



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	Obsolete
Core Processor	PIC
Core Size	8-Bit
Speed	4MHz
Connectivity	-
Peripherals	Brown-out Detect/Reset, POR, WDT
Number of I/O	12
Program Memory Size	768B (512 x 12)
Program Memory Type	OTP
EEPROM Size	-
RAM Size	25 x 8
Voltage - Supply (Vcc/Vdd)	3.5V ~ 15V
Data Converters	-
Oscillator Type	External
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Surface Mount
Package / Case	20-SSOP (0.209", 5.30mm Width)
Supplier Device Package	20-SSOP
Purchase URL	<a href="https://www.e-xfl.com/product-detail/microchip-technology/pic16hv540t-04-ss">https://www.e-xfl.com/product-detail/microchip-technology/pic16hv540t-04-ss</a>

3.1 Clocking Scheme/Instruction Cycle

The clock input (OSC1/CLKIN pin) is internally divided by four to generate four non-overlapping quadrature clocks namely Q1, Q2, Q3 and Q4. Internally, the program counter is incremented every Q1, and the instruction is fetched from program memory and latched into instruction register in Q4. It is decoded and executed during the following Q1 through Q4. The clocks and instruction execution flow is shown in Figure 3-2 and Example 3-1.

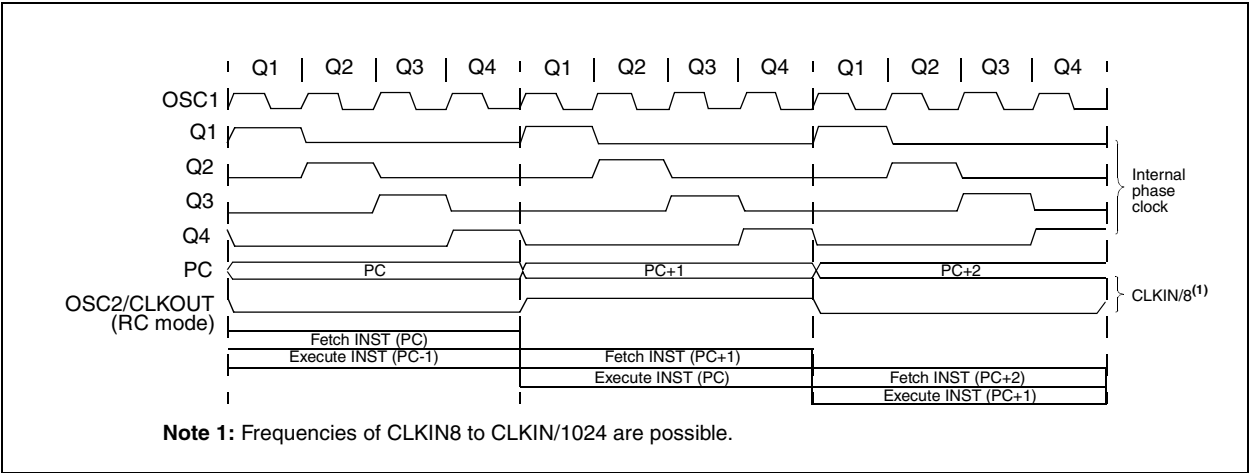
3.2 Instruction Flow/Pipelining

An Instruction Cycle consists of four Q cycles (Q1, Q2, Q3 and Q4). The instruction fetch and execute are pipelined such that fetch takes one instruction cycle while decode and execute takes another instruction cycle. However, due to the pipelining, each instruction effectively executes in one cycle. If an instruction causes the program counter to change (e.g., GOTO) then two cycles are required to complete the instruction (Example 3-1).

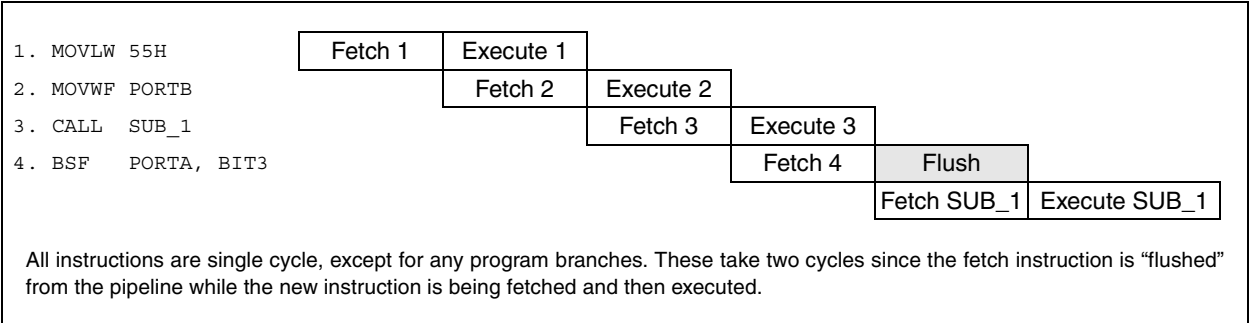
A fetch cycle begins with the program counter (PC) incrementing in Q1.

In the execution cycle, the fetched instruction is latched into the Instruction Register (IR) in cycle Q1. This instruction is then decoded and executed during the Q2, Q3, and Q4 cycles. Data memory is read during Q2 (operand read) and written during Q4 (destination write).

FIGURE 3-2: CLOCK/INSTRUCTION CYCLE



EXAMPLE 3-1: INSTRUCTION PIPELINE FLOW



## 4.0 MEMORY ORGANIZATION

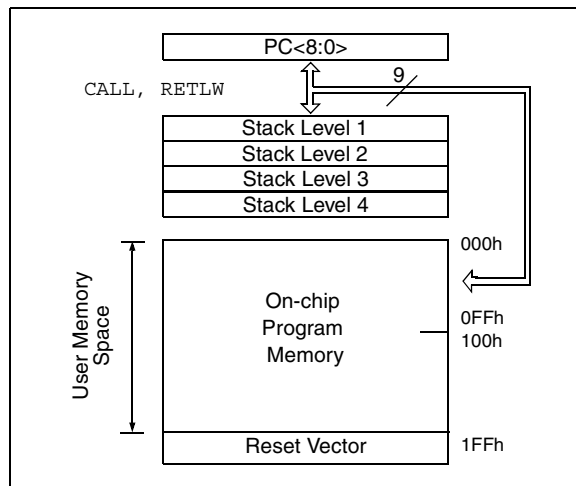
PIC16HV540 memory is organized into program memory and data memory. For devices with more than 512 bytes of program memory, a paging scheme is used. Program memory pages are accessed using one or two STATUS register bits. For devices with a data memory register file of more than 32 registers, a banking scheme is used. Data memory banks are accessed using the File Selection Register (FSR).

### 4.1 Program Memory Organization

The PIC16HV540 has a 9-bit Program Counter (PC) capable of addressing a 512 x 12 program memory space (Figure 4-1). Accessing a location above the physically implemented address will cause a wrap-around.

The reset vector for the PIC16HV540 is at 1FFh. A NOP at the reset vector location will cause a restart at location 000h.

**FIGURE 4-1: PIC16HV540 PROGRAM MEMORY MAP AND STACK**



### 4.2 Data Memory Organization

Data memory is composed of registers, or bytes of RAM. Therefore, data memory for a device is specified by its register file. The register file is divided into two functional groups: special function registers and general purpose registers.

The special function registers include the TMR0 register, the Program Counter (PC), the Status Register, the I/O registers (ports), and the File Select Register (FSR). In addition, special purpose registers are used to control the I/O port configuration and prescaler options.

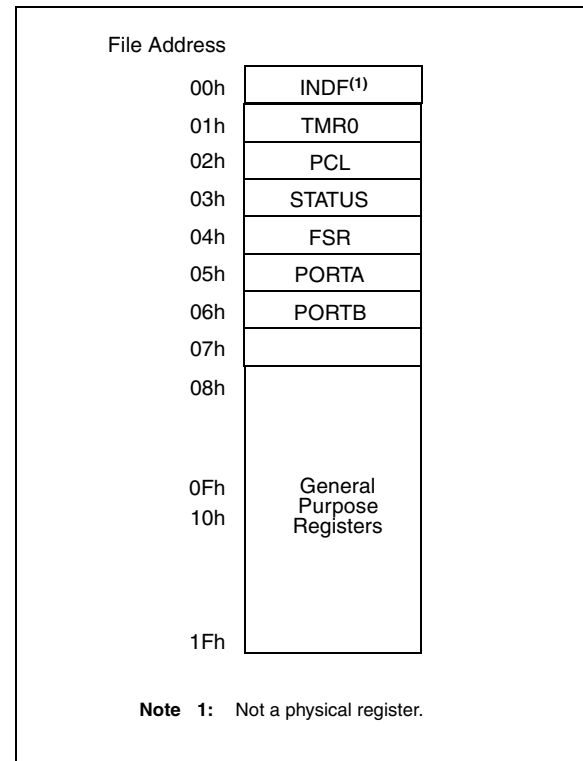
The general purpose registers are used for data and control information under command of the instructions.

For the PIC16HV540, the register file is composed of 10 special function registers and 25 general purpose registers (Figure 4-2).

### 4.2.1 GENERAL PURPOSE REGISTER FILE

The register file is accessed either directly or indirectly through the file select register FSR (Section 4.8).

**FIGURE 4-2: PIC16HV540 REGISTER FILE MAP**



### 4.2.2 SPECIAL FUNCTION REGISTERS

The Special Function Registers are registers used by the CPU and peripheral functions to control the operation of the device (Table 4-1).

The special registers can be classified into two sets. The special function registers associated with the "core" functions are described in this section. Those related to the operation of the peripheral features are described in the section for each peripheral feature.

**TABLE 4-1: SPECIAL FUNCTION REGISTER SUMMARY**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-On Reset	Value on MCLR and WDT Reset	Value on Wake-up on Pin Change	Value on Brown-Out Reset
N/A	TRIS	I/O control registers (TRISA, TRISB)								1111 1111	1111 1111	1111 1111	1111 1111
N/A	OPTION	Contains control bits to configure Timer0 and Timer0/WDT prescaler								--11 1111	--11 1111	--11 1111	--11 1111
N/A	OPTION2	Contains control bits to configure pin changes, software enabled WDT, regulation and brown-out								--11 1111	--uu uuuu	--uu uuuu	--xx xxxx
00h	INDF	Uses contents of FSR to address data memory (not a physical register)								xxxx xxxx	uuuu uuuu	uuuu uuuu	xxxx xxxx
01h	TMR0	8-bit real-time clock/counter								xxxx xxxx	uuuu uuuu	uuuu uuuu	xxxx xxxx
02h <sup>(1)</sup>	PCL	Low order 8 bits of PC								1111 1111	1111 1111	1111 1111	1111 1111
03h	STATUS	PCWUF	PA1	PA0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	1001 1xxx	100q quuu	000u uuuu	x00x xxxx
04h	FSR	Indirect data memory address pointer								111x xxxx	111u uuuu	111u uuuu	111x xxxx
05h	PORTA	—	—	—	—	RA3	RA2	RA1	RA0	---- xxxx	---- uuuu	---- uuuu	---- xxxx
06h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx	uuuu uuuu	uuuu uuuu	xxxx xxxx

Legend: Shaded boxes = unimplemented or unused, — = unimplemented, read as '0' (if applicable)

x = unknown, u = unchanged, q = value depends on condition.

**Note 1:** The upper byte of the Program Counter is not directly accessible. See Section 4.6 of the PIC16HV540 data sheet (DS40197B) for an explanation of how to access these bits.

## 4.4 OPTION Register

The OPTION register is a 6-bit wide, write-only register which contains various control bits to configure the Timer0/WDT prescaler and Timer0.

By executing the OPTION instruction, the contents of the W register will be transferred to the OPTION register. A RESET sets the OPTION<5:0> bits.

Example 4-1 illustrates how to initialize the OPTION register.

### EXAMPLE 4-1: INSTRUCTIONS FOR INITIALIZING OPTION REGISTER

```
movlw    '0000 0111'b    ; load OPTION setup value into W
OPTION                    ; initialize OPTION register
```

### REGISTER 4-2: OPTION REGISTER

U-0	U-0	W-1	W-1	W-1	W-1	W-1	W-1
—	—	T0CS	T0SE	PSA	PS2	PS1	PS0
bit7							0

W = Writable bit  
U = Unimplemented bit  
- n = Value at POR reset

bit 7-6: **Unimplemented**

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

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

bit 3: **PSA:** Prescaler Assignment bit  
1 = Prescaler assigned to the WDT  
0 = Prescaler assigned to Timer0

bit 2-0: **PS<2:0>:** Prescaler Rate Select bits

Bit Value	Timer0 Rate	WDT Rate
000	1 : 2	1 : 1
001	1 : 4	1 : 2
010	1 : 8	1 : 4
011	1 : 16	1 : 8
100	1 : 32	1 : 16
101	1 : 64	1 : 32
110	1 : 128	1 : 64
111	1 : 256	1 : 128

## 4.5 OPTION2 Register

The OPTION2 register is a 6-bit wide, write-only register which contains various control bits to configure the added features on the PIC16HV540. A Power-on Reset sets the OPTION2<5:0> bits.

Example 4-2 illustrates how to initialize the OPTION2 register.

**Note:** All Power-on Resets will disable the Brown-out Detect circuit. All subsequent resets will not disable the Brown-out Detect if enabled.

### EXAMPLE 4-2: INSTRUCTIONS FOR INITIALIZING OPTION2 REGISTER

```
movlw    '0001 0111'b    ; load OPTION2 setup value into W
tris     0x07             ; initialize OPTION2 register
```

### REGISTER 4-3: OPTION2 REGISTER (TRIS 07H)

U-0	U-0	W-1	W-1	W-1	W-1	W-1	W-1
—	—	PCWU	SWDTEN	RL	SL	BODL	BODEN
bit7							0

W = Writable bit  
 U = Unimplemented bit  
 - n = Value at POR reset

bit 7-6: **Unimplemented**

bit 5: **PCWU**: Wake-up on Pin Change  
 1 = Disabled  
 0 = Enabled

bit 4: **SWDTEN**: Software Controlled WDT Enable bit  
 1 = WDT is turned off if the WDTE configuration bit = 0  
 0 = WDT is on if the WDTE configuration bit = 0; if WDTE bit = 1, then SWDTEN is 'don't care'

bit 3: **RL**: Regulated Voltage Level Select bit  
 1 = 5 volt  
 0 = 3 volt

bit 2: **SL**: Sleep Voltage Level Select bit  
 1 = **RL** bit setting  
 0 = 3 volt

bit 1: **BODL**: Brown-out Voltage Level Select bit  
 1 = **RL** bit setting, but **SL** during SLEEP  
 0 = 3 volt

bit 0: **BODEN**: Brown-out Enabled  
 1 = Disabled  
 0 = Enabled

## 4.6 Program Counter

As a program instruction is executed, the Program Counter (PC) will contain the address of the next program instruction to be executed. The PC value is increased by one every instruction cycle, unless an instruction changes the PC.

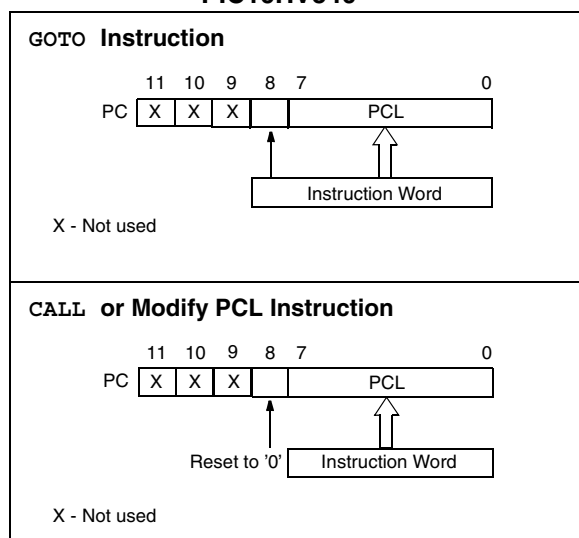
For a **GOTO** instruction, bits 8:0 of the PC are provided by the **GOTO** instruction word. (Figure 4-3).

For a **CALL** instruction, or any instruction where the PCL is the destination, bits 7:0 of the PC again are provided by the instruction word. However, PC<8> does not come from the instruction word, but is always cleared (Figure 4-3).

Instructions where the PCL is the destination, or Modify PCL instructions, include **MOVWF PC**, **ADDWF PC**, and **BSF PC, 5**.

**Note:** Because PC<8> is cleared in the **CALL** instruction, or any Modify PCL instruction, all subroutine calls or computed jumps are limited to the first 256 locations of any program memory page (512 words long).

**FIGURE 4-3: LOADING OF PC BRANCH INSTRUCTIONS - PIC16HV540**



### 4.6.1 EFFECTS OF RESET

The Program Counter is set upon a **RESET**, which means that the PC addresses the last location in the last page i.e., the reset vector.

The **STATUS** register page preselect bits are cleared upon a **RESET**, which means that page 0 is pre-selected.

Therefore, upon a **RESET**, a **GOTO** instruction at the reset vector location will automatically cause the program to jump to page 0.

## 4.7 Stack

PIC16HV540 device has a 12-bit wide L.I.F.O. (last in, first out) hardware 4 level stack.

A **CALL** instruction will *push* the current value of stack 1 into stack 2 and then push the current program counter value, incremented by one, into stack level 1. If more than four sequential **CALL**'s are executed, only the most recent four return addresses are stored.

A **RETLW** instruction will *pop* the contents of stack level 1 into the program counter and then copy stack level 2 contents into level 1. If more than four sequential **RETLW**'s are executed, the stack will be filled with the address previously stored in level 4. Note that the W register will be loaded with the literal value specified in the instruction. This is particularly useful for the implementation of data look-up tables within the program memory.

Upon any reset, the contents of the stack remain unchanged, however the program counter (PCL) will also be reset to 0.

**Note 1:** There are no **STATUS** bits to indicate stack overflows or stack underflow conditions.

**Note 2:** There are no instructions mnemonics called **PUSH** or **POP**. These are actions that occur from the execution of the **CALL** and **RETLW** instructions.

## 4.8 Indirect Data Addressing: INDF and FSR Registers

The INDF register is not a physical register. Addressing INDF actually addresses the register whose address is contained in the FSR register (FSR is a *pointer*). This is indirect addressing.

### EXAMPLE 4-3: INDIRECT ADDRESSING

- Register file 05 contains the value 10h
- Register file 06 contains the value 0Ah
- Load the value 05 into the FSR register
- A read of the INDF register will return the value of 10h
- Increment the value of the FSR register by one (FSR = 06)
- A read of the INDR register now will return the value of 0Ah.

Reading INDF itself indirectly (FSR = 0) will produce 00h. Writing to the INDF register indirectly results in a no-operation (although STATUS bits may be affected).

A simple program to clear RAM locations 10h-1Fh using indirect addressing is shown in Example 4-4.

### EXAMPLE 4-4: HOW TO CLEAR RAM USING INDIRECT ADDRESSING

```

movlw 0x10 ;initialize pointer
movwf FSR ; to RAM
NEXT   clrf INDF ;clear INDF register
       incf FSR,F ;inc pointer
       btfsc FSR,4 ;all done?
       goto NEXT ;NO, clear next

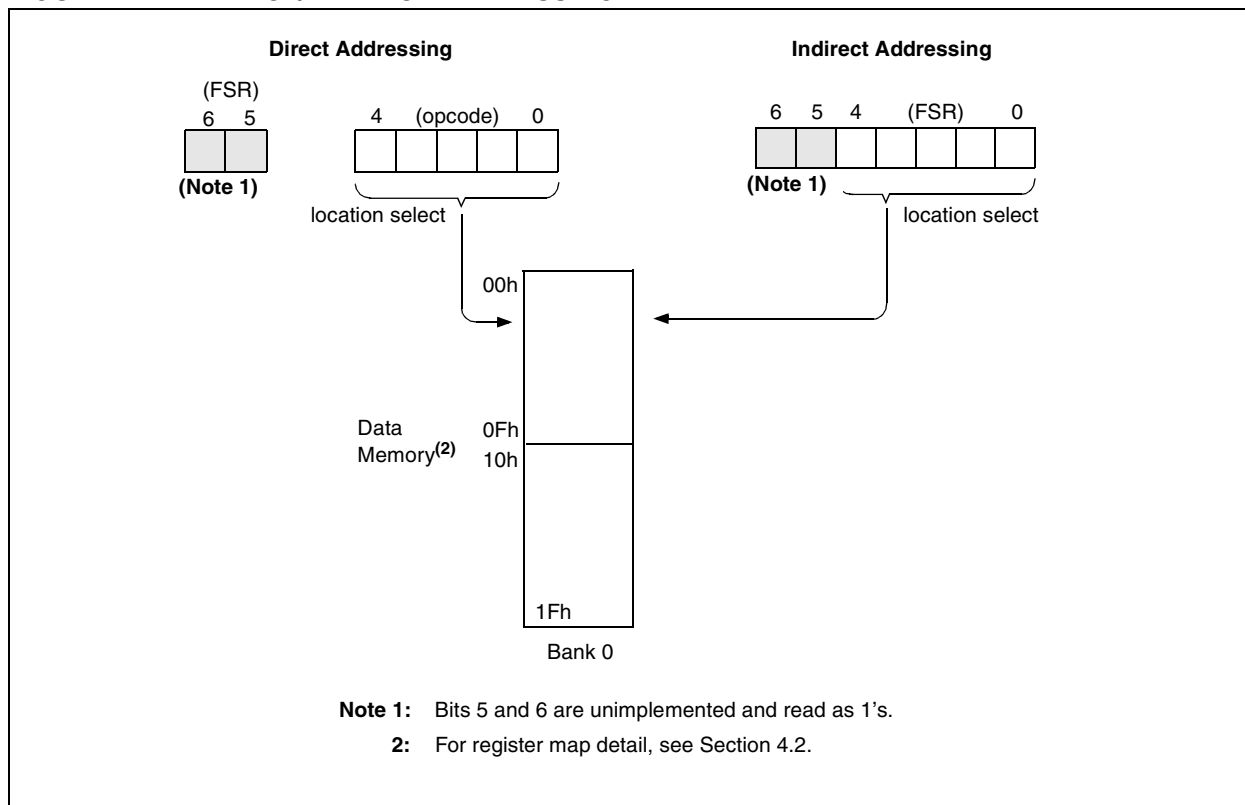
CONTINUE
:      ;YES, continue
    
```

The FSR is a 5-bit (PIC16HV540) wide register. It is used in conjunction with the INDF register to indirectly address the data memory area.

The FSR<4:0> bits are used to select data memory addresses 00h to 1Fh.

**PIC16HV540:** Do not use banking. FSR<6:5> are unimplemented and read as '1's.

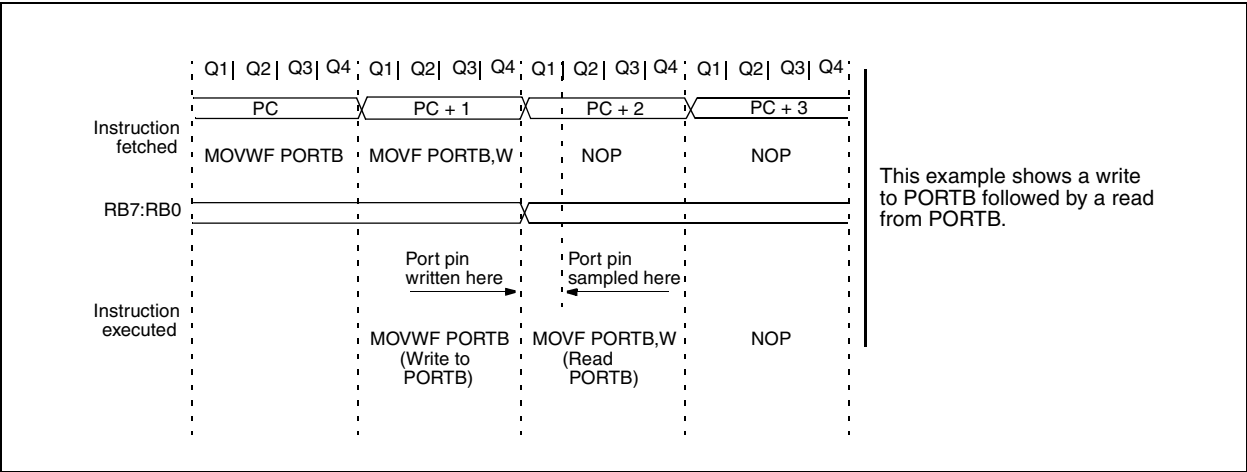
FIGURE 4-4: DIRECT/INDIRECT ADDRESSING





NOTES:

FIGURE 5-5: SUCCESSIVE I/O OPERATION



## 7.2 Oscillator Configurations

### 7.2.1 OSCILLATOR TYPES

The PIC16HV540 can be operated in four different oscillator modes. The user can program two configuration bits (FOSC1:FOSC0) to select one of these four modes:

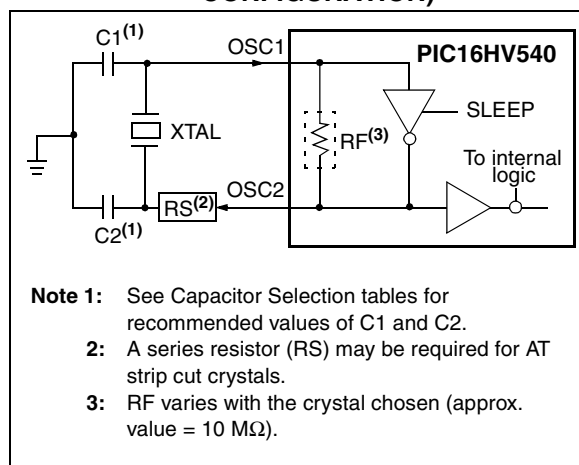
- LP: Low Power Crystal
- XT: Crystal/Resonator
- HS: High Speed Crystal/Resonator
- RC: Resistor/Capacitor

**Note:** Not all oscillator selections available for all parts. See Section 7.1.

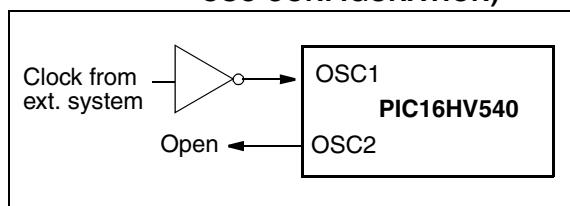
### 7.2.2 CRYSTAL OSCILLATOR / CERAMIC RESONATORS

In XT, LP or HS modes, a crystal or ceramic resonator is connected to the OSC1/CLKIN and OSC2/CLKOUT pins to establish oscillation (Figure 7-1). The PIC16HV540 oscillator design requires the use of a parallel cut crystal. Use of a series cut crystal may give a frequency out of the crystal manufacturers specifications. When in XT, LP or HS modes, the device can have an external clock source drive the OSC1/CLKIN pin (Figure 7-2).

**FIGURE 7-1: CRYSTAL OPERATION (OR CERAMIC RESONATOR) (HS, XT OR LP OSC CONFIGURATION)**



**FIGURE 7-2: EXTERNAL CLOCK INPUT OPERATION (HS, XT OR LP OSC CONFIGURATION)**



**TABLE 7-1: CAPACITOR SELECTION FOR CERAMIC RESONATORS - PIC16HV540**

Osc Type	Resonator Freq	Cap. Range C1	Cap. Range C2
XT	455 kHz	68-100 pF	68-100 pF
	2.0 MHz	15-33 pF	15-33 pF
	4.0 MHz	10-22 pF	10-22 pF
HS	8.0 MHz	10-22 pF	10-22 pF
	16.0 MHz	10 pF	10 pF

**Note:** These values are for design guidance only. Since each resonator has its own characteristics, the user should consult the resonator manufacturer for appropriate values of external components.

**TABLE 7-2: CAPACITOR SELECTION FOR CRYSTAL OSCILLATOR - PIC16HV540**

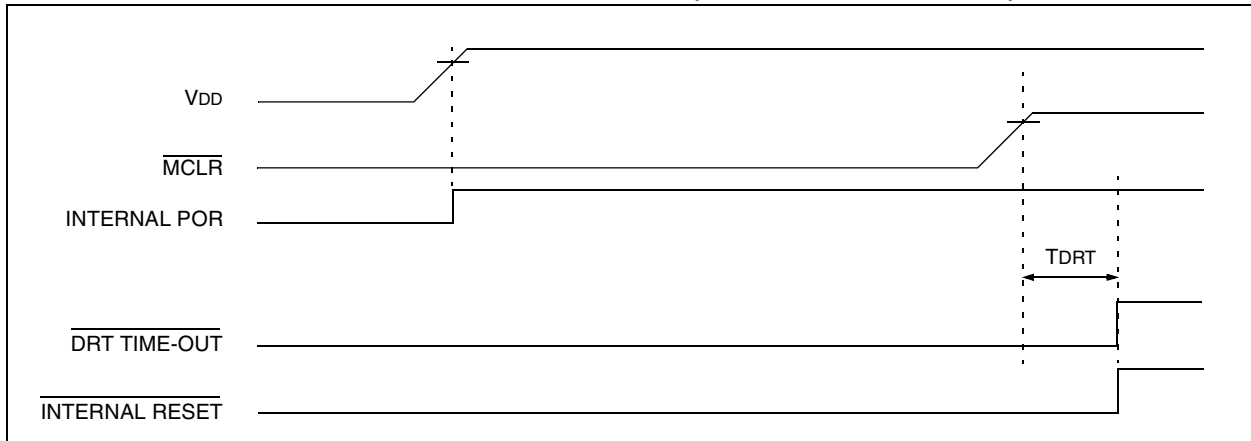
Osc Type	Resonator Freq	Cap.Range C1	Cap. Range C2
LP	32 kHz <sup>(1)</sup>	15 pF	15 pF
XT	100 kHz	15-30 pF	200-300 pF
	200 kHz	15-30 pF	100-200 pF
	455 kHz	15-30 pF	15-100 pF
	1 MHz	15-30 pF	15-30 pF
	2 MHz	15 pF	15 pF
	4 MHz	15 pF	15 pF
HS	4 MHz	15 pF	15 pF
	8 MHz	15 pF	15 pF
	20 MHz	15 pF	15 pF

**Note 1:** For VDD > 4.5V, C1 = C2 ≈ 30 pF is recommended.

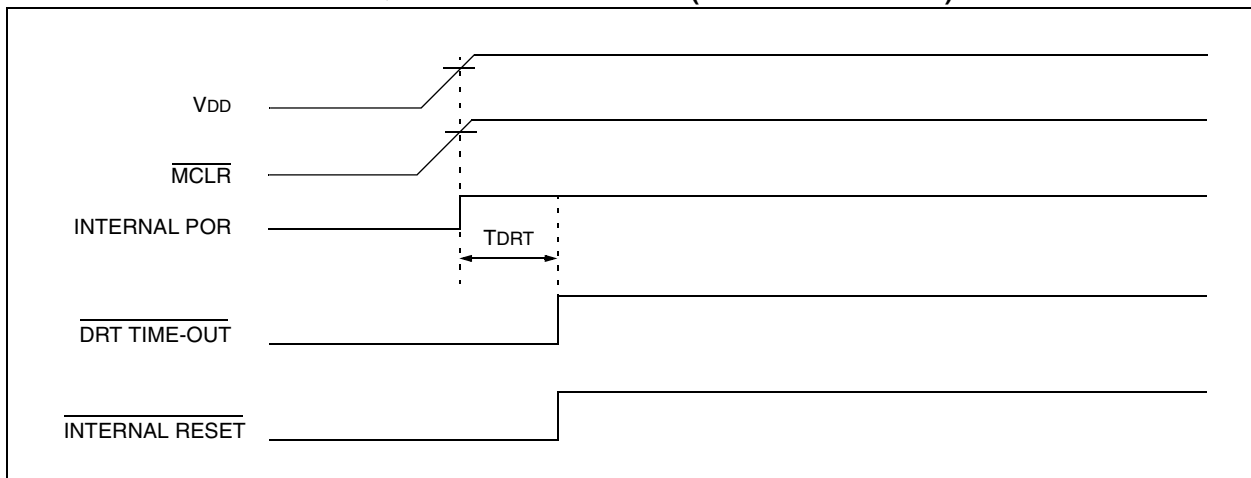
**2:** These values are for design guidance only. Rs may be required in HS mode as well as XT mode to avoid overdriving crystals with low drive level specification. Since each crystal has its own characteristics, the user should consult the crystal manufacturer for appropriate values of external components.

**Note:** If you change from this device to another device, please verify oscillator characteristics in your application.

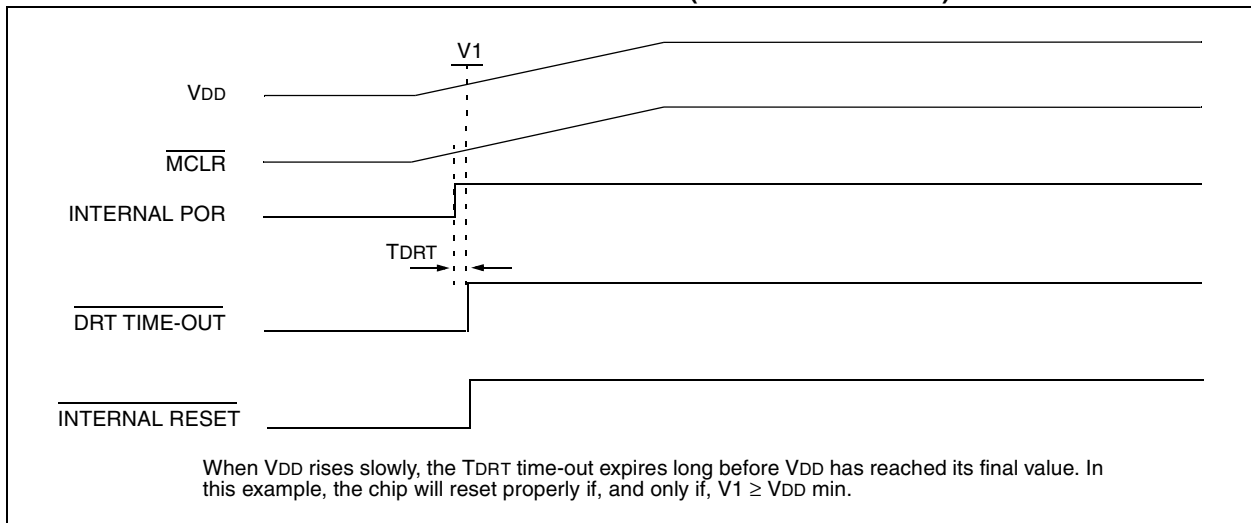
**FIGURE 7-8: TIME-OUT SEQUENCE ON POWER-UP ( $\overline{\text{MCLR}}$  NOT TIED TO  $V_{DD}$ )**



**FIGURE 7-9: TIME-OUT SEQUENCE ON POWER-UP ( $\overline{\text{MCLR}}$  TIED TO  $V_{DD}$ ): FAST  $V_{DD}$  RISE TIME**



**FIGURE 7-10: TIME-OUT SEQUENCE ON POWER-UP ( $\overline{\text{MCLR}}$  TIED TO  $V_{DD}$ ): SLOW  $V_{DD}$  RISE TIME**



## 7.5 Device Reset Timer (DRT)

In the PIC16HV540, the Device Reset Timer (DRT) runs any time the device is powered up. DRT runs from reset and varies based on oscillator selection (see Table 7-5).

The DRT provides a fixed 18 ms nominal time-out on reset. The DRT operates on an internal RC oscillator. The processor is kept in RESET as long as the DRT is active. The DRT delay allows V<sub>DD</sub> to rise above V<sub>DD</sub> min., and for the oscillator to stabilize.

Oscillator circuits based on crystals or ceramic resonators require a certain time after power-up to establish a stable oscillation. The on-chip DRT keeps the device in a RESET condition for approximately 18 ms after the voltage on the  $\overline{\text{MCLR}}$ /V<sub>PP</sub> pin has reach a logic high (V<sub>IH</sub>) level. Thus, external RC networks connected to the  $\overline{\text{MCLR}}$  input are not required in most cases, allowing for savings in cost-sensitive and/or space restricted applications.

The Device Reset time delay will vary from chip to chip due to V<sub>DD</sub>, temperature, and process variation. See AC parameters for details.

The DRT will also be triggered upon a Watchdog Timer time-out,  $\overline{\text{MCLR}}$  Reset, Wake-up from SLEEP on Pin Change and Brown-out Reset. When the external RC oscillator mode is selected, all DRT periods, after the initial POR, are 1 ms (typical).

**TABLE 7-5: DRT (DEVICE RESET TIMER PERIOD)**

Oscillator Configuration	POR Reset	Subsequent Resets
EXTRC	18 ms (typical)	1 ms (typical)
LP, XT & HS	18 ms (typical)	18 ms (typical)

## 7.6 Brown-Out Detect (BOD)

The PIC16HV540 has on-chip Brown-out Detect circuitry. If enabled and if the internal power, V<sub>REG</sub>, falls below parameter B<sub>VDD</sub> (See Section 10.1), for greater time than parameter T<sub>BOD</sub> (See Table 10-3) the brown-out condition will reset the chip. A reset is not guaranteed if V<sub>REG</sub> falls below B<sub>VDD</sub> for less time than parameter (T<sub>BOD</sub>).

On resets (Brown-out, Watchdog,  $\overline{\text{MCLR}}$  and Wake-up on Pin Change), the chip will remain in reset until V<sub>REG</sub> rises above B<sub>VDD</sub>. Once the B<sub>VDD</sub> threshold has been met the DRT will now be invoked and will keep the chip in reset an additional 18mS (LP, XT and HS oscillator modes) or 1mS for EXTRC.

If V<sub>REG</sub> drops below B<sub>VDD</sub> while the DRT is running, the chip will go back into a Brown-out Reset and the DRT will be re-initialized. Once V<sub>REG</sub> rises above the B<sub>VDD</sub>, the DRT will execute the specified time period. Figure 7-11 shows typical Brown-out situations.

The Brown-out Detect circuit can be disabled or enabled by setting the BODEN bit in the OPTION2 SFR. The Brown-out Detect is disabled upon all Power-on Resets (POR).

### 7.6.1 IMPLEMENTING THE ON-CHIP BOD CIRCUIT

The PIC16HV540 BOD circuitry differs from “conventional” brown-out detect circuitry in that the BOD circuitry on the PIC16HV540 does not directly detect “dips” in the external V<sub>DD</sub> supply voltage but rather the internal V<sub>REG</sub>. The functionality of the BOD circuitry ensures that program execution will halt and a reset state will be entered into prior to the internal logic becoming corrupted. The BOD circuit has two selectable voltage settings, nominally 5V and 3V. Each regulation voltage setting with its associated minimum and maximum B<sub>VDD</sub> parameters has an intended operational mode that must be carefully considered.

For the 5V V<sub>REG</sub> setting, the minimum B<sub>VDD</sub> parameter is 2.7V. This minimum B<sub>VDD</sub> voltage is below the part V<sub>DD</sub> minimum requirements. This operational setting is primarily intended for use when the PIC16HV540 is operating at 4Mhz and V<sub>DD</sub> > 5.5V.

For the 3V V<sub>REG</sub> setting, the minimum B<sub>VDD</sub> parameter is 1.8V. This minimum B<sub>VDD</sub> voltage is below the part V<sub>DD</sub> minimum requirements. This operational setting is primarily intended for use when the PIC16HV540 is in SLEEP. RAM retention is protected by the 1.8V trip level.

For the regulation and Brown-out circuits to function as intended the applied V<sub>DD</sub> is nominally 0.5V greater than the regulation voltage setting.

Finally, if the internal brown-out circuit is deemed not to meet system design requirements then an external brown-out protection circuit may be required. Microchip offers a complete family of voltage supervisor products which can meet most design requirements.

**TABLE 8-2: INSTRUCTION SET SUMMARY**

Mnemonic, Operands		Description	Cycles	12-Bit Opcode			Status Affected	Notes
				MSb	LSb			
ADDWF	f, d	Add W and f	1	0001	11df	ffff	C, DC, Z	1, 2, 4
ANDWF	f, d	AND W with f	1	0001	01df	ffff	Z	2, 4
CLRF	f	Clear f	1	0000	011f	ffff	Z	4
CLRW	—	Clear W	1	0000	0100	0000	Z	
COMF	f, d	Complement f	1	0010	01df	ffff	Z	
DECF	f, d	Decrement f	1	0000	11df	ffff	Z	2, 4
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	0010	11df	ffff	None	2, 4
INCF	f, d	Increment f	1	0010	10df	ffff	Z	2, 4
INCFSZ	f, d	Increment f, Skip if 0	1(2)	0011	11df	ffff	None	2, 4
IORWF	f, d	Inclusive OR W with f	1	0001	00df	ffff	Z	2, 4
MOVF	f, d	Move f	1	0010	00df	ffff	Z	2, 4
MOVWF	f	Move W to f	1	0000	001f	ffff	None	1, 4
NOP	—	No Operation	1	0000	0000	0000	None	
RLF	f, d	Rotate left f through Carry	1	0011	01df	ffff	C	2, 4
RRF	f, d	Rotate right f through Carry	1	0011	00df	ffff	C	2, 4
SUBWF	f, d	Subtract W from f	1	0000	10df	ffff	C, DC, Z	1, 2, 4
SWAPF	f, d	Swap f	1	0011	10df	ffff	None	2, 4
XORWF	f, d	Exclusive OR W with f	1	0001	10df	ffff	Z	2, 4
BIT-ORIENTED FILE REGISTER OPERATIONS								
BCF	f, b	Bit Clear f	1	0100	bbbf	ffff	None	2, 4
BSF	f, b	Bit Set f	1	0101	bbbf	ffff	None	2, 4
BTFSC	f, b	Bit Test f, Skip if Clear	1 (2)	0110	bbbf	ffff	None	
BTFSS	f, b	Bit Test f, Skip if Set	1 (2)	0111	bbbf	ffff	None	
LITERAL AND CONTROL OPERATIONS								
ANDLW	k	AND literal with W	1	1110	kkkk	kkkk	Z	1
CALL	k	Call subroutine	2	1001	kkkk	kkkk	None	
CLRWDT	k	Clear Watchdog Timer	1	0000	0000	0100	TO, PD	
GOTO	k	Unconditional branch	2	101k	kkkk	kkkk	None	
IORLW	k	Inclusive OR Literal with W	1	1101	kkkk	kkkk	Z	
MOVLW	k	Move Literal to W	1	1100	kkkk	kkkk	None	
OPTION	k	Load OPTION register	1	0000	0000	0010	None	
RETLW	k	Return, place Literal in W	2	1000	kkkk	kkkk	None	
SLEEP	—	Go into standby mode	1	0000	0000	0011	TO, PD, PCWUF	
TRIS	f	Load TRIS register	1	0000	0000	0fff	None	
XORLW	k	Exclusive OR Literal to W	1	1111	kkkk	kkkk	Z	3

**Note 1:** The 9th bit of the program counter will be forced to a '0' by any instruction that writes to the PC except for GOTO. (See individual device data sheets, Memory Section/Indirect Data Addressing, INDF and FSR Registers)

- When an I/O register is modified as a function of itself (e.g. MOVF PORTB, 1), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.
- The instruction TRIS f, where f = 5 or 6 causes the contents of the W register to be written to the tristate latches of PORTA or B respectively. A '1' forces the pin to a hi-impedance state and disables the output buffers.
- If this instruction is executed on the TMR0 register (and, where applicable, d = 1), the prescaler will be cleared (if assigned to TMR0).

CALL	Subroutine Call			
Syntax:	[ <i>label</i> ] CALL k			
Operands:	$0 \leq k \leq 255$			
Operation:	(PC) + 1 → Top of Stack; k → PC<7:0>; (STATUS<6:5>) → PC<10:9>; 0 → PC<8>			
Status Affected:	None			
Encoding:	<table border="1"><tr><td>1001</td><td>kkkk</td><td>kkkk</td></tr></table>	1001	kkkk	kkkk
1001	kkkk	kkkk		
Description:	Subroutine call. First, return address (PC+1) is pushed onto the stack. The eight bit immediate address is loaded into PC bits <7:0>. The upper bits PC<10:9> are loaded from STATUS<6:5>, PC<8> is cleared. CALL is a two cycle instruction.			
Words:	1			
Cycles:	2			
Example:	HERE      CALL      THERE			

Before Instruction

PC = address (HERE)

After Instruction

PC = address (THERE)

TOS = address (HERE + 1)

CLRF	Clear f			
Syntax:	[ <i>label</i> ] CLRF f			
Operands:	$0 \leq f \leq 31$			
Operation:	00h $\rightarrow$ (f); 1 $\rightarrow$ Z			
Status Affected:	Z			
Encoding:	<table border="1"><tr><td>0000</td><td>011f</td><td>ffff</td></tr></table>	0000	011f	ffff
0000	011f	ffff		
Description:	The contents of register 'f' are cleared and the Z bit is set.			
Words:	1			
Cycles:	1			
Example:	CLRF FLAG_REG			

Before Instruction

FLAG\_REG = 0x5A

After Instruction

FLAG\_REG = 0x00

Z = 1

CLRW		Clear W				
Syntax:	[ <i>label</i> ] CLRW					
Operands:	None					
Operation:	00h → (W); 1 → Z					
Status Affected:	Z					
Encoding:	<table border="1"><tr><td>0000</td><td>0100</td><td>0000</td></tr></table>			0000	0100	0000
0000	0100	0000				
Description:	The W register is cleared. Zero bit (Z) is set.					
Words:	1					
Cycles:	1					
Example:	CLRW					
Before Instruction						
W = 0x5A						
After Instruction						
W = 0x00						
Z = 1						

CLRWDWT		Clear Watchdog Timer				
Syntax:	[ <i>label</i> ] CLRWDWT					
Operands:	None					
Operation:	00h → WDT; 0 → WDT prescaler (if assigned); 1 → $\overline{TO}$ ; 1 → $\overline{PD}$					
Status Affected:	$\overline{TO}$ , $\overline{PD}$					
Encoding:	<table border="1"><tr><td>0000</td><td>0000</td><td>0100</td></tr></table>			0000	0000	0100
0000	0000	0100				
Description:	The CLRWDWT instruction resets the WDT. It also resets the prescaler, if the prescaler is assigned to the WDT and not Timer0. Status bits $\overline{TO}$ and $\overline{PD}$ are set.					
Words:	1					
Cycles:	1					
Example:	CLRWDWT					

Before Instruction

WDT counter = ?

After Instruction

WDT counter = 0x00

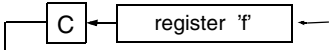
WDT prescale = 0

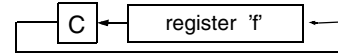
$\overline{TO}$  = 1

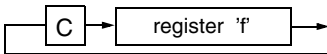
PD = 1

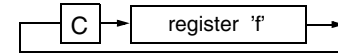
OPTION		Load OPTION Register				
Syntax:	[ <i>label</i> ]    OPTION					
Operands:	None					
Operation:	(W) → OPTION					
Status Affected:	None					
Encoding:	<table border="1"><tr><td>0000</td><td>0000</td><td>0010</td></tr></table>			0000	0000	0010
0000	0000	0010				
Description:	The content of the W register is loaded into the OPTION register.					
Words:	1					
Cycles:	1					
Example	OPTION N					
Before Instruction						
W	=	0x07				
After Instruction						
OPTION	=	0x07				

RETLW		Return with Literal in W			
Syntax:	[ <i>label</i> ] RETLW k				
Operands:	0 ≤ k ≤ 255				
Operation:	k → (W); TOS → PC				
Status Affected:	None				
Encoding:	<table border="1"><tr><td>1000</td><td>kkkk</td><td>kkkk</td></tr></table>		1000	kkkk	kkkk
1000	kkkk	kkkk			
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				
Example:	<pre>CALL TABLE ;W contains                 ;table offset                 ;value.     •           ;W now has table     •           ;value.     • TABLE  ADDWF PC    ;W = offset       RETLW k1     ;Begin table       RETLW k2     ;     •     •     •       RETLW kn     ; End of table</pre>				
Before Instruction	W = 0x07				
After Instruction	W = value of k8				

RLF		Rotate Left f through Carry				
Syntax:	[ label ]	RLF	f,d			
Operands:	$0 \leq f \leq 31$ $d \in [0,1]$					
Operation:	See description below					
Status Affected:	C					
Encoding:	<table border="1"><tr><td>0011</td><td>01df</td><td>ffff</td></tr></table>			0011	01df	ffff
0011	01df	ffff				
Description:	<p>The contents of register 'f' are rotated one bit to the left through the Carry Flag. If 'd' is 0 the result is placed in the W register. If 'd' is 1 the result is stored back in register 'f'.</p> 					
Words:	1					
Cycles:	1					
Example:	RLF REG1, 0					



RRF	Rotate Right f through Carry			
Syntax:	[ <i>label</i> ] RRF f,d			
Operands:	$0 \leq f \leq 31$ $d \in [0,1]$			
Operation:	See description below			
Status Affected:	C			
Encoding:	<table><tr><td>0011</td><td>00df</td><td>ffff</td></tr></table>	0011	00df	ffff
0011	00df	ffff		
Description:	The contents of register 'f' are rotated one bit to the right through the Carry Flag. If 'd' is 0 the result is placed in the W register. If 'd' is 1 the result is placed back in register 'f'.			
				
Words:	1			
Cycles:	1			
Example:	RRF REG1,0			
Before Instruction	REG1 = 1110 0110 C = 0			
After Instruction	REG1 = 1110 0110 W = 0111 0011 C = 0			





NOTES:

## **9.4 MPLINK/MPLIB Linker/Librarian**

MPLINK is a relocatable linker for MPASM and MPLAB-C17 and MPLAB-C18. It can link relocatable objects from assembly or C source files along with pre-compiled libraries using directives from a linker script.

MPLIB is a librarian for pre-compiled code to be used with MPLINK. When a routine from a library is called from another source file, only the modules that contains that routine will be linked in with the application. This allows large libraries to be used efficiently in many different applications. MPLIB manages the creation and modification of library files.

MPLINK features include:

- MPLINK works with MPASM and MPLAB-C17 and MPLAB-C18.
- MPLINK allows all memory areas to be defined as sections to provide link-time flexibility.

MPLIB features include:

- MPLIB makes linking easier because single libraries can be included instead of many smaller files.
- MPLIB helps keep code maintainable by grouping related modules together.
- MPLIB commands allow libraries to be created and modules to be added, listed, replaced, deleted, or extracted.

## **9.5 MPLAB-SIM Software Simulator**

The MPLAB-SIM Software Simulator allows code development in a PC host environment by simulating the PICmicro series microcontrollers on an instruction level. On any given instruction, the data areas can be examined or modified and stimuli can be applied from a file or user-defined key press to any of the pins. The execution can be performed in single step, execute until break, or trace mode.

MPLAB-SIM fully supports symbolic debugging using MPLAB-C17 and MPLAB-C18 and MPASM. The Software Simulator offers the flexibility to develop and debug code outside of the laboratory environment making it an excellent multi-project software development tool.

## **9.6 MPLAB-ICE High Performance Universal In-Circuit Emulator with MPLAB IDE**

The MPLAB-ICE Universal In-Circuit Emulator is intended to provide the product development engineer with a complete microcontroller design tool set for PICmicro microcontrollers (MCUs). Software control of MPLAB-ICE is provided by the MPLAB Integrated Development Environment (IDE), which allows editing, “make” and download, and source debugging from a single environment.

Interchangeable processor modules allow the system to be easily reconfigured for emulation of different processors. The universal architecture of the MPLAB-ICE allows expansion to support new PICmicro microcontrollers.

The MPLAB-ICE Emulator System has been designed as a real-time emulation system with advanced features that are generally found on more expensive development tools. The PC platform and Microsoft® Windows 3.x/95/98 environment were chosen to best make these features available to you, the end user.

MPLAB-ICE 2000 is a full-featured emulator system with enhanced trace, trigger, and data monitoring features. Both systems use the same processor modules and will operate across the full operating speed range of the PICmicro MCU.

## **9.7 PICMASTER/PICMASTER CE**

The PICMASTER system from Microchip Technology is a full-featured, professional quality emulator system. This flexible in-circuit emulator provides a high-quality, universal platform for emulating Microchip 8-bit PICmicro microcontrollers (MCUs). PICMASTER systems are sold worldwide, with a CE compliant model available for European Union (EU) countries.

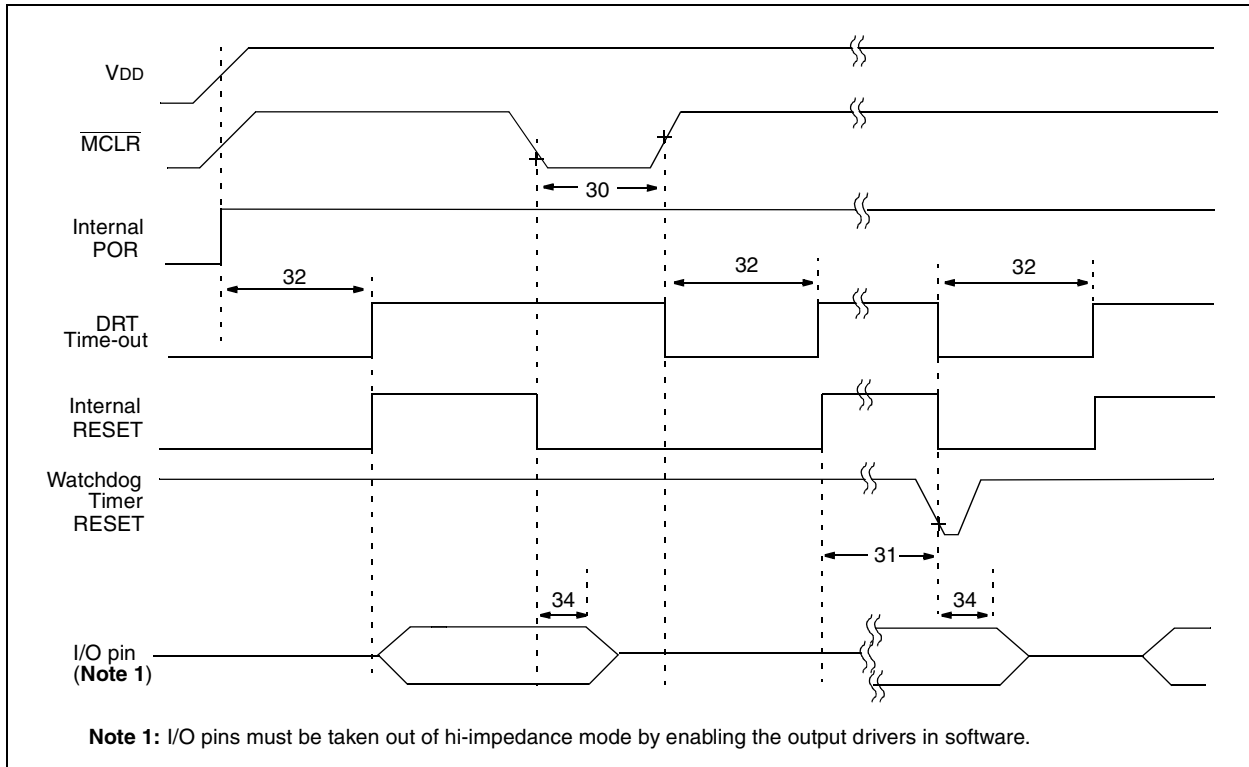
## **9.8 ICEPIC**

ICEPIC is a low-cost in-circuit emulation solution for the Microchip Technology PIC16C5X, PIC16C6X, PIC16C7X, and PIC16CXXX families of 8-bit one-time-programmable (OTP) microcontrollers. The modular system can support different subsets of PIC16C5X or PIC16CXXX products through the use of interchangeable personality modules or daughter boards. The emulator is capable of emulating without target application circuitry being present.

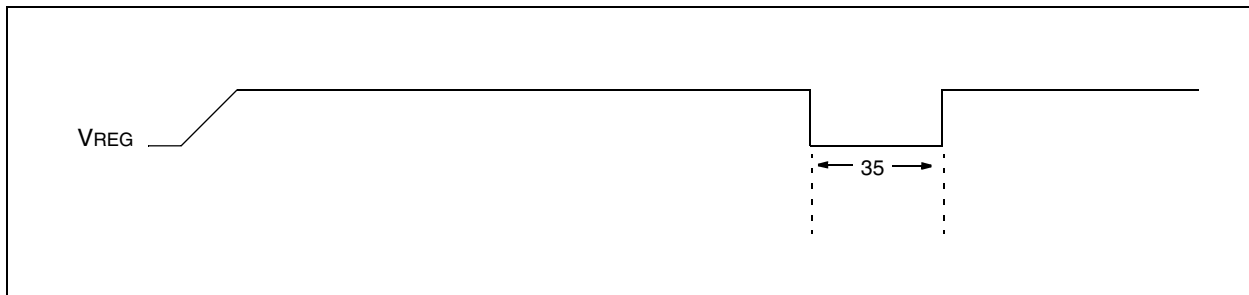
## **9.9 MPLAB-ICD In-Circuit Debugger**

Microchip's In-Circuit Debugger, MPLAB-ICD, is a powerful, low-cost run-time development tool. This tool is based on the flash PIC16F877 and can be used to develop for this and other PICmicro microcontrollers from the PIC16CXXX family. MPLAB-ICD utilizes the In-Circuit Debugging capability built into the PIC16F87X. This feature, along with Microchip's In-Circuit Serial Programming protocol, offers cost-effective in-circuit flash programming and debugging from the graphical user interface of the MPLAB Integrated Development Environment. This enables a designer to develop and debug source code by watching variables, single-stepping and setting break points. Running at full speed enables testing hardware in real-time. The MPLAB-ICD is also a programmer for the flash PIC16F87X family.

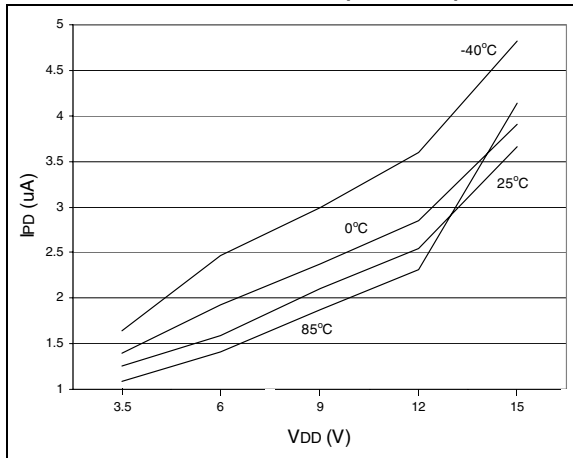
**FIGURE 10-4: RESET, WATCHDOG TIMER, AND DEVICE RESET TIMER TIMING - PIC16HV540**



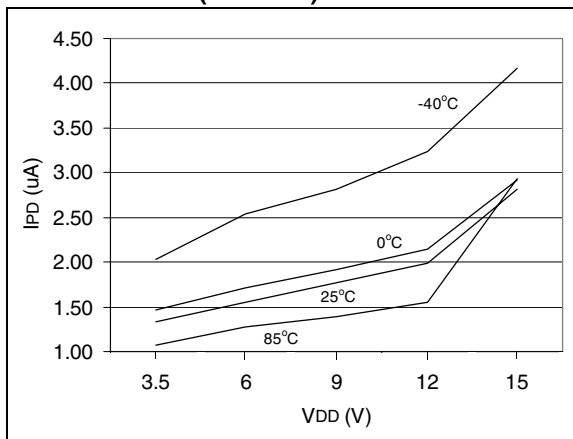
**FIGURE 10-5: BROWN-OUT DETECT TIMING**



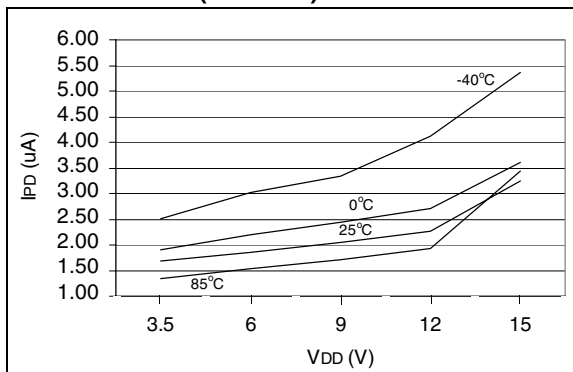
**FIGURE 11-10: MAXIMUM  $I_{PD}$  vs.  $V_{DD}$ , WATCHDOG TIMER DISABLED ( $V_{IO} = 3V$ )**



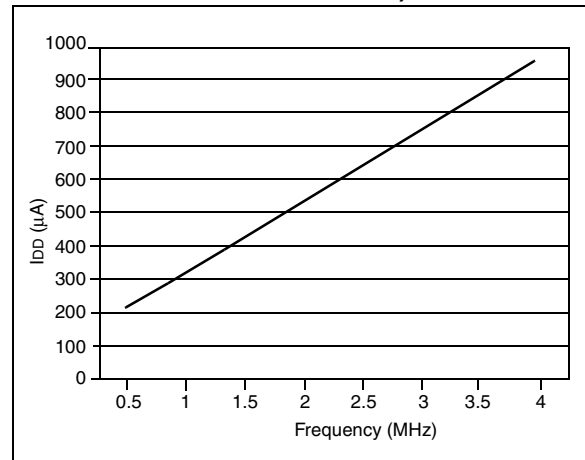
**FIGURE 11-11: TYPICAL  $I_{PD}$  vs.  $V_{DD}$ , WATCHDOG TIMER ENABLED ( $V_{IO} = 3V$ )**



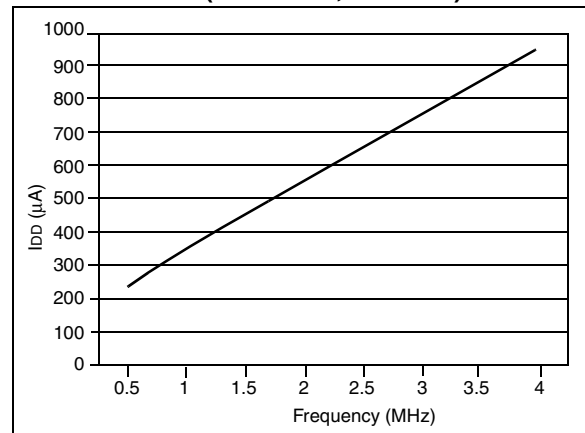
**FIGURE 11-12: MAXIMUM  $I_{PD}$  vs.  $V_{DD}$ , WATCHDOG TIMER ENABLED ( $V_{IO} = 3V$ )**



**FIGURE 11-13: MAXIMUM  $I_{DD}$  vs. FREQUENCY, WATCHDOG TIMER DISABLED, RC MODE ( $V_{DD} = 15V$ ,  $V_{IO} = 5V$ , -40°C TO +85°C)**

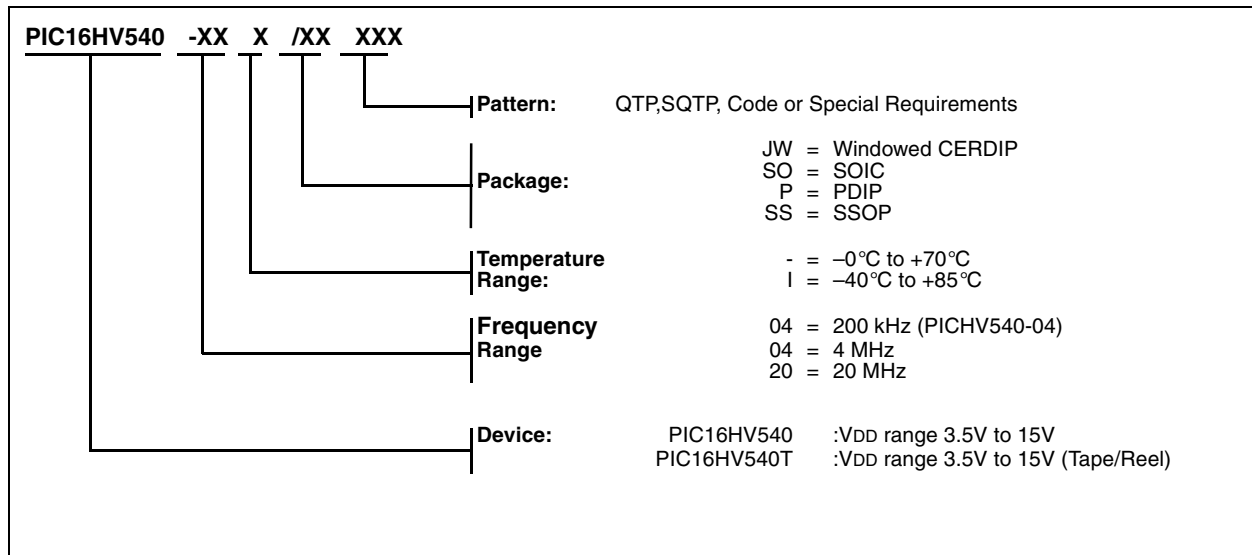


**FIGURE 11-14: MAXIMUM  $I_{DD}$  vs. FREQUENCY, WATCHDOG TIMER ENABLED, RC MODE ( $V_{DD} = 15V$ ,  $V_{IO} = 5V$ )**



## PIC16HV540 PRODUCT IDENTIFICATION SYSTEM

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.



## Sales and Support

### Data Sheets

Products supported by a preliminary Data Sheet may have an errata sheet describing minor operational differences and recommended workarounds. To determine if an errata sheet exists for a particular device, please contact one of the following:

1. Your local Microchip sales office
2. The Microchip Corporate Literature Center U.S. FAX: (480) 786-7277.
3. The Microchip Worldwide Site ([www.microchip.com](http://www.microchip.com))

Please specify which device, revision of silicon and Data Sheet (include Literature #) you are using.

### New Customer Notification System

Register on our web site ([www.microchip.com/cn](http://www.microchip.com/cn)) to receive the most current information on our products.