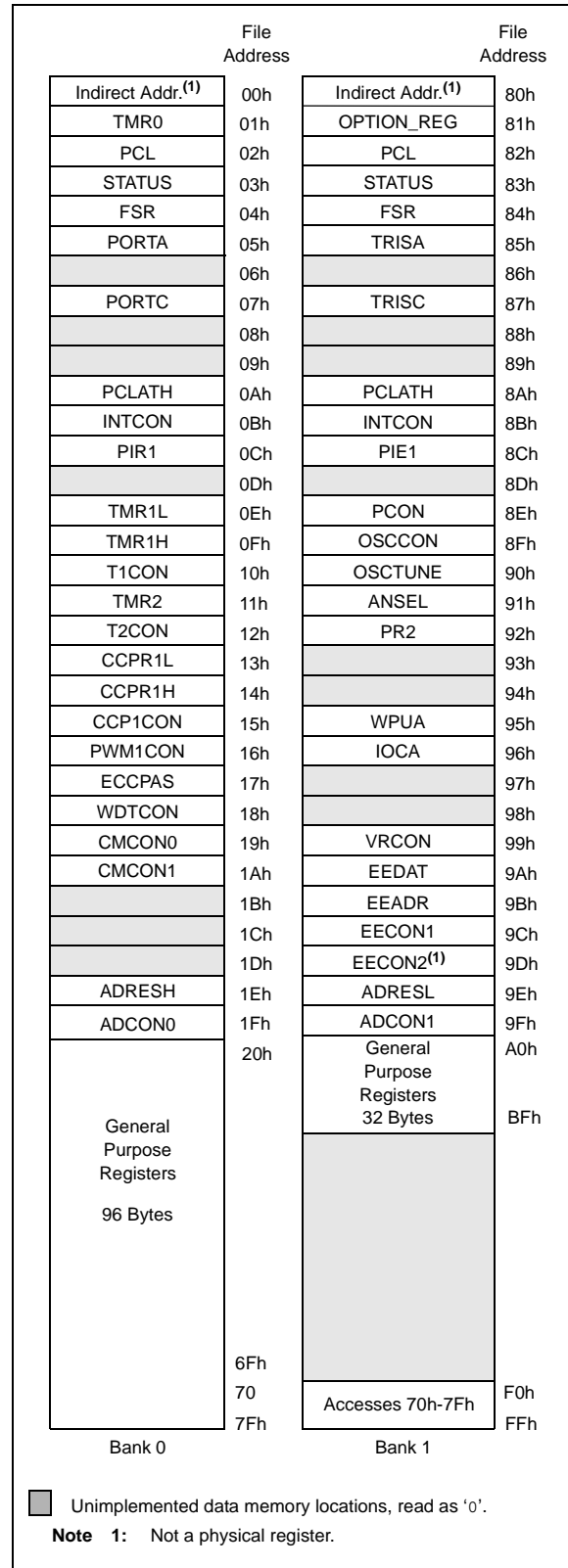
The register file is organized as 128 x 8 in the PIC16F684. Each register is accessed, either directly or indirectly, through the File Select Register (FSR) (see **Section 2.4 “Indirect Addressing, INDF and FSR Registers”**).

2.2.2 SPECIAL FUNCTION REGISTERS

The Special Function Registers are registers used by the CPU and peripheral functions for controlling the desired operation of the device (see Table 2-1). These registers are static RAM.

The special registers can be classified into two sets: core and peripheral. The Special Function Registers associated with the “core” are described in this section. Those related to the operation of the peripheral features are described in the section of that peripheral feature.

FIGURE 2-2: DATA MEMORY MAP OF THE PIC16F684



PIC16F684

TABLE 2-2: PIC16F684 SPECIAL FUNCTION REGISTERS SUMMARY BANK 1

Addr	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Page
Bank 1											
80h	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								xxxx xxxx	19, 104
81h	OPTION_REG	RAPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	14, 104
82h	PCL	Program Counter's (PC) Least Significant Byte								0000 0000	19, 104
83h	STATUS	IRP ⁽¹⁾	RP1 ⁽¹⁾	RP0	\overline{TO}	\overline{PD}	Z	DC	C	0001 1xxx	13, 104
84h	FSR	Indirect Data Memory Address Pointer								xxxx xxxx	19, 104
85h	TRISA	—	—	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	--11 1111	31, 104
86h	—	Unimplemented								—	—
87h	TRISC	—	—	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	--11 1111	40, 104
88h	—	Unimplemented								—	—
89h	—	Unimplemented								—	—
8Ah	PCLATH	—	—	—	Write Buffer for upper 5 bits of Program Counter				---0 0000	19, 104	
8Bh	INTCON	GIE	PEIE	T0IE	INTE	RAIE	T0IF	INTF	RAIF	0000 0000	15, 104
8Ch	PIE1	EEIE	ADIE	CCP1IE	C2IE	C1IE	OSFIE	TMR2IE	TMR1IE	0000 0000	16, 104
8Dh	—	Unimplemented								—	—
8Eh	PCON	—	—	ULPWUE	SBOREN	—	—	\overline{POR}	\overline{BOR}	--01 --qq	18, 104
8Fh	OSCCON	—	IRCF2	IRCF1	IRCF0	OSTS ⁽²⁾	HTS	LTS	SCS	-110 x000	20, 104
90h	OSCTUNE	—	—	—	TUN4	TUN3	TUN2	TUN1	TUN0	---0 0000	24, 105
91h	ANSEL	ANS7	ANS6	ANS5	ANS4	ANS3	ANS2	ANS1	ANS0	1111 1111	32, 105
92h	PR2	Timer2 Module Period Register								1111 1111	53, 105
93h	—	Unimplemented								—	—
94h	—	Unimplemented								—	—
95h	WPUA ⁽³⁾	—	—	WPUA5	WPUA4	—	WPUA2	WPUA1	WPUA0	--11 -111	33, 105
96h	IOCA	—	—	IOCA5	IOCA4	IOCA3	IOCA2	IOCA1	IOCA0	--00 0000	33, 105
97h	—	Unimplemented								—	—
98h	—	Unimplemented								—	—
99h	VRCON	VREN	—	VRR	—	VR3	VR2	VR1	VR0	0-0- 0000	63, 105
9Ah	EEDAT	EEDAT7	EEDAT6	EEDAT5	EEDAT4	EEDAT3	EEDAT2	EEDAT1	EEDAT0	0000 0000	75, 105
9Bh	EEADR	EEADR7	EEADR6	EEADR5	EEADR4	EEADR3	EEADR2	EEADR1	EEADR0	0000 0000	75, 105
9Ch	EECON1	—	—	—	—	WRERR	WREN	WR	RD	---- x000	76, 105
9Dh	EECON2	EEPROM Control Register 2 (not a physical register)								---- ----	76, 105
9Eh	ADRESL	Least Significant 2 bits of the left shifted result or 8 bits of the right shifted result								xxxx xxxx	71, 105
9Fh	ADCON1	—	ADCS2	ADCS1	ADCS0	—	—	—	—	-000 ----	70, 105

Legend: — = Unimplemented locations read as '0', u = unchanged, x = unknown, q = value depends on condition, shaded = unimplemented

Note 1: IRP and RP1 bits are reserved, always maintain these bits clear.

Note 2: OSTS bit of the OSCCON register reset to '0' with Dual Speed Start-up and LP, HS or XT selected as the oscillator.

Note 3: RA3 pull-up is enabled when MCLRE is '1' in the Configuration Word register.

2.2.2.5 PIR1 Register

The PIR1 register contains the peripheral interrupt flag bits, as shown in Register 2-5.

Note: Interrupt flag bits are set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the global enable bit, GIE of the INTCON register. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

REGISTER 2-5: PIR1: PERIPHERAL INTERRUPT REQUEST REGISTER 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
EEIF	ADIF	CCP1IF	C2IF	C1IF	OSFIF	TMR2IF	TMR1IF
bit 7							bit 0

Legend:

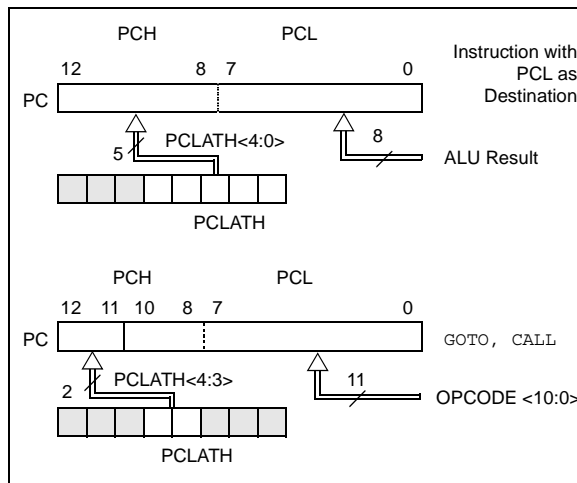
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared x = Bit is unknown

- bit 7 **EEIF:** EEPROM Write Operation Interrupt Flag bit
 1 = The write operation completed (must be cleared in software)
 0 = The write operation has not completed or has not been started
- bit 6 **ADIF:** A/D Interrupt Flag bit
 1 = A/D conversion complete
 0 = A/D conversion has not completed or has not been started
- bit 5 **CCP1IF:** CCP1 Interrupt Flag bit
Capture mode:
 1 = A TMR1 register capture occurred (must be cleared in software)
 0 = No TMR1 register capture occurred
Compare mode:
 1 = A TMR1 register compare match occurred (must be cleared in software)
 0 = No TMR1 register compare match occurred
PWM mode:
 Unused in this mode
- bit 4 **C2IF:** Comparator 2 Interrupt Flag bit
 1 = Comparator 2 output has changed (must be cleared in software)
 0 = Comparator 2 output has not changed
- bit 3 **C1IF:** Comparator 1 Interrupt Flag bit
 1 = Comparator 1 output has changed (must be cleared in software)
 0 = Comparator 1 output has not changed
- bit 2 **OSFIF:** Oscillator Fail Interrupt Flag bit
 1 = System oscillator failed, clock input has changed to INTOSC (must be cleared in software)
 0 = System clock operating
- bit 1 **TMR2IF:** Timer2 to PR2 Match Interrupt Flag bit
 1 = Timer2 to PR2 match occurred (must be cleared in software)
 0 = Timer2 to PR2 match has not occurred
- bit 0 **TMR1IF:** Timer1 Overflow Interrupt Flag bit
 1 = Timer1 register overflowed (must be cleared in software)
 0 = Timer1 has not overflowed

2.3 PCL and PCLATH

The Program Counter (PC) is 13 bits wide. The low byte comes from the PCL register, which is a readable and writable register. The high byte (PC<12:8>) is not directly readable or writable and comes from PCLATH. On any Reset, the PC is cleared. Figure 2-3 shows the two situations for the loading of the PC. The upper example in Figure 2-3 shows how the PC is loaded on a write to PCL (PCLATH<4:0> → PCH). The lower example in Figure 2-3 shows how the PC is loaded during a CALL or GOTO instruction (PCLATH<4:3> → PCH).

FIGURE 2-3: LOADING OF PC IN DIFFERENT SITUATIONS



2.3.1 MODIFYING PCL

Executing any instruction with the PCL register as the destination simultaneously causes the Program Counter PC<12:8> bits (PCH) to be replaced by the contents of the PCLATH register. This allows the entire contents of the program counter to be changed by first writing the desired upper 5 bits to the PCLATH register. Then, when the lower 8 bits are written to the PCL register, all 13 bits of the program counter will change to the values contained in the PCLATH register and those being written to the PCL register.

A computed GOTO is accomplished by adding an offset to the program counter (ADDWF PCL). Care should be exercised when jumping into a look-up table or program branch table (computed GOTO) by modifying the PCL register. Assuming that PCLATH is set to the table start address, if the table length is greater than 255 instructions or if the lower 8 bits of the memory address rolls over from 0xFF to 0x00 in the middle of the table, then PCLATH must be incremented for each address rollover that occurs between the table beginning and the target location within the table.

For more information refer to Application Note AN556, "Implementing a Table Read" (DS00556).

2.3.2 STACK

The PIC16F684 Family has an 8-level x 13-bit wide hardware stack (see Figure 2-1). The stack space is not part of either program or data space and the Stack Pointer is not readable or writable. The PC is PUSHed onto the stack when a CALL instruction is executed or an interrupt causes a branch. The stack is POPed in the event of a RETURN, RETLW or a RETFIE instruction execution. PCLATH is not affected by a PUSH or POP operation.

The stack operates as a circular buffer. This means that after the stack has been PUSHed eight times, the ninth push overwrites the value that was stored from the first push. The tenth push overwrites the second push (and so on).

- Note 1:** There are no Status bits to indicate stack overflow or stack underflow conditions.
- 2:** There are no instructions/mnemonics called PUSH or POP. These are actions that occur from the execution of the CALL, RETURN, RETLW and RETFIE instructions or the vectoring to an interrupt address.

2.4 Indirect Addressing, INDF and FSR Registers

The INDF register is not a physical register. Addressing the INDF register will cause indirect addressing.

Indirect addressing is possible by using the INDF register. Any instruction using the INDF register actually accesses data pointed to by the File Select Register (FSR). Reading INDF itself indirectly will produce 00h. Writing to the INDF register indirectly results in a no operation (although Status bits may be affected). An effective 9-bit address is obtained by concatenating the 8-bit FSR and the IRP bit of the STATUS register, as shown in Figure 2-4.

A simple program to clear RAM location 20h-2Fh using indirect addressing is shown in Example 2-1.

EXAMPLE 2-1: INDIRECT ADDRESSING

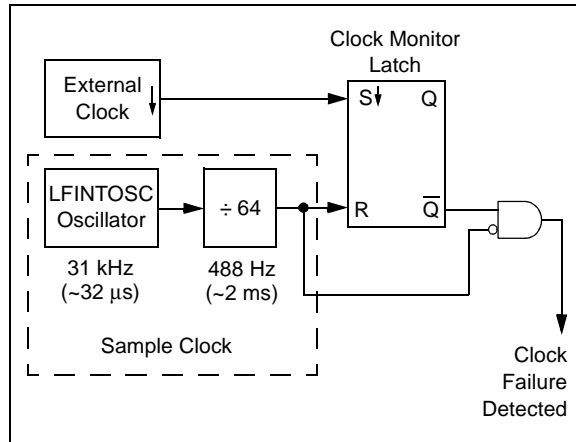
```

MOV LW 0x20 ;initialize pointer
MOV WF FSR ;to RAM
NEXT CLR F INDF ;clear INDF register
INCF FSR, f ;inc pointer
BTFSS FSR, 4 ;all done?
GOTO NEXT ;no clear next
CONTINUE ;yes continue
    
```

3.8 Fail-Safe Clock Monitor

The Fail-Safe Clock Monitor (FSCM) allows the device to continue operating should the external oscillator fail. The FSCM can detect oscillator failure any time after the Oscillator Start-up Timer (OST) has expired. The FSCM is enabled by setting the FCMEN bit in the Configuration Word register (CONFIG). The FSCM is applicable to all external oscillator modes (LP, XT, HS, EC, RC and RCIO).

FIGURE 3-8: FSCM BLOCK DIAGRAM



3.8.1 FAIL-SAFE DETECTION

The FSCM module detects a failed oscillator by comparing the external oscillator to the FSCM sample clock. The sample clock is generated by dividing the LFINTOSC by 64. See Figure 3-8. Inside the fail detector block is a latch. The external clock sets the latch on each falling edge of the external clock. The sample clock clears the latch on each rising edge of the sample clock. A failure is detected when an entire half-cycle of the sample clock elapses before the primary clock goes low.

3.8.2 FAIL-SAFE OPERATION

When the external clock fails, the FSCM switches the device clock to an internal clock source and sets the bit flag OSFIF of the PIR1 register. Setting this flag will generate an interrupt if the OSFIE bit of the PIE1 register is also set. The device firmware can then take steps to mitigate the problems that may arise from a failed clock. The system clock will continue to be sourced from the internal clock source until the device firmware successfully restarts the external oscillator and switches back to external operation.

The internal clock source chosen by the FSCM is determined by the IRCF<2:0> bits of the OSCCON register. This allows the internal oscillator to be configured before a failure occurs.

3.8.3 FAIL-SAFE CONDITION CLEARING

The Fail-Safe condition is cleared after a Reset, executing a SLEEP instruction or toggling the SCS bit of the OSCCON register. When the SCS bit is toggled, the OST is restarted. While the OST is running, the device continues to operate from the INTOSC selected in OSCCON. When the OST times out, the Fail-Safe condition is cleared and the device will be operating from the external clock source. The Fail-Safe condition must be cleared before the OSFIF flag can be cleared.

3.8.4 RESET OR WAKE-UP FROM SLEEP

The FSCM is designed to detect an oscillator failure after the Oscillator Start-up Timer (OST) has expired. The OST is used after waking up from Sleep and after any type of Reset. The OST is not used with the EC or RC Clock modes so that the FSCM will be active as soon as the Reset or wake-up has completed. When the FSCM is enabled, the Two-Speed Start-up is also enabled. Therefore, the device will always be executing code while the OST is operating.

Note: Due to the wide range of oscillator start-up times, the Fail-Safe circuit is not active during oscillator start-up (i.e., after exiting Reset or Sleep). After an appropriate amount of time, the user should check the OSTS bit of the OSCCON register to verify the oscillator start-up and that the system clock switchover has successfully completed.

PIC16F684

FIGURE 3-9: FSCM TIMING DIAGRAM

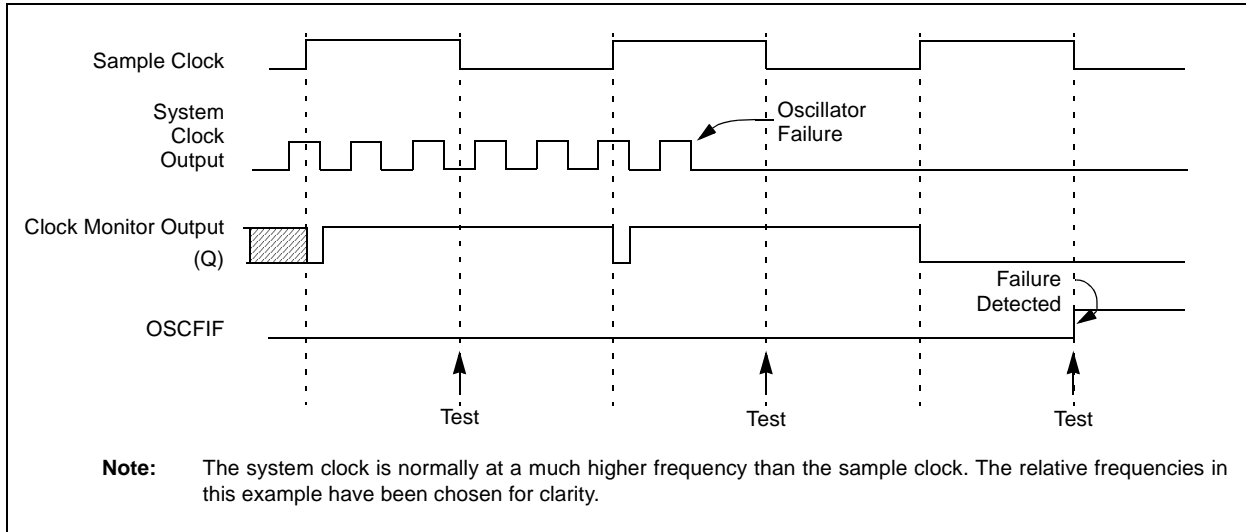


TABLE 3-2: SUMMARY OF REGISTERS ASSOCIATED WITH CLOCK SOURCES

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets ⁽¹⁾
CONFIG ⁽²⁾	CPD	CP	MCLRE	PWRTE	WDTE	FOSC2	FOSC1	FOSC0	—	—
OSCCON	—	IRCF2	IRCF1	IRCF0	OSTS	HTS	LTS	SCS	-110 x000	-110 x000
OSCTUNE	—	—	—	TUN4	TUN3	TUN2	TUN1	TUN0	---0 0000	---u uuuu
PIE1	EEIE	ADIE	CCP1IE	C2IE	C1IE	OSFIE	TMR2IE	TMR1IE	0000 0000	0000 0000
PIR1	EEIF	ADIF	CCP1IF	C2IF	C1IF	OSFIF	TMR2IF	TMR1IF	0000 0000	0000 0000

Legend: x = unknown, u = unchanged, - = unimplemented locations read as '0'. Shaded cells are not used by oscillators.

Note 1: Other (non Power-up) Resets include MCLR Reset and Watchdog Timer Reset during normal operation.

2: See Configuration Word register (Register 12-1) for operation of all register bits.

4.0 I/O PORTS

There are as many as twelve general purpose I/O pins available. Depending on which peripherals are enabled, some or all of the pins may not be available as general purpose I/O. In general, when a peripheral is enabled, the associated pin may not be used as a general purpose I/O pin.

4.1 PORTA and the TRISA Registers

PORTA is a 6-bit wide, bidirectional port. The corresponding data direction register is TRISA (Register 4-2). Setting a TRISA bit (= 1) will make the corresponding PORTA pin an input (i.e., disable the output driver). Clearing a TRISA bit (= 0) will make the corresponding PORTA pin an output (i.e., enables output driver and puts the contents of the output latch on the selected pin). The exception is RA3, which is input only and its TRIS bit will always read as '1'. Example 4-1 shows how to initialize PORTA.

Reading the PORTA register (Register 4-1) reads the status of the pins, whereas writing to it will write to the PORT latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the

port pins are read, this value is modified and then written to the PORT data latch. RA3 reads '0' when MCLR = 1.

The TRISA register controls the direction of the PORTA pins, even when they are being used as analog inputs. The user must ensure the bits in the TRISA register are maintained set when using them as analog inputs. I/O pins configured as analog input always read '0'.

Note: The ANSEL and CMCON0 registers must be initialized to configure an analog channel as a digital input. Pins configured as analog inputs will read '0'.

EXAMPLE 4-1: INITIALIZING PORTA

```
BCF STATUS,RP0 ;Bank 0
CLRF PORTA ;Init PORTA
MOVLW 07h ;Set RA<2:0> to
MOVWF CMCON0 ;digital I/O
BSF STATUS,RP0 ;Bank 1
CLRF ANSEL ;digital I/O
MOVLW 0Ch ;Set RA<3:2> as inputs
MOVWF TRISA ;and set RA<5:4,1:0>
;as outputs
BCF STATUS,RP0 ;Bank 0
```

REGISTER 4-1: PORTA: PORTA REGISTER

U-0	U-0	R/W-x	R/W-0	R-x	R/W-0	R/W-0	R/W-0
—	—	RA5	RA4	RA3	RA2	RA1	RA0
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 **Unimplemented:** Read as '0'
 bit 5-0 **RA<5:0>:** PORTA I/O Pin bit
 1 = PORTA pin is > V_{IH}
 0 = PORTA pin is < V_{IL}

REGISTER 4-2: TRISA: PORTA TRI-STATE REGISTER

U-0	U-0	R/W-1	R/W-1	R-1	R/W-1	R/W-1	R/W-1
—	—	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0
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 **Unimplemented:** Read as '0'
 bit 5-0 **TRISA<5:0>:** PORTA Tri-State Control bit
 1 = PORTA pin configured as an input (tri-stated)
 0 = PORTA pin configured as an output

Note 1: TRISA<3> always reads '1'.
Note 2: TRISA<5:4> always reads '1' in XT, HS and LP OSC modes.

PIC16F684

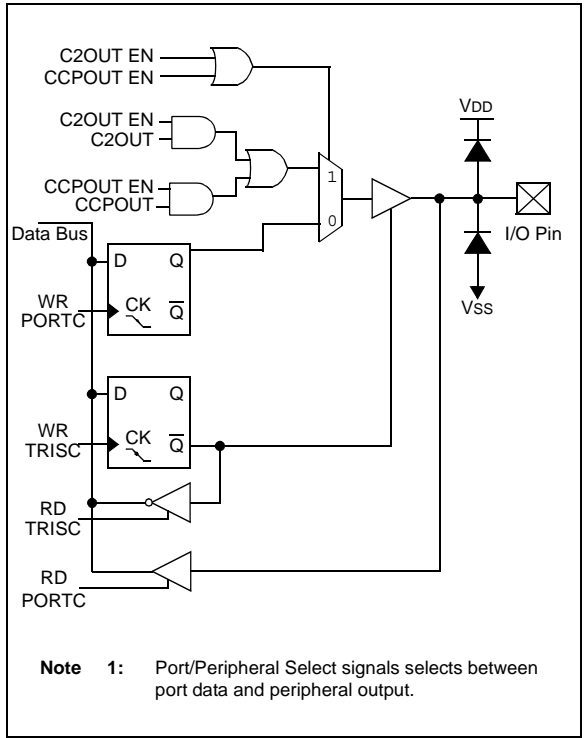
4.3.5 RC4/C2OUT/P1B

The RC4 is configurable to function as one of the following:

- a general purpose I/O
- a digital output from the comparator
- a digital output from the Enhanced CCP

Note: Enabling both C2OUT and P1B will cause a conflict on RC4 and create unpredictable results. Therefore, if C2OUT is enabled, the ECCP can not be used in Half-Bridge or Full-Bridge mode and vice-versa.

FIGURE 4-9: BLOCK DIAGRAM OF RC4



4.3.6 RC5/CCP1/P1A

The RC5 is configurable to function as one of the following:

- a general purpose I/O
- a digital input/output for the Enhanced CCP

FIGURE 4-10: BLOCK DIAGRAM OF RC5 PIN

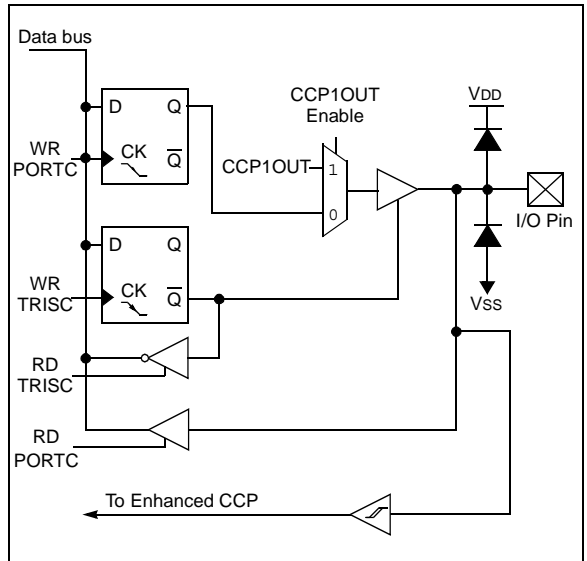


TABLE 4-2: SUMMARY OF REGISTERS ASSOCIATED WITH PORTC

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
ANSEL	ANS7	ANS6	ANS5	ANS4	ANS3	ANS2	ANS1	ANS0	1111 1111	1111 1111
CMCON0	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0000	0000 0000
PORTC	—	—	RC5	RC4	RC3	RC2	RC1	RC0	--xx 0000	--uu uu00
TRISC	—	—	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	--11 1111	--11 1111

Legend: x = unknown, u = unchanged, - = unimplemented locations read as '0'. Shaded cells are not used by PORTC.

7.0 TIMER2 MODULE

The Timer2 module is an eight-bit timer with the following features:

- 8-bit timer register (TMR2)
- 8-bit period register (PR2)
- Interrupt on TMR2 match with PR2
- Software programmable prescaler (1:1, 1:4, 1:16)
- Software programmable postscaler (1:1 to 1:16)

See Figure 7-1 for a block diagram of Timer2.

7.1 Timer2 Operation

The clock input to the Timer2 module is the system instruction clock ($F_{osc}/4$). The clock is fed into the Timer2 prescaler, which has prescale options of 1:1, 1:4 or 1:16. The output of the prescaler is then used to increment the TMR2 register.

The values of TMR2 and PR2 are constantly compared to determine when they match. TMR2 will increment from 00h until it matches the value in PR2. When a match occurs, two things happen:

- TMR2 is reset to 00h on the next increment cycle.
- The Timer2 postscaler is incremented

The match output of the Timer2/PR2 comparator is fed into the Timer2 postscaler. The postscaler has postscale options of 1:1 to 1:16 inclusive. The output of the Timer2 postscaler is used to set the TMR2IF interrupt flag bit in the PIR1 register.

The TMR2 and PR2 registers are both fully readable and writable. On any Reset, the TMR2 register is set to 00h and the PR2 register is set to FFh.

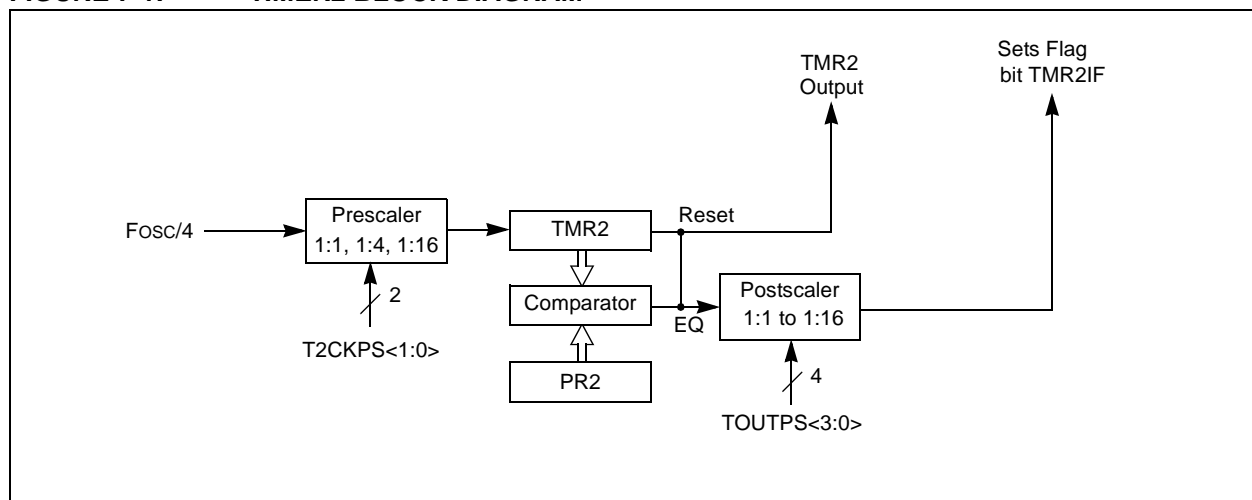
Timer2 is turned on by setting the TMR2ON bit in the T2CON register to a '1'. Timer2 is turned off by clearing the TMR2ON bit to a '0'.

The Timer2 prescaler is controlled by the T2CKPS bits in the T2CON register. The Timer2 postscaler is controlled by the TOUTPS bits in the T2CON register. The prescaler and postscaler counters are cleared when:

- A write to TMR2 occurs.
- A write to T2CON occurs.
- Any device Reset occurs (Power-on Reset, \overline{MCLR} Reset, Watchdog Timer Reset, or Brown-out Reset).

Note: TMR2 is not cleared when T2CON is written.

FIGURE 7-1: TIMER2 BLOCK DIAGRAM



PIC16F684

11.3.4 OPERATION IN SLEEP MODE

In Sleep mode, the TMR2 register will not increment and the state of the module will not change. If the CCP1 pin is driving a value, it will continue to drive that value. When the device wakes up, TMR2 will continue from its previous state.

11.3.5 CHANGES IN SYSTEM CLOCK FREQUENCY

The PWM frequency is derived from the system clock frequency. Any changes in the system clock frequency will result in changes to the PWM frequency. See **Section 3.0 “Oscillator Module (With Fail-Safe Clock Monitor)”** for additional details.

11.3.6 EFFECTS OF RESET

Any Reset will force all ports to Input mode and the CCP registers to their Reset states.

11.3.7 SETUP FOR PWM OPERATION

The following steps should be taken when configuring the CCP module for PWM operation:

1. Disable the PWM pin (CCP1) output driver by setting the associated TRIS bit.
2. Set the PWM period by loading the PR2 register.
3. Configure the CCP module for the PWM mode by loading the CCP1CON register with the appropriate values.
4. Set the PWM duty cycle by loading the CCPR1L register and CCP1 bits of the CCP1CON register.
5. Configure and start Timer2:
 - Clear the TMR2IF interrupt flag bit of the PIR1 register.
 - Set the Timer2 prescale value by loading the T2CKPS bits of the T2CON register.
 - Enable Timer2 by setting the TMR2ON bit of the T2CON register.
6. Enable PWM output after a new PWM cycle has started:
 - Wait until Timer2 overflows (TMR2IF bit of the PIR1 register is set).
 - Enable the CCP1 pin output driver by clearing the associated TRIS bit.

11.4.2.1 Direction Change in Full-Bridge Mode

In the Full-Bridge mode, the P1M1 bit in the CCP1CON register allows users to control the forward/reverse direction. When the application firmware changes this direction control bit, the module will change to the new direction on the next PWM cycle.

A direction change is initiated in software by changing the P1M1 bit of the CCP1CON register. The following sequence occurs four Timer2 cycles prior to the end of the current PWM period:

- The modulated outputs (P1B and P1D) are placed in their inactive state.
- The associated unmodulated outputs (P1A and P1C) are switched to drive in the opposite direction.
- PWM modulation resumes at the beginning of the next period.

See Figure 11-12 for an illustration of this sequence.

The Full-Bridge mode does not provide dead-band delay. As one output is modulated at a time, dead-band delay is generally not required. There is a situation where dead-band delay is required. This situation occurs when both of the following conditions are true:

1. The direction of the PWM output changes when the duty cycle of the output is at or near 100%.
2. The turn off time of the power switch, including the power device and driver circuit, is greater than the turn on time.

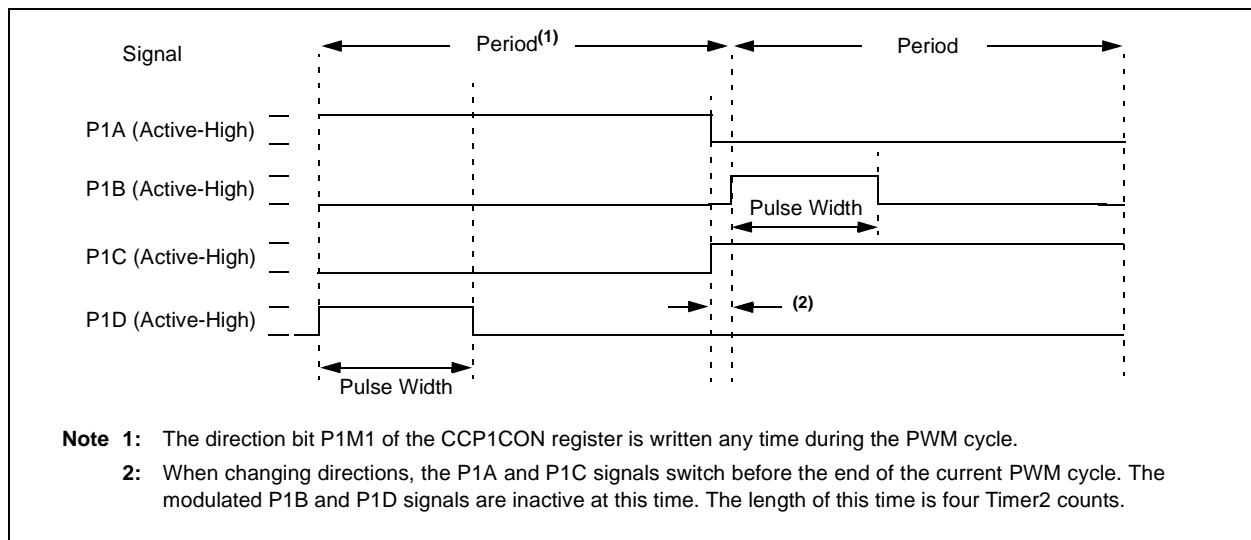
Figure 11-13 shows an example of the PWM direction changing from forward to reverse, at a near 100% duty cycle. In this example, at time t1, the output P1A and P1D become inactive, while output P1C becomes active. Since the turn off time of the power devices is longer than the turn on time, a shoot-through current will flow through power devices QC and QD (see Figure 11-10) for the duration of 't'. The same phenomenon will occur to power devices QA and QB for PWM direction change from reverse to forward.

If changing PWM direction at high duty cycle is required for an application, two possible solutions for eliminating the shoot-through current are:

1. Reduce PWM duty cycle for one PWM period before changing directions.
2. Use switch drivers that can drive the switches off faster than they can drive them on.

Other options to prevent shoot-through current may exist.

FIGURE 11-12: EXAMPLE OF PWM DIRECTION CHANGE



PIC16F684

REGISTER 11-3: PWM1CON: ENHANCED PWM CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PRSEN	PDC6	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0
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 **PRSEN:** PWM Restart Enable bit
 1 = Upon auto-shutdown, the ECCPASE bit clears automatically once the shutdown event goes away; the PWM restarts automatically
 0 = Upon auto-shutdown, ECCPASE must be cleared in software to restart the PWM
- bit 6-0 **PDC<6:0>:** PWM Delay Count bits
 PDCn = Number of Fosc/4 (4 * Tosc) cycles between the scheduled time when a PWM signal **should** transition active and the **actual** time it transitions active

Note 1: Bit resets to '0' with Two-Speed Start-up and LP, XT or HS selected as the Oscillator mode or Fail-Safe mode is enabled.

TABLE 11-5: SUMMARY OF REGISTERS ASSOCIATED WITH CAPTURE, COMPARE AND PWM

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
CCPR1L	Capture/Compare/PWM Register 1 Low Byte								xxxx xxxx	uuuu uuuu
CCPR1H	Capture/Compare/PWM Register 1 High Byte								xxxx xxxx	uuuu uuuu
CCP1CON	P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	0000 0000	0000 0000
CMCON0	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0000	0000 0000
CMCON1	—	—	—	—	—	—	T1GSS	C2SYNC	---- --10	---- --10
ECCPAS	ECCPASE	ECCPAS2	ECCPAS1	ECCPAS0	PSSAC1	PSSAC0	PSSBD1	PSSBD0	0000 0000	0000 0000
INTCON	GIE	PEIE	T0IE	INTE	RAIE	T0IF	INTF	RAIF	0000 0000	0000 0000
PIE1	EEIE	ADIE	CCP1IE	C2IE	C1IE	OSFIE	TMR2IE	TMR1IE	0000 0000	0000 0000
PIR1	EEIF	ADIF	CCP1IF	C2IF	C1IF	OSFIF	TMR2IF	TMR1IF	0000 0000	0000 0000
PR2	Timer2 Module Period Register								1111 1111	1111 1111
PWM1CON	PRSEN	PDC6	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0	0000 0000	0000 0000
T1CON	T1GINV	TMR1GE	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	TMR1CS	TMR1ON	0000 0000	uuuu uuuu
T2CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	-000 0000
TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
TMR2	Timer2 Module Register								0000 0000	0000 0000
TRISA	—	—	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	--11 1111	--11 1111
TRISC	—	—	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	--11 1111	--11 1111

Legend: - = Unimplemented locations, read as '0', u = unchanged, x = unknown. Shaded cells are not used by the Capture, Compare and PWM.

PIC16F684

REGISTER 12-1: CONFIG: CONFIGURATION WORD REGISTER

					FCMEN	IESO	BOREN1	BOREN0	
bit 15									bit 8

$\overline{\text{CPD}}$	$\overline{\text{CP}}$	MCLRE	$\overline{\text{PWRTE}}$	WDTE	FOSC2	FOSC1	FOSC0		
bit 7									bit 0

Legend:

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

bit 15-12 **Unimplemented:** Read as '1'

bit 11 **FCMEN:** Fail-Safe Clock Monitor Enabled bit
 1 = Fail-Safe Clock Monitor is enabled
 0 = Fail-Safe Clock Monitor is disabled

bit 10 **IESO:** Internal External Switchover bit
 1 = Internal External Switchover mode is enabled
 0 = Internal External Switchover mode is disabled

bit 9-8 **BOREN<1:0>:** Brown-out Reset Selection bits⁽¹⁾
 11 = BOR enabled
 10 = BOR enabled during operation and disabled in Sleep
 01 = BOR controlled by SBOREN bit of the PCON register
 00 = BOR disabled

bit 7 **CPD:** Data Code Protection bit⁽²⁾
 1 = Data memory code protection is disabled
 0 = Data memory code protection is enabled

bit 6 **CP:** Code Protection bit⁽³⁾
 1 = Program memory code protection is disabled
 0 = Program memory code protection is enabled

bit 5 **MCLRE:** RA3/MCLR pin function select bit⁽⁴⁾
 1 = RA3/MCLR pin function is MCLR
 0 = RA3/MCLR pin function is digital input, MCLR internally tied to VDD

bit 4 **PWRTE:** Power-up Timer Enable bit
 1 = PWRT disabled
 0 = PWRT enabled

bit 3 **WDTE:** Watchdog Timer Enable bit
 1 = WDT enabled
 0 = WDT disabled and can be enabled by SWDTEN bit of the WDTCON register

bit 2-0 **FOSC<2:0>:** Oscillator Selection bits
 111 = RC oscillator: CLKOUT function on RA4/OSC2/CLKOUT pin, RC on RA5/OSC1/CLKIN
 110 = RCIO oscillator: I/O function on RA4/OSC2/CLKOUT pin, RC on RA5/OSC1/CLKIN
 101 = INTOSC oscillator: CLKOUT function on RA4/OSC2/CLKOUT pin, I/O function on RA5/OSC1/CLKIN
 100 = INTOSCIO oscillator: I/O function on RA4/OSC2/CLKOUT pin, I/O function on RA5/OSC1/CLKIN
 011 = EC: I/O function on RA4/OSC2/CLKOUT pin, CLKIN on RA5/OSC1/CLKIN
 010 = HS oscillator: High-speed crystal/resonator on RA4/OSC2/CLKOUT and RA5/OSC1/CLKIN
 001 = XT oscillator: Crystal/resonator on RA4/OSC2/CLKOUT and RA5/OSC1/CLKIN
 000 = LP oscillator: Low-power crystal on RA4/OSC2/CLKOUT and RA5/OSC1/CLKIN

- Note**
- 1: Enabling Brown-out Reset does not automatically enable Power-up Timer.
 - 2: The entire data EEPROM will be erased when the code protection is turned off.
 - 3: The entire program memory will be erased when the code protection is turned off.
 - 4: When MCLR is asserted in INTOSC or RC mode, the internal clock oscillator is disabled.

PIC16F684

12.3.6 TIME-OUT SEQUENCE

On power-up, the time-out sequence is as follows:

- PWRT time-out is invoked after POR has expired.
- OST is activated after the PWRT time-out has expired.

The total time-out will vary based on oscillator configuration and PWRTE bit status. For example, in EC mode with PWRTE bit erased (PWRT disabled), there will be no time-out at all. Figure 12-4, Figure 12-5 and Figure 12-6 depict time-out sequences. The device can execute code from the INTOSC while OST is active by enabling Two-Speed Start-up or Fail-Safe Monitor (see **Section 3.7.2 “Two-speed Start-up Sequence”** and **Section 3.8 “Fail-Safe Clock Monitor”**).

Since the time-outs occur from the POR pulse, if $\overline{\text{MCLR}}$ is kept low long enough, the time-outs will expire. Then, bringing $\overline{\text{MCLR}}$ high will begin execution immediately (see Figure 12-5). This is useful for testing purposes or to synchronize more than one PIC16F684 device operating in parallel.

Table 12-5 shows the Reset conditions for some special registers, while Table 12-4 shows the Reset conditions for all the registers.

12.3.7 POWER CONTROL (PCON) REGISTER

The Power Control register PCON (address 8Eh) has two Status bits to indicate what type of Reset occurred last.

Bit 0 is $\overline{\text{BOR}}$ (Brown-out). $\overline{\text{BOR}}$ is unknown on Power-on Reset. It must then be set by the user and checked on subsequent Resets to see if $\overline{\text{BOR}} = 0$, indicating that a Brown-out has occurred. The $\overline{\text{BOR}}$ Status bit is a “don’t care” and is not necessarily predictable if the brown-out circuit is disabled ($\text{BOREN}<1:0> = 00$ in the Configuration Word register).

Bit 1 is $\overline{\text{POR}}$ (Power-on Reset). It is a ‘0’ on Power-on Reset and unaffected otherwise. The user must write a ‘1’ to this bit following a Power-on Reset. On a subsequent Reset, if $\overline{\text{POR}}$ is ‘0’, it will indicate that a Power-on Reset has occurred (i.e., VDD may have gone too low).

For more information, see **Section 4.2.4 “Ultra Low-Power Wake-up”** and **Section 12.3.4 “Brown-Out Reset (BOR)”**.

TABLE 12-1: TIME-OUT IN VARIOUS SITUATIONS

Oscillator Configuration	Power-up		Brown-out Reset		Wake-up from Sleep
	$\overline{\text{PWRTE}} = 0$	$\overline{\text{PWRTE}} = 1$	$\overline{\text{PWRTE}} = 0$	$\overline{\text{PWRTE}} = 1$	
XT, HS, LP	$\text{TPWRT} + 1024 \cdot \text{TOSC}$	$1024 \cdot \text{TOSC}$	$\text{TPWRT} + 1024 \cdot \text{TOSC}$	$1024 \cdot \text{TOSC}$	$1024 \cdot \text{TOSC}$
RC, EC, INTOSC	TPWRT	—	TPWRT	—	—

TABLE 12-2: STATUS/PCON BITS AND THEIR SIGNIFICANCE

$\overline{\text{POR}}$	$\overline{\text{BOR}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	Condition
0	x	1	1	Power-on Reset
u	0	1	1	Brown-out Reset
u	u	0	u	WDT Reset
u	u	0	0	WDT Wake-up
u	u	u	u	$\overline{\text{MCLR}}$ Reset during normal operation
u	u	1	0	$\overline{\text{MCLR}}$ Reset during Sleep

Legend: u = unchanged, x = unknown

TABLE 12-3: SUMMARY OF REGISTERS ASSOCIATED WITH BROWN-OUT RESET

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets ⁽¹⁾
PCON	—	—	ULPWUE	SBOREN	—	—	$\overline{\text{POR}}$	$\overline{\text{BOR}}$	--01 --qq	--0u --uu
STATUS	IRP	RP1	RP0	$\overline{\text{TO}}$	$\overline{\text{PD}}$	Z	DC	C	0001 1xxx	000q quuu

Legend: u = unchanged, x = unknown, — = unimplemented bit, reads as ‘0’, q = value depends on condition. Shaded cells are not used by BOR.

Note 1: Other (non Power-up) Resets include $\overline{\text{MCLR}}$ Reset and Watchdog Timer Reset during normal operation.

PIC16F684

12.7 Power-Down Mode (Sleep)

The Power-down mode is entered by executing a `SLEEP` instruction.

If the Watchdog Timer is enabled:

- WDT will be cleared but keeps running.
- \overline{PD} bit in the STATUS register is cleared.
- \overline{TO} bit is set.
- Oscillator driver is turned off.
- I/O ports maintain the status they had before `SLEEP` was executed (driving high, low or high-impedance).

For lowest current consumption in this mode, all I/O pins should be either at V_{DD} or V_{SS} , with no external circuitry drawing current from the I/O pin and the comparators and CV_{REF} should be disabled. I/O pins that are high-impedance inputs should be pulled high or low externally to avoid switching currents caused by floating inputs. The $TOCKI$ input should also be at V_{DD} or V_{SS} for lowest current consumption. The contribution from on-chip pull-ups on $PORTA$ should be considered.

The \overline{MCLR} pin must be at a logic high level.

Note: It should be noted that a Reset generated by a WDT time-out does not drive \overline{MCLR} pin low.

12.7.1 WAKE-UP FROM SLEEP

The device can wake-up from Sleep through one of the following events:

1. External Reset input on \overline{MCLR} pin.
2. Watchdog Timer wake-up (if WDT was enabled).
3. Interrupt from $RA2/INT$ pin, $PORTA$ change or a peripheral interrupt.

The first event will cause a device Reset. The two latter events are considered a continuation of program execution. The \overline{TO} and \overline{PD} bits in the STATUS register can be used to determine the cause of a device Reset. The \overline{PD} bit, which is set on power-up, is cleared when Sleep is invoked. \overline{TO} bit is cleared if WDT wake-up occurred.

The following peripheral interrupts can wake the device from Sleep:

1. Timer1 interrupt. Timer1 must be operating as an asynchronous counter.
2. ECCP Capture mode interrupt.
3. A/D conversion (when A/D clock source is FRC).
4. EEPROM write operation completion.
5. Comparator output changes state.
6. Interrupt-on-change.
7. External Interrupt from INT pin.

Other peripherals cannot generate interrupts since during Sleep, no on-chip clocks are present.

When the `SLEEP` instruction is being executed, the next instruction ($PC + 1$) is prefetched. For the device to wake-up through an interrupt event, the corresponding interrupt enable bit must be set (enabled). Wake-up occurs regardless of the state of the GIE bit. If the GIE bit is clear (disabled), the device continues execution at the instruction after the `SLEEP` instruction. If the GIE bit is set (enabled), the device executes the instruction after the `SLEEP` instruction, then branches to the interrupt address (0004h). In cases where the execution of the instruction following `SLEEP` is not desirable, the user should have a `NOP` after the `SLEEP` instruction.

Note: If the global interrupts are disabled (GIE is cleared) and any interrupt source has both its interrupt enable bit and the corresponding interrupt flag bits set, the device will immediately wake-up from Sleep.

The WDT is cleared when the device wakes up from Sleep, regardless of the source of wake-up.

12.7.2 WAKE-UP USING INTERRUPTS

When global interrupts are disabled (GIE cleared) and any interrupt source has both its interrupt enable bit and interrupt flag bit set, one of the following will occur:

- If the interrupt occurs **before** the execution of a `SLEEP` instruction, the `SLEEP` instruction will complete as a `NOP`. Therefore, the WDT and WDT prescaler and postscaler (if enabled) will not be cleared, the \overline{TO} bit will not be set and the \overline{PD} bit will not be cleared.
- If the interrupt occurs **during or after** the execution of a `SLEEP` instruction, the device will immediately wake-up from Sleep. The `SLEEP` instruction is executed. Therefore, the WDT and WDT prescaler and postscaler (if enabled) will be cleared, the \overline{TO} bit will be set and the \overline{PD} bit will be cleared.

Even if the flag bits were checked before executing a `SLEEP` instruction, it may be possible for flag bits to become set before the `SLEEP` instruction completes. To determine whether a `SLEEP` instruction executed, test the \overline{PD} bit. If the \overline{PD} bit is set, the `SLEEP` instruction was executed as a `NOP`.

To ensure that the WDT is cleared, a `CLRWDT` instruction should be executed before a `SLEEP` instruction. See Figure 12-10 for more details.

13.2 Instruction Descriptions

ADDLW **Add literal and W**

Syntax: [*label*] ADDLW *k*

Operands: $0 \leq k \leq 255$

Operation: $(W) + k \rightarrow (W)$

Status Affected: C, DC, Z

Description: The contents of the W register are added to the eight-bit literal 'k' and the result is placed in the W register.

BCF **Bit Clear f**

Syntax: [*label*] BCF *f*,*b*

Operands: $0 \leq f \leq 127$
 $0 \leq b \leq 7$

Operation: $0 \rightarrow (f)$

Status Affected: None

Description: Bit 'b' in register 'f' is cleared.

ADDWF **Add W and f**

Syntax: [*label*] ADDWF *f*,*d*

Operands: $0 \leq f \leq 127$
 $d \in [0,1]$

Operation: $(W) + (f) \rightarrow (\text{destination})$

Status Affected: C, DC, Z

Description: Add the contents of the W register with register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'.

BSF **Bit Set f**

Syntax: [*label*] BSF *f*,*b*

Operands: $0 \leq f \leq 127$
 $0 \leq b \leq 7$

Operation: $1 \rightarrow (f)$

Status Affected: None

Description: Bit 'b' in register 'f' is set.

ANDLW **AND literal with W**

Syntax: [*label*] ANDLW *k*

Operands: $0 \leq k \leq 255$

Operation: $(W) .\text{AND.} (k) \rightarrow (W)$

Status Affected: Z

Description: The contents of W register are AND'ed with the eight-bit literal 'k'. The result is placed in the W register.

BTFSC **Bit Test, Skip if Clear**

Syntax: [*label*] BTFSC *f*,*b*

Operands: $0 \leq f \leq 127$
 $0 \leq b \leq 7$

Operation: skip if $(f) = 0$

Status Affected: None

Description: If bit 'b' in register 'f' is '1', the next instruction is executed.
If bit 'b', in register 'f', is '0', the next instruction is discarded, and a NOP is executed instead, making this a 2-cycle instruction.

ANDWF **AND W with f**

Syntax: [*label*] ANDWF *f*,*d*

Operands: $0 \leq f \leq 127$
 $d \in [0,1]$

Operation: $(W) .\text{AND.} (f) \rightarrow (\text{destination})$

Status Affected: Z

Description: AND the W register with register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'.

PIC16F684

MOVF	Move f
Syntax:	[<i>label</i>] MOVF f,d
Operands:	$0 \leq f \leq 127$ $d \in [0,1]$
Operation:	(f) → (dest)
Status Affected:	Z
Description:	The contents of register f is moved to a destination dependent upon the status of d. If d = 0, destination is W register. If d = 1, the destination is file register f itself. d = 1 is useful to test a file register since status flag Z is affected.
Words:	1
Cycles:	1
Example:	<pre>MOVF FSR, 0</pre> <p>After Instruction</p> <pre>W = value in FSR register Z = 1</pre>

MOVLW	Move literal to W
Syntax:	[<i>label</i>] MOVLW k
Operands:	$0 \leq k \leq 255$
Operation:	k → (W)
Status Affected:	None
Description:	The eight-bit literal 'k' is loaded into W register. The "don't cares" will assemble as '0's.
Words:	1
Cycles:	1
Example:	<pre>MOVLW 0x5A</pre> <p>After Instruction</p> <pre>W = 0x5A</pre>

MOVWF	Move W to f
Syntax:	[<i>label</i>] MOVWF f
Operands:	$0 \leq f \leq 127$
Operation:	(W) → (f)
Status Affected:	None
Description:	Move data from W register to register 'f'.
Words:	1
Cycles:	1
Example:	<pre>MOVWF OPTION F</pre> <p>Before Instruction</p> <pre>OPTION = 0xFF W = 0x4F</pre> <p>After Instruction</p> <pre>OPTION = 0x4F W = 0x4F</pre>

NOP	No Operation
Syntax:	[<i>label</i>] NOP
Operands:	None
Operation:	No operation
Status Affected:	None
Description:	No operation.
Words:	1
Cycles:	1
Example:	<pre>NOP</pre>

RETFIE	Return from Interrupt
Syntax:	[<i>label</i>] RETFIE
Operands:	None
Operation:	TOS → PC, 1 → GIE
Status Affected:	None
Description:	Return from Interrupt. Stack is POPed and Top-of-Stack (TOS) is loaded in the PC. Interrupts are enabled by setting Global Interrupt Enable bit, GIE (INTCON<7>). This is a two-cycle instruction.
Words:	1
Cycles:	2
<u>Example:</u>	<pre>RETFIE</pre> <p>After Interrupt</p> <pre>PC = TOS GIE = 1</pre>

RETLW	Return with literal in W
Syntax:	[<i>label</i>] RETLW k
Operands:	0 ≤ k ≤ 255
Operation:	k → (W); TOS → PC
Status Affected:	None
Description:	The W register is loaded with the eight bit literal 'k'. The program counter is loaded from the top of the stack (the return address). This is a two-cycle instruction.
Words:	1
Cycles:	2
<u>Example:</u>	<pre>CALL TABLE;W contains table ;offset value ;W now has table value • • • ADDWF PC ;W = offset RETLW k1 ;Begin table RETLW k2 ; • • • RETLW kn ; End of table</pre> <p>Before Instruction W = 0x07</p> <p>After Instruction W = value of k8</p>

RETURN	Return from Subroutine
Syntax:	[<i>label</i>] RETURN
Operands:	None
Operation:	TOS → PC
Status Affected:	None
Description:	Return from subroutine. The stack is POPed and the top of the stack (TOS) is loaded into the program counter. This is a two-cycle instruction.

FIGURE 16-28: SCHMITT TRIGGER INPUT THRESHOLD V_{IN} vs. V_{DD} OVER TEMPERATURE

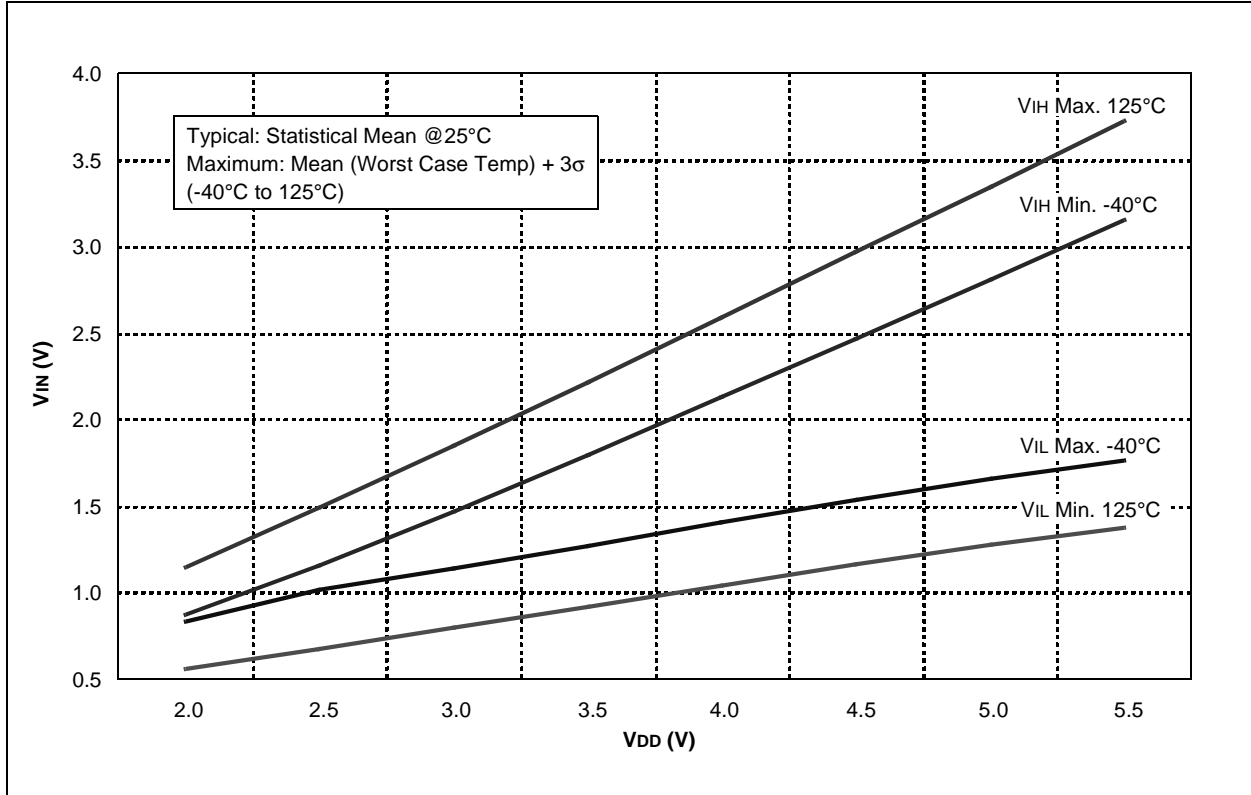
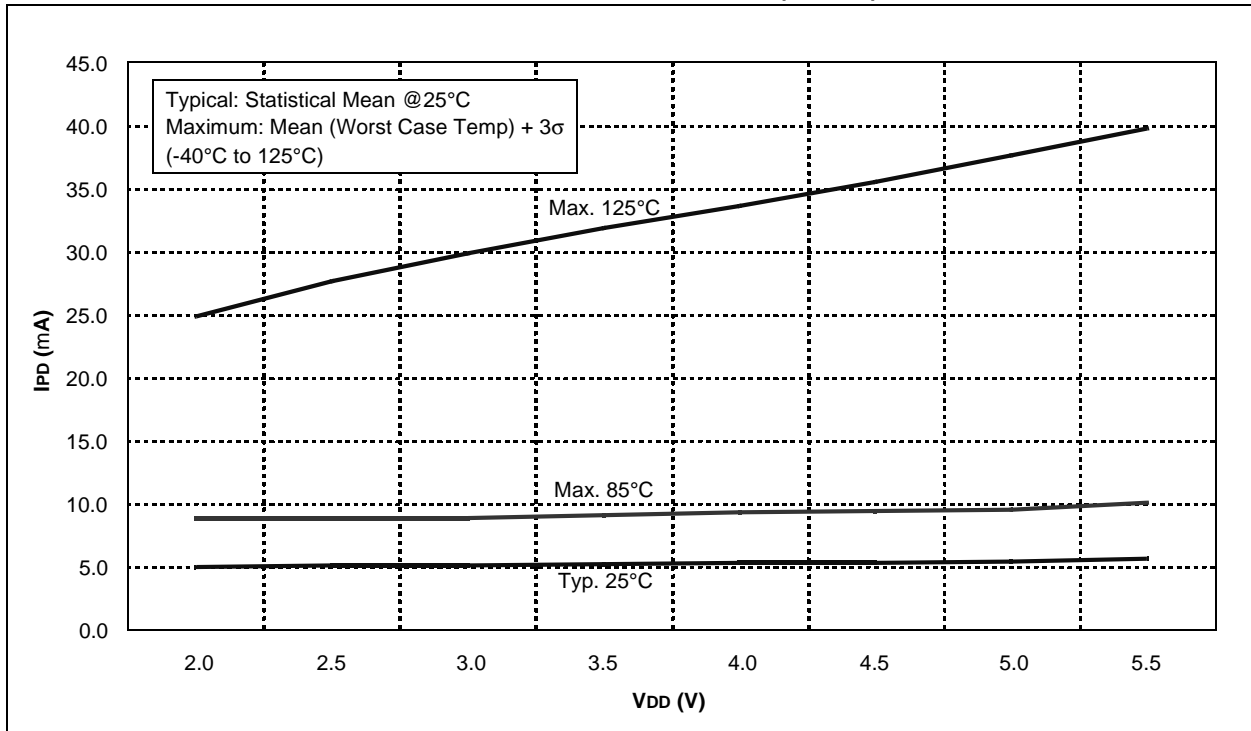


FIGURE 16-29: T1OSC I_{PD} vs. V_{DD} OVER TEMPERATURE (32 kHz)



INDEX

A

A/D	
Specifications.....	147, 148
Absolute Maximum Ratings	129
AC Characteristics	
Industrial and Extended	139
Load Conditions	138
ADC	65
Acquisition Requirements	72
Associated registers.....	74
Block Diagram.....	65
Calculating Acquisition Time.....	72
Channel Selection.....	66
Configuration.....	66
Configuring Interrupt.....	68
Conversion Clock.....	66
Conversion Procedure.....	68
Internal Sampling Switch (Rss) Impedance.....	72
Interrupts.....	67
Operation.....	68
Operation During Sleep	68
Port Configuration.....	66
Reference Voltage (VREF).....	66
Result Formatting.....	67
Source Impedance.....	72
Special Event Trigger.....	68
Starting an A/D Conversion	67
ADCON0 Register.....	70
ADCON1 Register.....	70
ADRESH Register (ADFM = 0).....	71
ADRESH Register (ADFM = 1).....	71
ADRESL Register (ADFM = 0).....	71
ADRESL Register (ADFM = 1).....	71
Analog Input Connection Considerations.....	57
Analog-to-Digital Converter. <i>See</i> ADC	
ANSEL Register.....	32
Assembler	
MPASM Assembler.....	126

B

Block Diagrams	
(CCP) Capture Mode Operation	80
ADC	65
ADC Transfer Function	73
Analog Input Model.....	57, 73
CCP PWM.....	82
Clock Source.....	19
Comparator 1	56
Comparator 2	56
Comparator Modes	58
Compare	81
Crystal Operation.....	22
External RC Mode.....	23
Fail-Safe Clock Monitor (FSCM).....	29
In-Circuit Serial Programming Connections.....	114
Interrupt Logic.....	107
MCLR Circuit.....	100
On-Chip Reset Circuit.....	99
PIC16F684.....	5
PWM (Enhanced).....	85
RA0 Pins	35
RA1 Pins	36
RA2 Pin.....	36
RA3 Pin.....	37

RA4 Pin	37
RA5 Pin	38
RC0 and RC1 Pins	41
RC2 and RC3 Pins	41
RC4 Pin	42
RC5 Pin	42
Resonator Operation	22
Timer1	47
Timer2	53
TMR0/WDT Prescaler	43
Watchdog Timer (WDT).....	110
Brown-out Reset (BOR).....	101
Associated	102
Calibration	101
Specifications	143
Timing and Characteristics	142

C

C Compilers	
MPLAB C18.....	126
MPLAB C30.....	126
Calibration Bits.....	99
Capture Module. <i>See</i> Enhanced Capture/	
Compare/PWM (ECCP)	
Capture/Compare/PWM (CCP)	
Associated registers w/ Capture/Compare/PWM	96
Capture Mode.....	80
CCP1 Pin Configuration	80
Compare Mode.....	81
CCP1 Pin Configuration	81
Software Interrupt Mode.....	80, 81
Special Event Trigger	81
Timer1 Mode Selection.....	80, 81
Prescaler	80
PWM Mode.....	82
Duty Cycle	83
Effects of Reset	84
Example PWM Frequencies and	
Resolutions, 20 MHz.....	83
Example PWM Frequencies and	
Resolutions, 8 MHz	83
Operation in Sleep Mode.....	84
Setup for Operation	84
System Clock Frequency Changes	84
PWM Period	83
Setup for PWM Operation	84
CCP1CON (Enhanced) Register	79
Clock Sources	
External Modes.....	21
EC	21
HS	22
LP	22
OST	21
RC	23
XT	22
Internal Modes.....	23
Frequency Selection.....	25
HFINTOSC	23
HFINTOSC/LFINTOSC Switch Timing.....	25
INTOSC	23
INTOSCIO	23
LFINTOSC.....	25
Clock Switching	27
CMCON0 Register.....	61