

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	32MHz
Connectivity	LINbus, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	12
Program Memory Size	14KB (8K x 14)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	A/D 8x10b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	16-UQFN Exposed Pad
Supplier Device Package	16-UQFN (4x4)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16lf1575-i-jq

3.2.1 READING PROGRAM MEMORY AS DATA

There are two methods of accessing constants in program memory. The first method is to use tables of RETLW instructions. The second method is to set an FSR to point to the program memory.

3.2.1.1 RETLW Instruction

The RETLW instruction can be used to provide access to tables of constants. The recommended way to create such a table is shown in Example 3-1.

EXAMPLE 3-1: RETLW INSTRUCTION

```
constants
    BRW                ;Add Index in W to
                        ;program counter to
                        ;select data
    RETLW DATA0        ;Index0 data
    RETLW DATA1        ;Index1 data
    RETLW DATA2
    RETLW DATA3

my_function
    ;... LOTS OF CODE...
    MOVLW DATA_INDEX
    call constants
    ;... THE CONSTANT IS IN W
```

The BRW instruction makes this type of table very simple to implement. If your code must remain portable with previous generations of microcontrollers, then the BRW instruction is not available so the older table read method must be used.

3.2.1.2 Indirect Read with FSR

The program memory can be accessed as data by setting bit 7 of the FSRxH register and reading the matching INDFx register. The MOVIW instruction will place the lower eight bits of the addressed word in the W register. Writes to the program memory cannot be performed via the INDF registers. Instructions that access the program memory via the FSR require one extra instruction cycle to complete. Example 3-2 demonstrates accessing the program memory via an FSR.

The HIGH operator will set bit<7> if a label points to a location in program memory.

EXAMPLE 3-2: ACCESSING PROGRAM MEMORY VIA FSR

```
constants
    DW DATA0          ;First constant
    DW DATA1          ;Second constant
    DW DATA2
    DW DATA3

my_function
    ;... LOTS OF CODE...
    MOVLW DATA_INDEX
    ADDLW LOW constants
    MOVWF FSR1L
    MOVLW HIGH constants;MSb is set
                        automatically
    MOVWF FSR1H
    BTFSC STATUS,C      ;carry from ADDLW?
    INCF FSR1H,f        ;yes
    MOVIW 0[FSR1]
    ;THE PROGRAM MEMORY IS IN W
```

TABLE 3-8: PIC16(L)F1575/9 MEMORY MAP, BANKS 8-15

BANK 8		BANK 9		BANK 10		BANK 11		BANK 12		BANK 13		BANK 14		BANK 15							
400h	Core Registers (Table 3-2)	480h	Core Registers (Table 3-2)	500h	Core Registers (Table 3-2)	580h	Core Registers (Table 3-2)	600h	Core Registers (Table 3-2)	680h	Core Registers (Table 3-2)	700h	Core Registers (Table 3-2)	780h	Core Registers (Table 3-2)						
40Bh		48Bh		50Bh		58Bh		60Bh		68Bh		70Bh		78Bh							
40Ch	—	48Ch	—	50Ch	—	58Ch	—	60Ch	—	68Ch	—	70Ch	—	78Ch	—						
40Dh	—	48Dh	—	50Dh	—	58Dh	—	60Dh	—	68Dh	—	70Dh	—	78Dh	—						
40Eh	—	48Eh	—	50Eh	—	58Eh	—	60Eh	—	68Eh	—	70Eh	—	78Eh	—						
40Fh	—	48Fh	—	50Fh	—	58Fh	—	60Fh	—	68Fh	—	70Fh	—	78Fh	—						
410h	—	490h	—	510h	—	590h	—	610h	—	690h	—	710h	—	790h	—						
411h	—	491h	—	511h	—	591h	—	611h	—	691h	CWG1DBR	711h	—	791h	—						
412h	—	492h	—	512h	—	592h	—	612h	—	692h	CWG1DBF	712h	—	792h	—						
413h	—	493h	—	513h	—	593h	—	613h	—	693h	CWG1CON0	713h	—	793h	—						
414h	—	494h	—	514h	—	594h	—	614h	—	694h	CWG1CON1	714h	—	794h	—						
415h	—	495h	—	515h	—	595h	—	615h	—	695h	CWG1CON2	715h	—	795h	—						
416h	—	496h	—	516h	—	596h	—	616h	—	696h	—	716h	—	796h	—						
417h	—	497h	—	517h	—	597h	—	617h	—	697h	—	717h	—	797h	—						
418h	—	498h	—	518h	—	598h	—	618h	—	698h	—	718h	—	798h	—						
419h	—	499h	—	519h	—	599h	—	619h	—	699h	—	719h	—	799h	—						
41Ah	—	49Ah	—	51Ah	—	59Ah	—	61Ah	—	69Ah	—	71Ah	—	79Ah	—						
41Bh	—	49Bh	—	51Bh	—	59Bh	—	61Bh	—	69Bh	—	71Bh	—	79Bh	—						
41Ch	—	49Ch	—	51Ch	—	59Ch	—	61Ch	—	69Ch	—	71Ch	—	79Ch	—						
41Dh	—	49Dh	—	51Dh	—	59Dh	—	61Dh	—	69Dh	—	71Dh	—	79Dh	—						
41Eh	—	49Eh	—	51Eh	—	59Eh	—	61Eh	—	69Eh	—	71Eh	—	79Eh	—						
41Fh	—	49Fh	—	51Fh	—	59Fh	—	61Fh	—	69Fh	—	71Fh	—	79Fh	—						
420h	General Purpose Register 80 Bytes	4A0h	General Purpose Register 80 Bytes	520h	General Purpose Register 80 Bytes	5A0h	General Purpose Register 80 Bytes	620h	General Purpose Register 32 Bytes	Unimplemented Read as '0'	Unimplemented Read as '0'	Unimplemented Read as '0'	Unimplemented Read as '0'	Unimplemented Read as '0'	Unimplemented Read as '0'						
								63Fh	Unimplemented Read as '0'												
								640h													
46Fh	Accesses 70h – 7Fh	4EFh	Accesses 70h – 7Fh	56Fh	Accesses 70h – 7Fh	5EFh	Accesses 70h – 7Fh	66Fh	Accesses 70h – 7Fh	6EFh	Accesses 70h – 7Fh	76Fh	Accesses 70h – 7Fh	7EFh	Accesses 70h – 7Fh						
470h		4F0h		570h		5F0h		670h		6F0h		770h		7F0h							
47Fh		4FFh		57Fh		5FFh		67Fh		6FFh		77Fh		7FFh							

Legend: = Unimplemented data memory locations, read as '0'

TABLE 3-15: SPECIAL FUNCTION REGISTER SUMMARY

Address	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
Bank 0											
00Ch	PORTA	—	—	RA5	RA4	RA3	RA2	RA1	RA0	--xx xxxx	--xx xxxx
00Dh	PORTB ⁽¹⁾	RB7	RB6	RB5	RB4	—	—	—	—	xxxx ----	xxxx ----
00Eh	PORTC	RC7 ⁽¹⁾	RC6 ⁽¹⁾	RC5	RC4	RC3	RC2	RC1	RC0	xxxx xxxx	xxxx xxxx
00Fh	—	Unimplemented								—	—
010h	—	Unimplemented								—	—
011h	PIR1	TMR1GIF	ADIF	RCIF	TXIF	—	—	TMR2IF	TMR1IF	0000 --00	0000 --00
012h	PIR2	—	C2IF	C1IF	—	—	—	—	—	-00- ----	-00- ----
013h	PIR3	PWM4IF	PWM3IF	PWM2IF	PWM1IF	—	—	—	—	0000 ----	0000 ----
014h	—									—	—
015h	TMR0	Holding Register for the 8-bit Timer0 Count								xxxx xxxx	uuuu uuuu
016h	TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Count								xxxx xxxx	uuuu uuuu
017h	TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Count								xxxx xxxx	uuuu uuuu
018h	T1CON	TMR1CS<1:0>		T1CKPS<1:0>		—	T1SYNC	—	TMR1ON	0000 -0-0	uuuu -u-u
019h	T1GCON	TMR1GE	T1GPOL	T1GTM	T1GSPM	T1GGO/ DONE	T1GVAL	T1GSS<1:0>		0000 0x00	uuuu uxuu
01Ah	TMR2	Timer2 Module Register								0000 0000	0000 0000
01Bh	PR2	Timer2 Period Register								1111 1111	1111 1111
01Ch	T2CON	—	T2OUTPS<3:0>				TMR2ON	T2CKPS<1:0>		-000 0000	-000 0000
01Dh	—	Unimplemented								—	—
01Eh	—	Unimplemented								—	—
01Fh	—	Unimplemented								—	—

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, r = reserved. Shaded locations are unimplemented, read as '0'.

- Note**
- 1: PIC16(L)F1578/9 only.
 - 2: PIC16F1574/5/8/9 only.
 - 3: Unimplemented, read as '1'.

3.5 Stack

All devices have a 16-level x 15-bit wide hardware stack (refer to Figures 3-5 through 3-8). The stack space is not part of either program or data space. The PC is PUSHed onto the stack when `CALL` or `CALLW` instructions are 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 if the `STVREN` bit is programmed to '0' (Configuration Words). This means that after the stack has been PUSHed sixteen times, the seventeenth `PUSH` overwrites the value that was stored from the first `PUSH`. The eighteenth `PUSH` overwrites the second `PUSH` (and so on). The `STKOVF` and `STKUNF` flag bits will be set on an Overflow/Underflow, regardless of whether the Reset is enabled.

Note 1: There are no instructions/mnemonics called `PUSH` or `POP`. These are actions that occur from the execution of the `CALL`, `CALLW`, `RETURN`, `RETLW` and `RETFIE` instructions or the vectoring to an interrupt address.

3.5.1 ACCESSING THE STACK

The stack is available through the `TOSH`, `TOSL` and `STKPTR` registers. `STKPTR` is the current value of the Stack Pointer. `TOSH:TOSL` register pair points to the TOP of the stack. Both registers are read/writable. `TOS` is split into `TOSH` and `TOSL` due to the 15-bit size of the PC. To access the stack, adjust the value of `STKPTR`, which will position `TOSH:TOSL`, then read/write to `TOSH:TOSL`. `STKPTR` is five bits to allow detection of overflow and underflow.

Note: Care should be taken when modifying the `STKPTR` while interrupts are enabled.

During normal program operation, `CALL`, `CALLW` and Interrupts will increment `STKPTR` while `RETLW`, `RETURN`, and `RETFIE` will decrement `STKPTR`. At any time `STKPTR` can be inspected to see how much stack is left. The `STKPTR` always points at the currently used place on the stack. Therefore, a `CALL` or `CALLW` will increment the `STKPTR` and then write the PC, and a return will unload the PC and then decrement the `STKPTR`.

Reference Figure 3-5 through Figure 3-8 for examples of accessing the stack.

FIGURE 3-5: ACCESSING THE STACK EXAMPLE 1

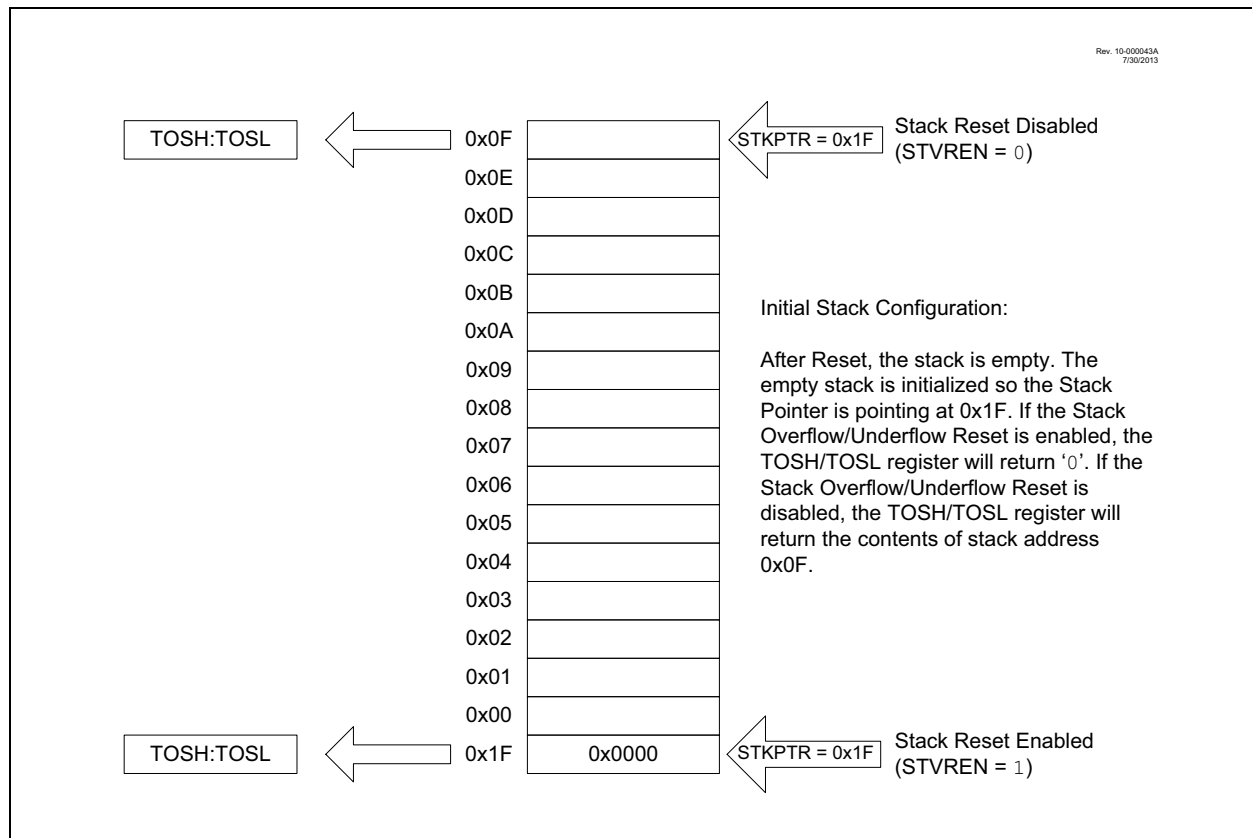
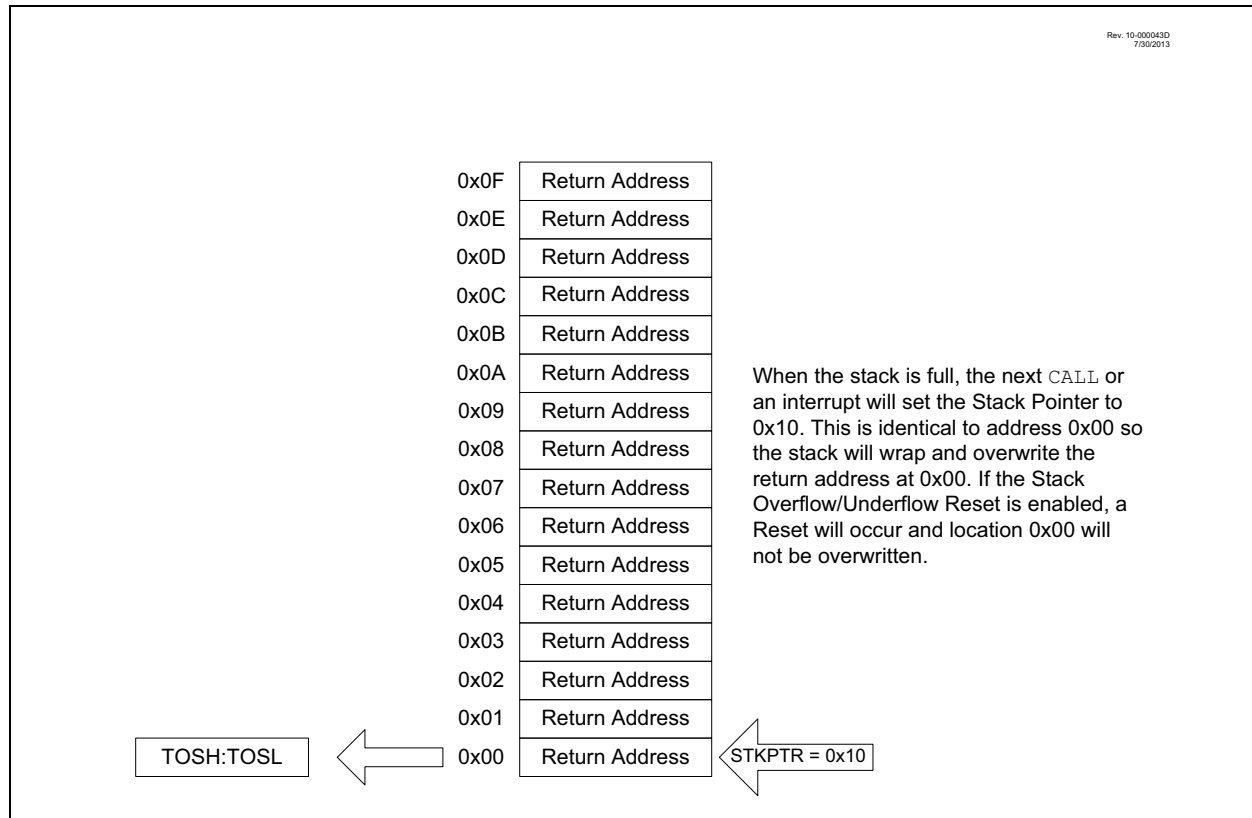


FIGURE 3-8: ACCESSING THE STACK EXAMPLE 4



3.5.2 OVERFLOW/UNDERFLOW RESET

If the STVREN bit in Configuration Words is programmed to '1', the device will be reset if the stack is PUSHed beyond the sixteenth level or POPed beyond the first level, setting the appropriate bits (STKOVF or STKUNF, respectively) in the PCON register.

3.6 Indirect Addressing

The INDFn registers are not physical registers. Any instruction that accesses an INDFn register actually accesses the register at the address specified by the File Select Registers (FSR). If the FSRn address specifies one of the two INDFn registers, the read will return '0' and the write will not occur (though Status bits may be affected). The FSRn register value is created by the pair FSRnH and FSRnL.

The FSR registers form a 16-bit address that allows an addressing space with 65536 locations. These locations are divided into three memory regions:

- Traditional Data Memory
- Linear Data Memory
- Program Flash Memory

PIC16(L)F1574/5/8/9

REGISTER 4-2: CONFIGURATION WORD 2 (CONTINUED)

bit 1-0

WRT<1:0>: Flash Memory Self-Write Protection bits

4 kW Flash memory: (PIC16(L)F1574/8):

11 = OFF	Write protection off
10 = BOOT	0000h to 1FFh write protected, 0200h to 0FFFh may be modified by PMCON control
01 = HALF	0000h to 07FFh write protected, 0800h to 0FFFh may be modified by PMCON control
00 = ALL	0000h to 0FFFh write protected, no addresses may be modified by PMCON control

8 kW Flash memory: (PIC16(L)F1575/9)

11 = OFF	Write protection off
10 = BOOT	0000h to 1FFh write protected, 0200h to 1FFFh may be modified by PMCON control
01 = HALF	0000h to 0FFFh write protected, 1000h to 1FFFh may be modified by PMCON control
00 = ALL	0000h to 1FFFh write protected, no addresses may be modified by PMCON control

- Note 1:** This bit cannot be programmed to '0' when programming mode is entered via LVP.
- 2:** The `DEBUG` bit in Configuration Words is managed automatically by device development tools including debuggers and programmers. For normal device operation, this bit should be maintained as a '1'.
- 3:** See VBOR parameter for specific trip point voltages.

PIC16(L)F1574/5/8/9

TABLE 5-2: SUMMARY OF REGISTERS ASSOCIATED WITH CLOCK SOURCES

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
OSCCON	SPLLEN	IRCF<3:0>				—	SCS<1:0>		69
OSCSTAT	—	PLL R	OSTS	HFIOFR	HFIOFL	MFIOFR	LFIOFR	HFIOFS	70
OSCTUNE	—	—	TUN<5:0>						71

Legend: — = unimplemented location, read as '0'. Shaded cells are not used by clock sources.

TABLE 5-3: SUMMARY OF CONFIGURATION WORD WITH CLOCK SOURCES

Name	Bits	Bit -/7	Bit -/6	Bit 13/5	Bit 12/4	Bit 11/3	Bit 10/2	Bit 9/1	Bit 8/0	Register on Page
CONFIG1	13:8	—	—	—	—	CLKOUTEN	BOREN<1:0>		—	56
	7:0	$\overline{\text{CP}}$	MCLRE	$\overline{\text{PWRTE}}$	WDTE<1:0>		—	FOSC<1:0>		

Legend: — = unimplemented location, read as '0'. Shaded cells are not used by clock sources.

6.0 RESETS

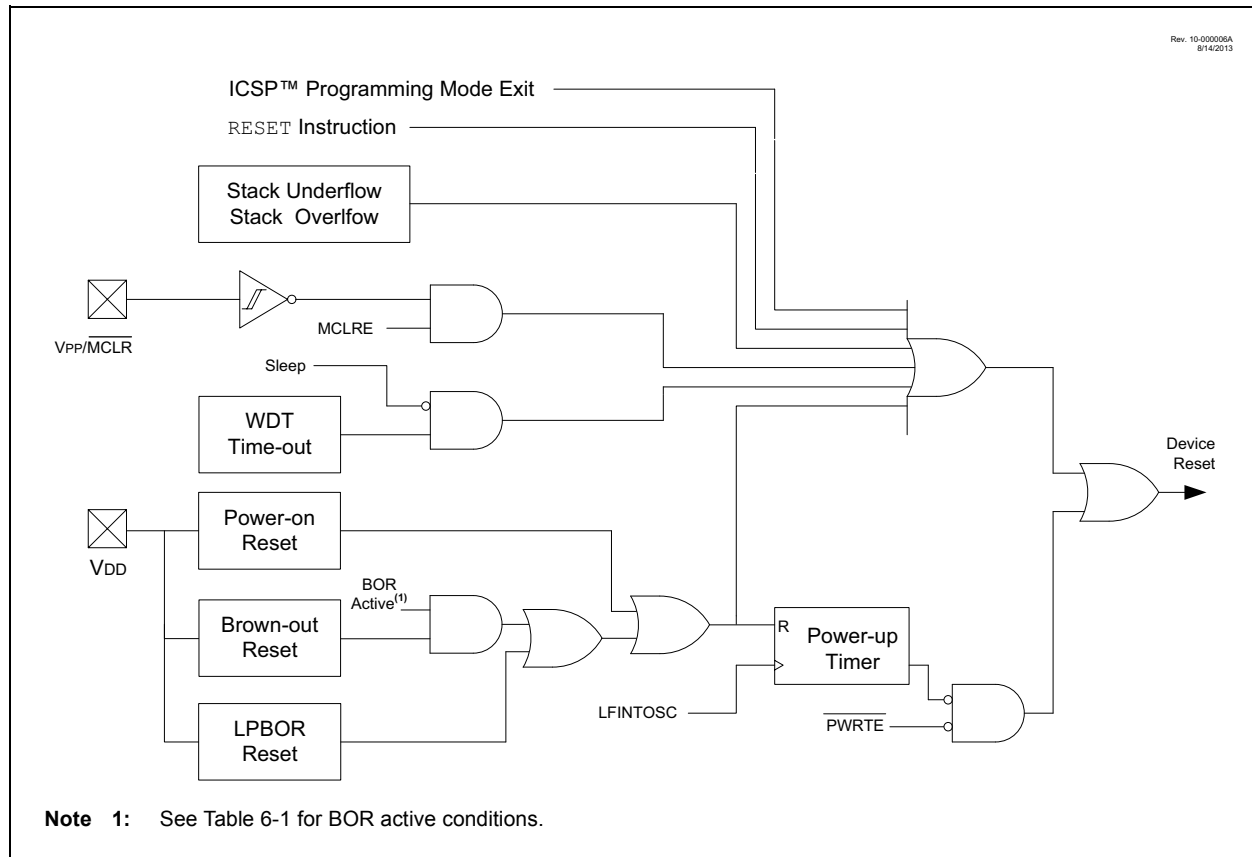
There are multiple ways to reset this device:

- Power-On Reset (POR)
- Brown-Out Reset (BOR)
- Low-Power Brown-Out Reset (LPBOR)
- MCLR Reset
- WDT Reset
- RESET instruction
- Stack Overflow
- Stack Underflow
- Programming mode exit

To allow VDD to stabilize, an optional Power-up Timer can be enabled to extend the Reset time after a BOR or POR event.

A simplified block diagram of the On-chip Reset Circuit is shown in Figure 6-1.

FIGURE 6-1: SIMPLIFIED BLOCK DIAGRAM OF ON-CHIP RESET CIRCUIT



PIC16(L)F1574/5/8/9

TABLE 6-5: SUMMARY OF REGISTERS ASSOCIATED WITH RESETS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
BORCON	SBOREN	BORFS	—	—	—	—	—	BORRDY	75
PCON	STKOVF	STKUNF	—	RWD \overline{T}	RMCLR	RI	POR	BOR	79
STATUS	—	—	—	TO	PD	Z	DC	C	23
WDTCON	—	—	WDTPS<4:0>					SWDTEN	99

Legend: — = unimplemented bit, reads as '0'. Shaded cells are not used by Resets.

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

TABLE 6-6: SUMMARY OF CONFIGURATION WORD WITH RESETS

Name	Bits	Bit -/7	Bit -/6	Bit 13/5	Bit 12/4	Bit 11/3	Bit 10/2	Bit 9/1	Bit 8/0	Register on Page
CONFIG1	13:8	—	—	—	—	CLKOUTEN	BOREN<1:0>		—	56
	7:0	CP	MCLRE	PWRTE	WDTE<1:0>		—	FOSC<1:0>		
CONFIG2	13:8	—	—	LVP	DEBUG	LPBOREN	BORV	STVREN	PLEN	57
	7:0	—	—	—	—	—	PPS1WAY	WRT<1:0>		

Legend: — = unimplemented location, read as '0'. Shaded cells are not used by Resets.

PIC16(L)F1574/5/8/9

REGISTER 7-5: PIR1: PERIPHERAL INTERRUPT REQUEST REGISTER 1

R/W-0/0	R/W-0/0	R-0/0	R-0/0	U-0	U-0	R/W-0/0	R/W-0/0
TMR1GIF	ADIF	RCIF	TXIF	—	—	TMR2IF	TMR1IF
bit 7							bit 0

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
 u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
 '1' = Bit is set '0' = Bit is cleared

- bit 7 **TMR1GIF:** Timer1 Gate Interrupt Flag bit
 1 = Interrupt is pending
 0 = Interrupt is not pending
 bit 6 **ADIF:** ADC Interrupt Flag bit
 1 = Interrupt is pending
 0 = Interrupt is not pending
 bit 5 **RCIF:** USART Receive Interrupt Flag bit
 1 = Interrupt is pending
 0 = Interrupt is not pending
 bit 4 **TXIF:** USART Transmit Interrupt Flag bit
 1 = Interrupt is pending
 0 = Interrupt is not pending
 bit 3-2 **Unimplemented:** Read as '0'
 bit 1 **TMR2IF:** Timer2 to PR2 Interrupt Flag bit
 1 = Interrupt is pending
 0 = Interrupt is not pending
 bit 0 **TMR1IF:** Timer1 Overflow Interrupt Flag bit
 1 = Interrupt is pending
 0 = Interrupt is not pending

Note: Interrupt flag bits are set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the Global Interrupt Enable bit, GIE of the INTCON register. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt. The USART RCIF and TXIF bits are read-only.

REGISTER 11-12: ANSELB: PORTB ANALOG SELECT REGISTER

U-0	U-0	R/W-1/1	R/W-1/1	U-0	U-0	U-0	U-0
—	—	ANSB5	ANSB4	—	—	—	—
bit 7				bit 0			

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

u = Bit is unchanged

x = Bit is unknown

-n/n = Value at POR and BOR/Value at all other Resets

'1' = Bit is set

'0' = Bit is cleared

bit 7-6 **Unimplemented:** Read as '0'

bit 5-4 **ANSB<5:4>:** Analog Select between Analog or Digital Function on pins RB<5:4>, respectively
 0 = Digital I/O. Pin is assigned to port or digital special function.
 1 = Analog input. Pin is assigned as analog input⁽¹⁾. Digital input buffer disabled.

bit 3-0 **Unimplemented:** Read as '0'

Note 1: When setting a pin to an analog input, the corresponding TRIS bit must be set to Input mode in order to allow external control of the voltage on the pin.

REGISTER 11-13: WPUB: WEAK PULL-UP PORTB REGISTER

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	U-0	U-0	U-0	U-0
WPUB7	WPUB6	WPUB5	WPUB4	—	—	—	—
bit 7				bit 0			

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

u = Bit is unchanged

x = Bit is unknown

-n/n = Value at POR and BOR/Value at all other Resets

'1' = Bit is set

'0' = Bit is cleared

bit 7-4 **WPUB<7:4>:** Weak Pull-up Register bits
 1 = Pull-up enabled
 0 = Pull-up disabled

bit 3-0 **Unimplemented:** Read as '0'

Note 1: Global $\overline{\text{WPUEN}}$ bit of the OPTION_REG register must be cleared for individual pull-ups to be enabled.

2: The weak pull-up device is automatically disabled if the pin is configured as an output.

PIC16(L)F1574/5/8/9

17.6 Register Definitions: DAC Control

REGISTER 17-1: DACCON0: VOLTAGE REFERENCE CONTROL REGISTER 0

R/W-0/0	U-0	R/W-0/0	U-0	R/W-0/0	R/W-0/0	U-0	U-0
DACEN	—	DACOE	—	DACPSS<1:0>		—	—
bit 7				bit 0			

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7 **DACEN:** DAC Enable bit
1 = DAC is enabled
0 = DAC is disabled

bit 6 **Unimplemented:** Read as '0'

bit 5 **DACOE:** DAC Voltage Output Enable bit
1 = DAC voltage level is output on the DACOUT1 pin
0 = DAC voltage level is disconnected from the DACOUT1 pin

bit 4 **Unimplemented:** Read as '0'

bit 3-2 **DACPSS<1:0>:** DAC Positive Source Select bits
11 = Reserved
10 = FVR_buffer2
01 = VREF+ pin
00 = VDD

bit 1-0 **Unimplemented:** Read as '0'

REGISTER 17-2: DACCON1: VOLTAGE REFERENCE CONTROL REGISTER 1

U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	DACR<4:0>				
bit 7			bit 0				

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-5 **Unimplemented:** Read as '0'

bit 4-0 **DACR<4:0>:** DAC Voltage Output Select bits

TABLE 17-1: SUMMARY OF REGISTERS ASSOCIATED WITH THE DAC MODULE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
DACCON0	DACEN	—	DACOE	—	DACPSS<1:0>		—	—	168
DACCON1	—	—	—	DACR<4:0>					168

Legend: — = Unimplemented location, read as '0'. Shaded cells are not used with the DAC module.

19.1.3 SOFTWARE PROGRAMMABLE PRESCALER

A software programmable prescaler is available for exclusive use with Timer0. The prescaler is enabled by clearing the PSA bit of the OPTION_REG register.

Note: The Watchdog Timer (WDT) uses its own independent prescaler.

There are eight prescaler options for the Timer0 module ranging from 1:2 to 1:256. The prescale values are selectable via the PS<2:0> bits of the OPTION_REG register. In order to have a 1:1 prescaler value for the Timer0 module, the prescaler must be disabled by setting the PSA bit of the OPTION_REG register.

The prescaler is not readable or writable. All instructions writing to the TMR0 register will clear the prescaler.

19.1.4 TIMER0 INTERRUPT

Timer0 will generate an interrupt when the TMR0 register overflows from FFh to 00h. The TMR0IF interrupt flag bit of the INTCON register is set every time the TMR0 register overflows, regardless of whether or not the Timer0 interrupt is enabled. The TMR0IF bit can only be cleared in software. The Timer0 interrupt enable is the TMR0IE bit of the INTCON register.

Note: The Timer0 interrupt cannot wake the processor from Sleep since the timer is frozen during Sleep.

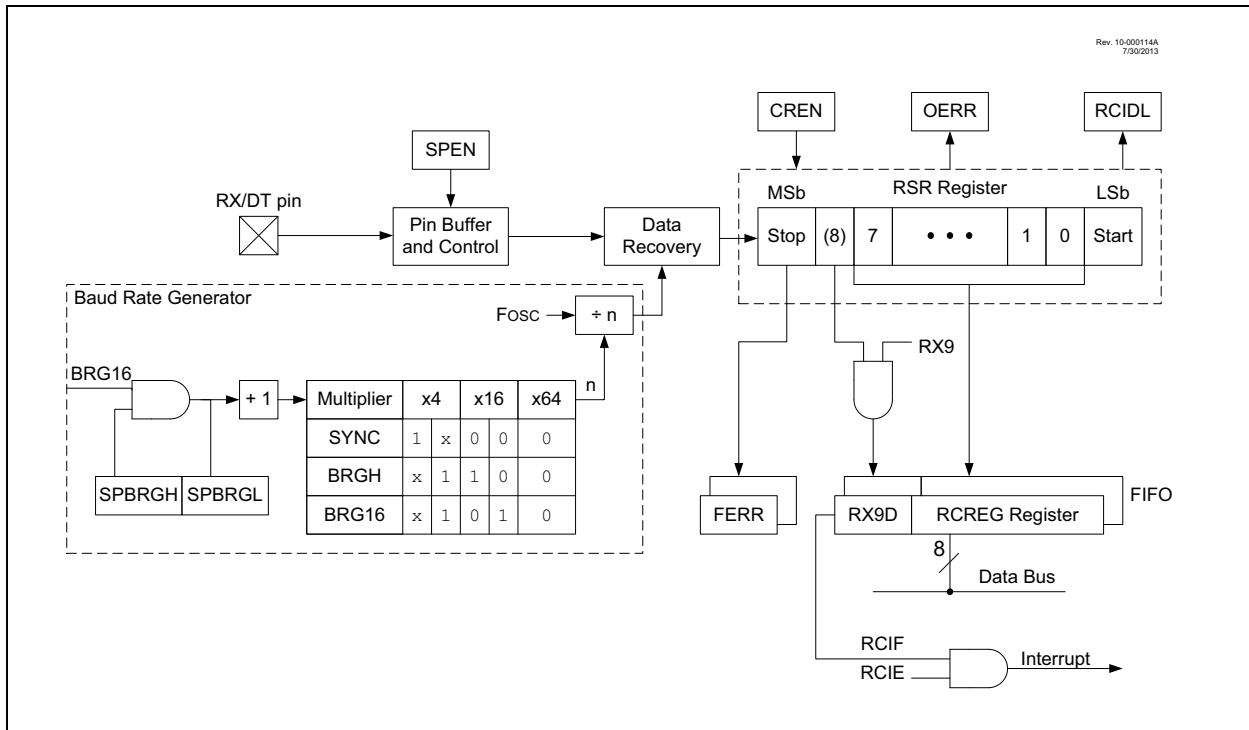
19.1.5 8-BIT COUNTER MODE SYNCHRONIZATION

When in 8-Bit Counter mode, the incrementing edge on the T0CKI pin must be synchronized to the instruction clock. Synchronization can be accomplished by sampling the prescaler output on the Q2 and Q4 cycles of the instruction clock. The high and low periods of the external clocking source must meet the timing requirements as shown in **Section 27.0 “Electrical Specifications”**.

19.1.6 OPERATION DURING SLEEP

Timer0 cannot operate while the processor is in Sleep mode. The contents of the TMR0 register will remain unchanged while the processor is in Sleep mode.

FIGURE 22-2: EUSART RECEIVE BLOCK DIAGRAM



The operation of the EUSART module is controlled through three registers:

- Transmit Status and Control (TXSTA)
- Receive Status and Control (RCSTA)
- Baud Rate Control (BAUDCON)

These registers are detailed in Register 22-1, Register 22-2 and Register 22-3, respectively.

When the receiver or transmitter section is not enabled then the corresponding RX or TX pin may be used for general purpose input and output.

TABLE 22-5: BAUD RATES FOR ASYNCHRONOUS MODES (CONTINUED)

BAUD RATE	SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1											
	Fosc = 20.000 MHz			Fosc = 18.432 MHz			Fosc = 16.000 MHz			Fosc = 11.0592 MHz		
	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)
300	300.0	0.00	16665	300.0	0.00	15359	300.0	0.00	13332	300.0	0.00	9215
1200	1200	-0.01	4166	1200	0.00	3839	1200.1	0.01	3332	1200	0.00	2303
2400	2400	0.02	2082	2400	0.00	1919	2399.5	-0.02	1666	2400	0.00	1151
9600	9597	-0.03	520	9600	0.00	479	9592	-0.08	416	9600	0.00	287
10417	10417	0.00	479	10425	0.08	441	10417	0.00	383	10433	0.16	264
19.2k	19.23k	0.16	259	19.20k	0.00	239	19.23k	0.16	207	19.20k	0.00	143
57.6k	57.47k	-0.22	86	57.60k	0.00	79	57.97k	0.64	68	57.60k	0.00	47
115.2k	116.3k	0.94	42	115.2k	0.00	39	114.29k	-0.79	34	115.2k	0.00	23

BAUD RATE	SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1											
	Fosc = 8.000 MHz			Fosc = 4.000 MHz			Fosc = 3.6864 MHz			Fosc = 1.000 MHz		
	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)
300	300.0	0.00	6666	300.0	0.01	3332	300.0	0.00	3071	300.1	0.04	832
1200	1200	-0.02	1666	1200	0.04	832	1200	0.00	767	1202	0.16	207
2400	2401	0.04	832	2398	0.08	416	2400	0.00	383	2404	0.16	103
9600	9615	0.16	207	9615	0.16	103	9600	0.00	95	9615	0.16	25
10417	10417	0	191	10417	0.00	95	10473	0.53	87	10417	0.00	23
19.2k	19.23k	0.16	103	19.23k	0.16	51	19.20k	0.00	47	19.23k	0.16	12
57.6k	57.14k	-0.79	34	58.82k	2.12	16	57.60k	0.00	15	—	—	—
115.2k	117.6k	2.12	16	111.1k	-3.55	8	115.2k	0.00	7	—	—	—

23.2 PWM Modes

PWM Modes are selected with MODE<1:0> bits of the PWMxCON register (Register 23-1).

In all PWM modes an offset match event can also be used to synchronize the PWMxTMR in three offset modes. See **Section 23.3 “Offset Modes”** for more information.

23.2.1 STANDARD MODE

The Standard mode (MODE = 00) selects a single phase PWM output. The PWM output in this mode is determined by when the period, duty cycle, and phase counts match the PWMxTMR value. The start of the duty cycle occurs on the phase match and the end of the duty cycle occurs on the duty cycle match. The period match resets the timer. The offset match can also be used to synchronize the PWMxTMR in the offset modes. See **Section 23.3 “Offset Modes”** for more information.

Equation 23-1 is used to calculate the PWM period in Standard mode.

Equation 23-2 is used to calculate the PWM duty-cycle ratio in Standard mode.

EQUATION 23-1: PWM PERIOD IN STANDARD MODE

$$Period = \frac{(PWMxPR + 1) \cdot Prescale}{PWMxCLK}$$

EQUATION 23-2: PWM DUTY CYCLE IN STANDARD MODE

$$Duty Cycle = \frac{(PWMxDC - PWMxPH)}{PWMxPR + 1}$$

A detailed timing diagram for Standard mode is shown in Figure 23-4.

23.2.2 SET ON MATCH MODE

The Set On Match mode (MODE = 01) generates an active output when the phase count matches the PWMxTMR value. The output stays active until the OUT bit of the PWMxCON register is cleared or the PWM module is disabled. The duty cycle count has no effect in this mode. The period count only determines the maximum PWMxTMR value above which no phase matches can occur.

The PWMxOUT bit can be used to set or clear the output of the PWM in this mode. Writes to this bit will take place on the next rising edge of the PWM_clock after the bit is written.

A detailed timing diagram for Set On Match is shown in Figure 23-5.

23.2.3 TOGGLE ON MATCH MODE

The Toggle On Match mode (MODE = 10) generates a 50% duty cycle PWM with a period twice as long as that computed for the standard PWM mode. Duty cycle count has no effect in this mode. The phase count determines how many PWMxTMR periods after a period event the output will toggle.

Writes to the OUT bit of the PWMxCON register will have no effect in this mode.

A detailed timing diagram for Toggle On Match is shown in Figure 23-6.

23.2.4 CENTER-ALIGNED MODE

The Center-Aligned mode (MODE = 11) generates a PWM waveform that is centered in the period. In this mode the period is two times the PWMxPR count. The PWMxTMR counts up to the period value then counts back down to 0. The duty cycle count determines both the start and end of the active PWM output. The start of the duty cycle occurs at the match event when PWMxTMR is incrementing and the duty cycle ends at the match event when PWMxTMR is decrementing. The incrementing match value is the period count minus the duty cycle count. The decrementing match value is the incrementing match value plus 1.

Equation 23-3 is used to calculate the PWM period in Center-Aligned mode.

EQUATION 23-3: PWM PERIOD IN CENTER-ALIGNED MODE

$$Period = \frac{(PWMxPR + 1) \cdot Prescale \cdot 2}{PWMxCLK}$$

Equation 23-4 is used to calculate the PWM duty cycle ratio in Center-Aligned mode

EQUATION 23-4: PWM DUTY CYCLE IN CENTER-ALIGNED MODE

$$Duty Cycle = \frac{PWMxDC \cdot 2}{(PWMxPR + 1) \cdot 2}$$

Writes to PWMxOUT will have no effect in this mode.

A detailed timing diagram for Center-Aligned mode is shown in Figure 23-7.

BCF	Bit Clear f
Syntax:	[<i>label</i>] BCF f,b
Operands:	$0 \leq f \leq 127$ $0 \leq b \leq 7$
Operation:	$0 \rightarrow (f < b)$
Status Affected:	None
Description:	Bit 'b' in register 'f' is cleared.

BTFSC	Bit Test f, Skip if Clear
Syntax:	[<i>label</i>] BTFSC f,b
Operands:	$0 \leq f \leq 127$ $0 \leq b \leq 7$
Operation:	skip if $(f < b) = 0$
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.

BRA	Relative Branch
Syntax:	[<i>label</i>] BRA label [<i>label</i>] BRA \$+k
Operands:	$-256 \leq \text{label} - \text{PC} + 1 \leq 255$ $-256 \leq k \leq 255$
Operation:	$(\text{PC}) + 1 + k \rightarrow \text{PC}$
Status Affected:	None
Description:	Add the signed 9-bit literal 'k' to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be $\text{PC} + 1 + k$. This instruction is a 2-cycle instruction. This branch has a limited range.

BTFSS	Bit Test f, Skip if Set
Syntax:	[<i>label</i>] BTFSS f,b
Operands:	$0 \leq f \leq 127$ $0 \leq b < 7$
Operation:	skip if $(f < b) = 1$
Status Affected:	None
Description:	If bit 'b' in register 'f' is '0', the next instruction is executed. If bit 'b' is '1', then the next instruction is discarded and a NOP is executed instead, making this a 2-cycle instruction.

BRW	Relative Branch with W
Syntax:	[<i>label</i>] BRW
Operands:	None
Operation:	$(\text{PC}) + (W) \rightarrow \text{PC}$
Status Affected:	None
Description:	Add the contents of W (unsigned) to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be $\text{PC} + 1 + (W)$. This instruction is a 2-cycle instruction.

BSF	Bit Set f
Syntax:	[<i>label</i>] BSF f,b
Operands:	$0 \leq f \leq 127$ $0 \leq b \leq 7$
Operation:	$1 \rightarrow (f < b)$
Status Affected:	None
Description:	Bit 'b' in register 'f' is set.

PIC16(L)F1574/5/8/9

TABLE 27-5: MEMORY PROGRAMMING SPECIFICATIONS

Standard Operating Conditions (unless otherwise stated)							
Param. No.	Sym.	Characteristic	Min.	Typ†	Max.	Units	Conditions
Program Memory Programming Specifications							
D110	VIHH	Voltage on $\overline{\text{MCLR}}$ /VPP pin	8.0	—	9.0	V	(Note 2)
D111	IDDP	Supply Current during Programming	—	—	10	mA	
D112	VBE	VDD for Bulk Erase	2.7	—	VDDMAX	V	
D113	VPEW	VDD for Write or Row Erase	VDDMIN	—	VDDMAX	V	
D114	IPPPGM	Current on $\overline{\text{MCLR}}$ /VPP during Erase/Write	—	1.0	—	mA	
D115	IDDPGM	Current on VDD during Erase/Write	—	5.0	—	mA	
Program Flash Memory							
D121	EP	Cell Endurance	10K	—	—	E/W	-40°C ≤ TA ≤ +85°C (Note 1)
D122	VPRW	VDD for Read/Write	VDDMIN	—	VDDMAX	V	
D123	TIW	Self-timed Write Cycle Time	—	2	2.5	ms	Provided no other specifications are violated
D124	TRETD	Characteristic Retention	—	40	—	Year	
D125	EHEFC	High-Endurance Flash Cell	100K	—	—	E/W	0°C ≤ TA ≤ +60°C, lower byte last 128 addresses

† Data in “Typ” column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

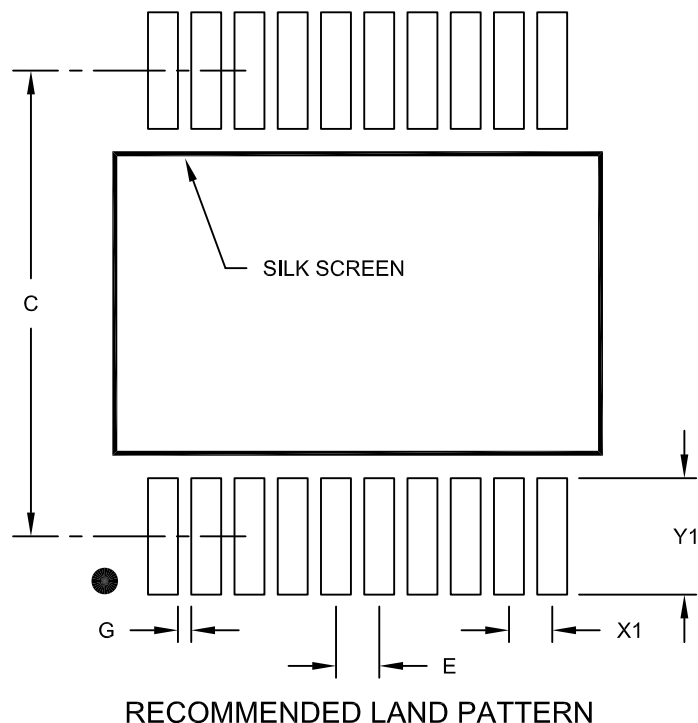
Note 1: Self-write and Block Erase.

2: Required only if single-supply programming is disabled.

PIC16(L)F1574/5/8/9

20-Lead Plastic Shrink Small Outline (SS) - 5.30 mm Body [SSOP]

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



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E	0.65 BSC		
Contact Pad Spacing	C		7.20	
Contact Pad Width (X20)	X1			0.45
Contact Pad Length (X20)	Y1			1.75
Distance Between Pads	G	0.20		

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

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

Microchip Technology Drawing No. C04-2072A

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

Trademarks

The Microchip name and logo, the Microchip logo, dsPIC, FlashFlex, flexPWR, JukeBlox, KEELOQ, KEELOQ logo, Klear, LANCheck, MediaLB, MOST, MOST logo, MPLAB, OptoLyzer, PIC, PICSTART, PIC³² logo, RightTouch, SpyNIC, SST, SST Logo, SuperFlash and UNI/O are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

The Embedded Control Solutions Company and mTouch are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Analog-for-the-Digital Age, BodyCom, chipKIT, chipKIT logo, CodeGuard, dsPICDEM, dsPICDEM.net, ECAN, In-Circuit Serial Programming, ICSP, Inter-Chip Connectivity, KlearNet, KlearNet logo, MiWi, motorBench, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, RightTouch logo, REAL ICE, SQI, Serial Quad I/O, Total Endurance, TSHARC, USBCheck, VariSense, ViewSpan, WiperLock, Wireless DNA, and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

Silicon Storage Technology is a registered trademark of Microchip Technology Inc. in other countries.

GestIC is a registered trademark of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

All other trademarks mentioned herein are property of their respective companies.

© 2016, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

ISBN: 978-1-5224-0190-2

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
= ISO/TS 16949 =

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.