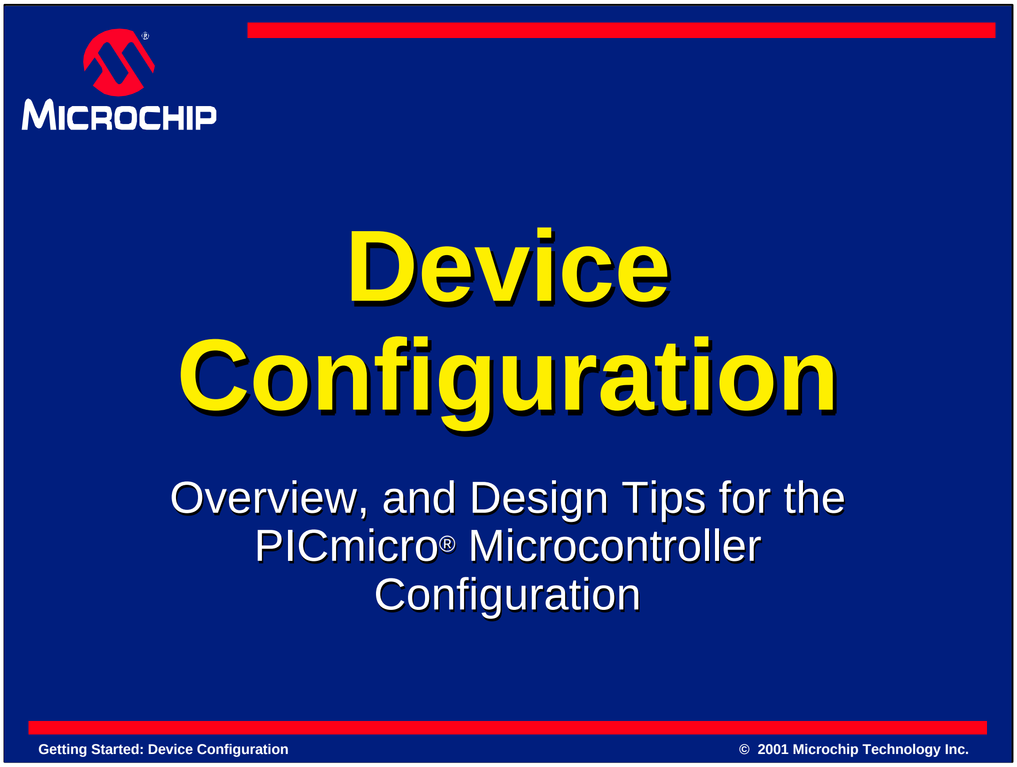
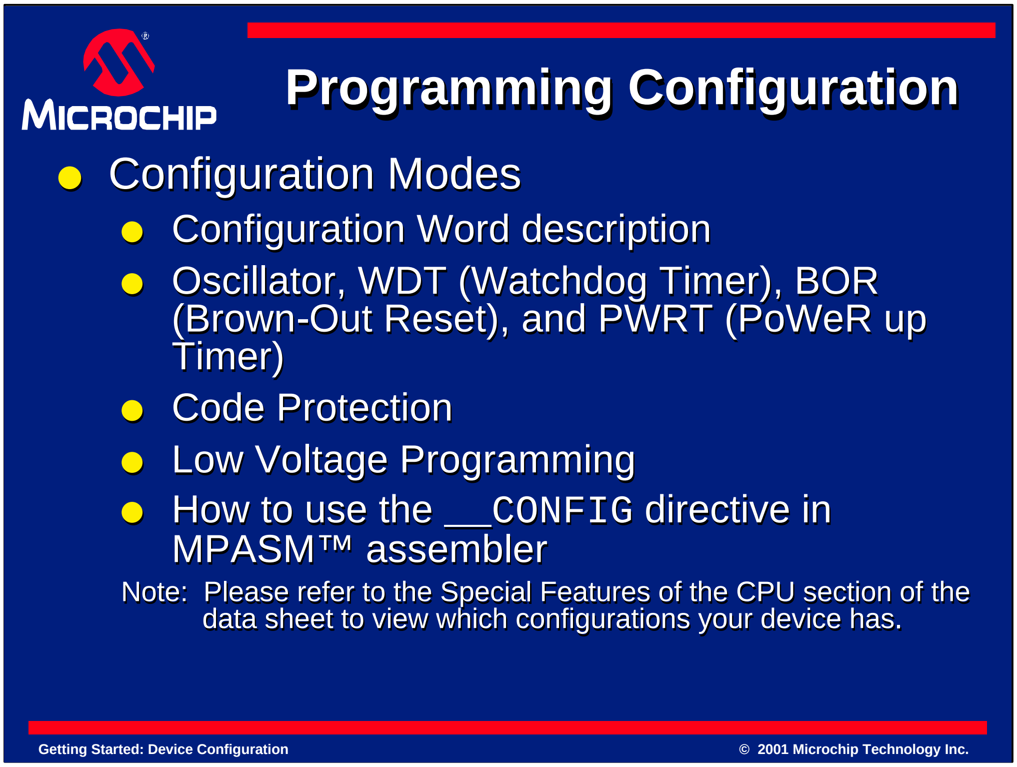


Device Configuration

Overview, and Design Tips for the PICmicro® Microcontroller Configuration

Getting Started: Device Configuration

In this presentation we will examine Device Configuration as it relates to Microchip’s midrange PICmicro® Microcontrollers (MCUs). Configuration refers to a location on the chip that specifies the characteristics that the device will have for operation.



Programming Configuration

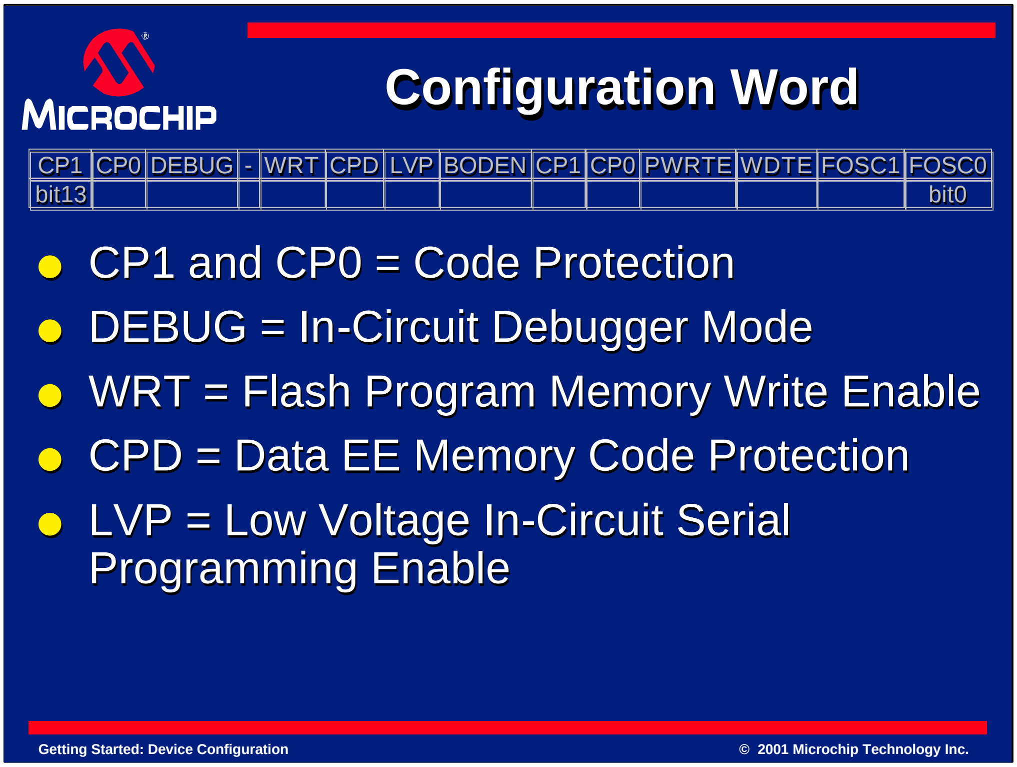
- Configuration Modes
  - Configuration Word description
  - Oscillator, WDT (Watchdog Timer), BOR (Brown-Out Reset), and PWRT (PoWeR up Timer)
  - Code Protection
  - Low Voltage Programming
  - How to use the `__CONFIG` directive in MPASM™ assembler

Note: Please refer to the Special Features of the CPU section of the data sheet to view which configurations your device has.

Getting Started: Device Configuration © 2001 Microchip Technology Inc.

This 14-bit register allows the user to set up the oscillator mode and enable or disable features such as the Watchdog Timer (WDT), Code Protection, and Low Voltage Programming.

In this tutorial, we will first explain the Configuration Word. Then we will examine the following options: OSCILLATOR modes, WDT (Watchdog Timer), BOR (Brown-Out Reset), and PWRT (PoWeR-up Timer). From there we will look at code protection and Low Voltage Programming, and lastly, we will explain the "__CONFIG" directive in MPASM™ Assembler.



Configuration Word

CP1	CP0	DEBUG	-	WRT	CPD	LVP	BODEN	CP1	CP0	PWRTE	WDTE	FOSC1	FOSC0
bit13													bit0

- CP1 and CP0 = Code Protection
- DEBUG = In-Circuit Debugger Mode
- WRT = Flash Program Memory Write Enable
- CPD = Data EE Memory Code Protection
- LVP = Low Voltage In-Circuit Serial Programming Enable

Getting Started: Device Configuration

© 2001 Microchip Technology Inc.

Here we can see the Configuration Word options available for the PIC16F87X family of microcontrollers.

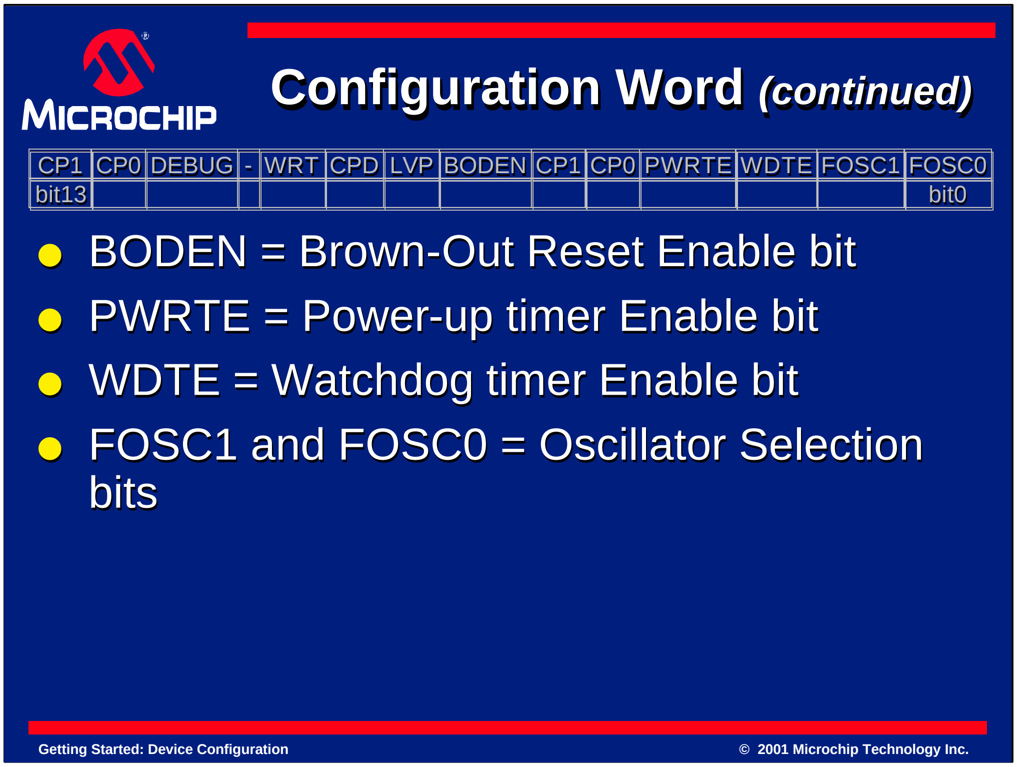
CP1 and CP0 = Code Protection

DEBUG = In-Circuit Debugger Mode

WRT = Flash Program Memory Write Enable

CPD = Data EE Memory Code Protection

LVP = Low Voltage In-Circuit Serial Programming Enable



MICROCHIP

Configuration Word *(continued)*

CP1	CP0	DEBUG	-	WRT	CPD	LVP	BODEN	CP1	CP0	PWRTE	WDTE	FOSC1	FOSC0
bit13													bit0

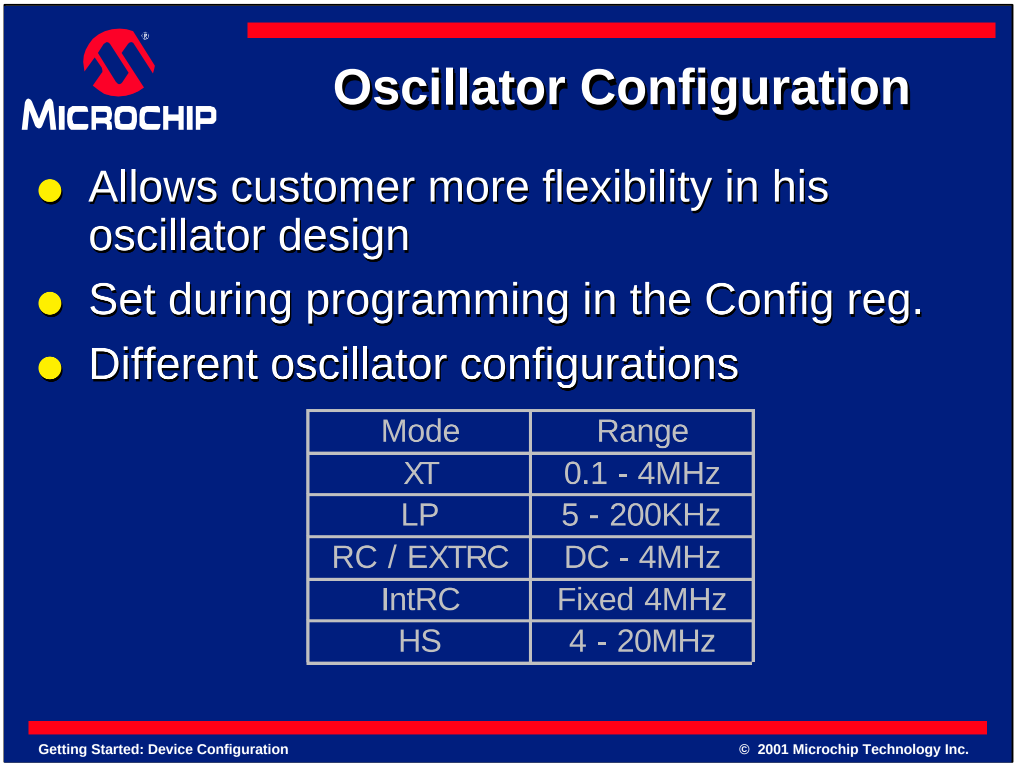
- BODEN = Brown-Out Reset Enable bit
- PWRTE = Power-up timer Enable bit
- WDTE = Watchdog timer Enable bit
- FOSC1 and FOSC0 = Oscillator Selection bits

BODEN = Brown-Out Reset Enable bit

PWRTE = Power-up timer Enable bit

WDTE = Watchdog timer Enable bit

FOSC1 and FOSC0 = Oscillator Selection bits



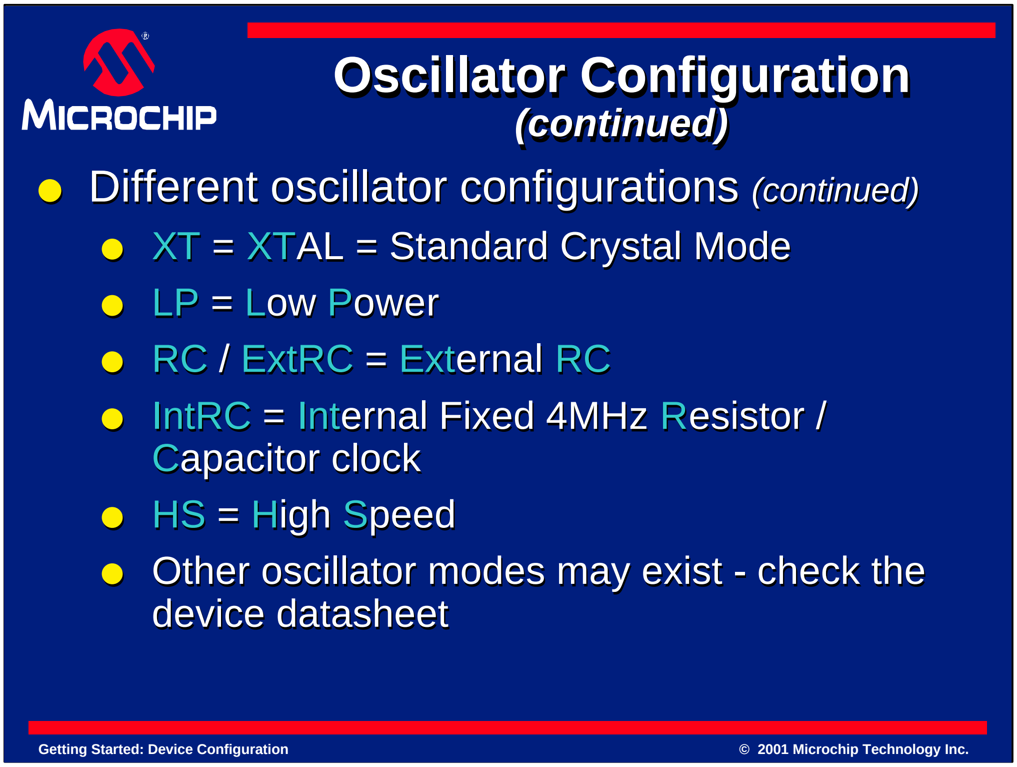
Oscillator Configuration

- Allows customer more flexibility in his oscillator design
- Set during programming in the Config reg.
- Different oscillator configurations

Mode	Range
XT	0.1 - 4MHz
LP	5 - 200KHz
RC / EXTRC	DC - 4MHz
IntRC	Fixed 4MHz
HS	4 - 20MHz

Getting Started: Device Configuration

The oscillator configuration is set during programming. The oscillator mode is selected by the device configuration bits. Midrange PICmicro devices can have up to eight oscillator modes, this allows a single device type the flexibility to fit applications with different oscillator requirements. Each mode provides varying amounts of oscillator gain for various oscillator designs. Listed here are the currently available oscillator configurations. See your specific device datasheet for the ones that apply.



Oscillator Configuration *(continued)*

- Different oscillator configurations *(continued)*
  - XT = XTAL = Standard Crystal Mode
  - LP = Low Power
  - RC / ExtRC = External RC
  - IntRC = Internal Fixed 4MHz Resistor / Capacitor clock
  - HS = High Speed
  - Other oscillator modes may exist - check the device datasheet

Getting Started: Device Configuration

© 2001 Microchip Technology Inc.

XT represents the Standard Crystal Mode, ranging from 200kHz - 4MHz crystals or 200kHz - 3MHz resonators

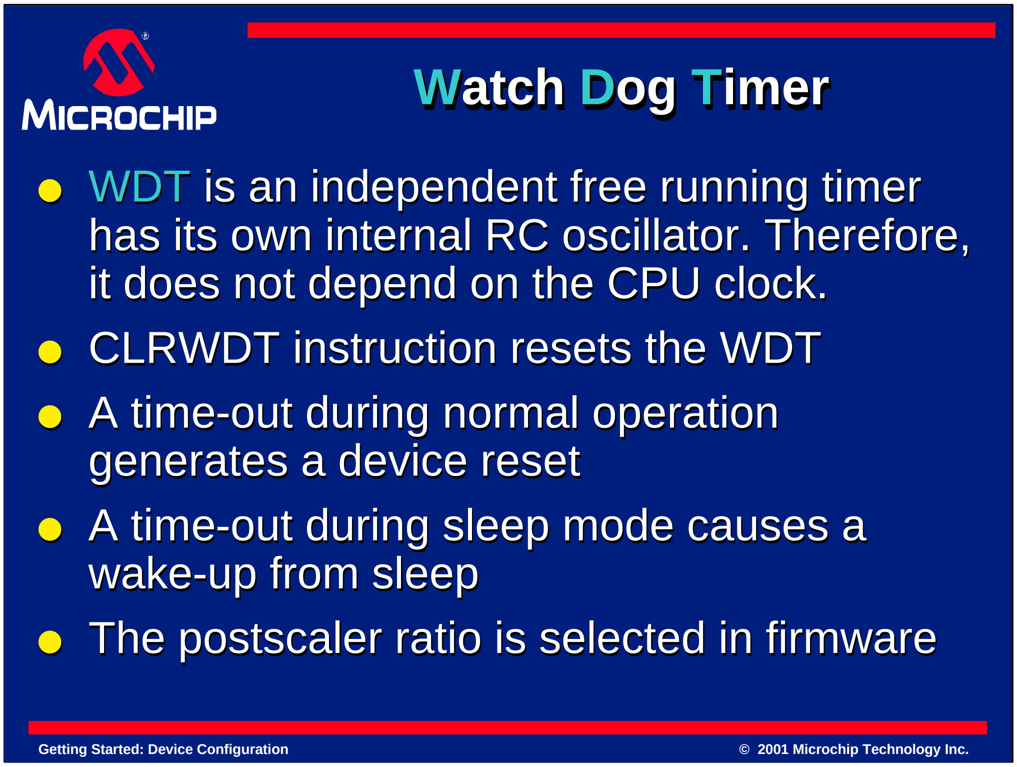
LP designates Low Power, ranging from 20kKHz to 200kHz, and usually used with 32.768kHz.

RC or ExtRC stands for External RC

IntRC is the Internal 4MHz Resistor / Capacitor clock

HS designates High Speed, and is generally used for crystals over 4MHz, and resonators over 3MHz.

Note that additional oscillator configurations exist, so it is important to check your device datasheet for the ones that apply to your device.



Watch Dog Timer

- WDT is an independent free running timer has its own internal RC oscillator. Therefore, it does not depend on the CPU clock.
- CLRWDT instruction resets the WDT
- A time-out during normal operation generates a device reset
- A time-out during sleep mode causes a wake-up from sleep
- The postscaler ratio is selected in firmware

Getting Started: Device Configuration

© 2001 Microchip Technology Inc.

One of the features that is enabled through the configuration bits is the Watchdog Timer or WDT. The Watchdog Timer is an independent free running timer because it has its own onboard internal RC oscillator. This means it does not depend on the CPU clock. Typically the timeout period for the Watchdog Timer is 18ms, but refer to the Electrical Specifications in you device datasheet for specifics and additional information regarding the timeout period for your device.

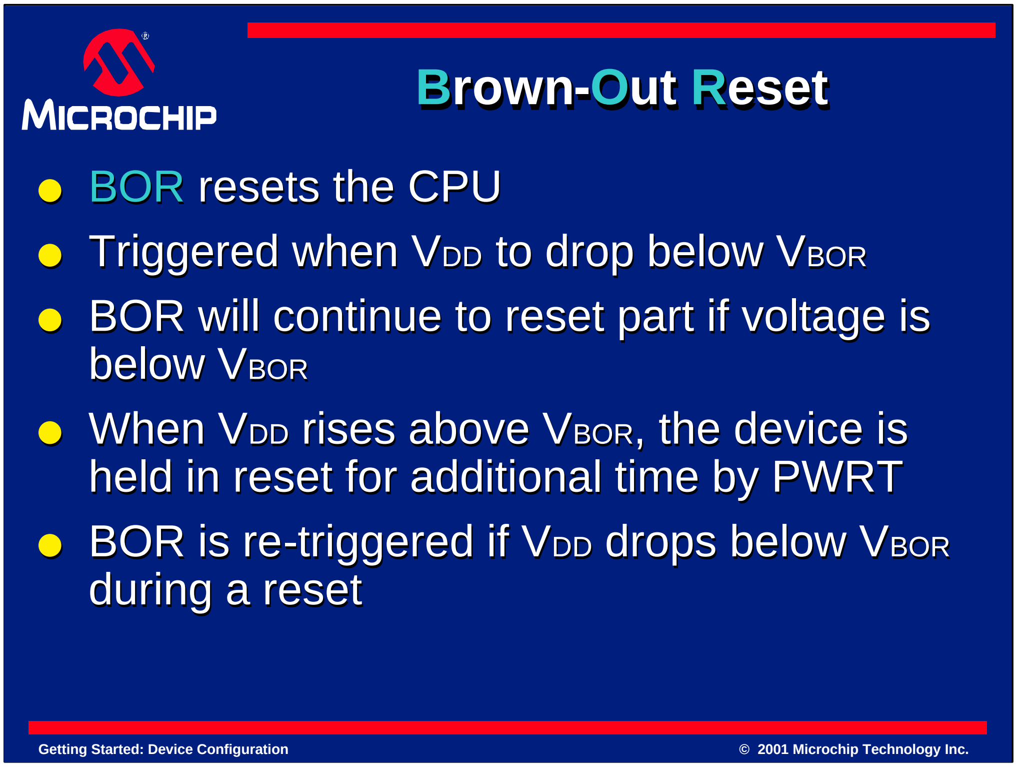
To prevent the timer from resetting a CLRWDT instruction may be used.

This timer generates two types of conditions when a time-out occurs.

The first condition is a device reset when running, and this resets the WDT and postscaller counter.

OR, the second condition that could be generated is a Wake-up from SLEEP.

The Watchdog Timer has an available postscaler to extend the Watchdog Timer time out.



Brown-Out Reset

- BOR resets the CPU
- Triggered when VDD to drop below VBOR
- BOR will continue to reset part if voltage is below VBOR
- When VDD rises above VBOR, the device is held in reset for additional time by PWRT
- BOR is re-triggered if VDD drops below VBOR during a reset

Getting Started: Device Configuration

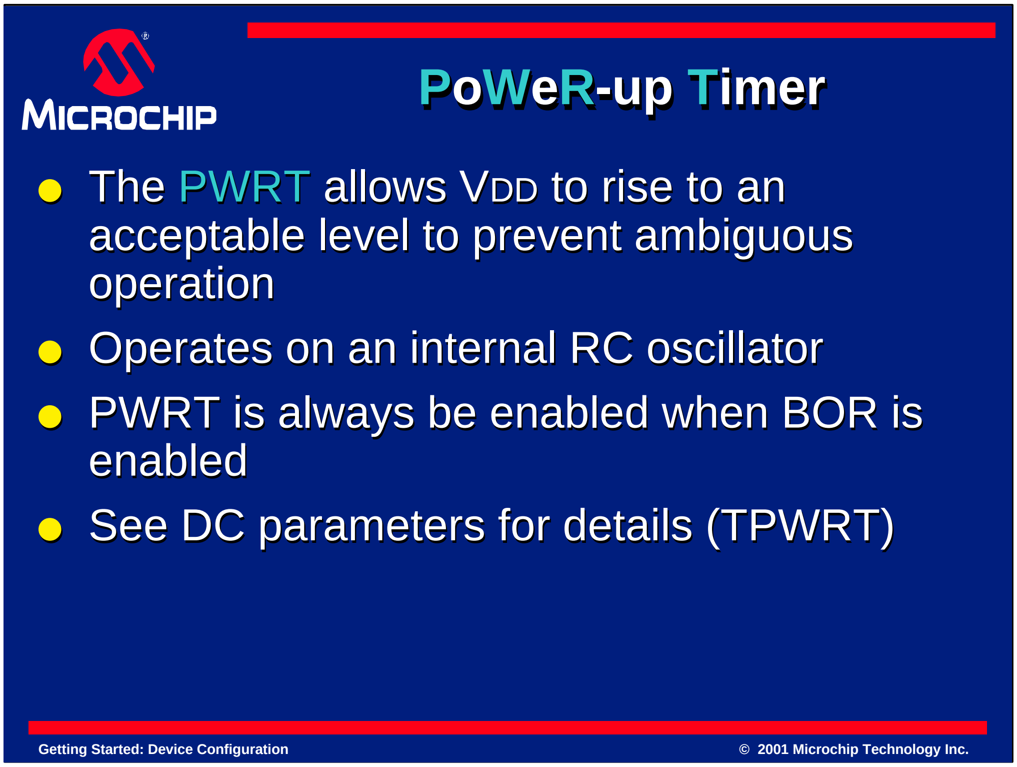
© 2001 Microchip Technology Inc.

Another feature enabled through the configuration bits is the Brown-Out Reset or BOR. The function of Brown-Out Reset is to reset the CPU when VDD drops below VBOR, this is typically at 4 volts.

The BOR will hold the device in reset for as long as VDD remains below VBOR.

When VDD rises above VBOR, the device is held in reset for additional time by the PoWeR-up Timer (PWRT).

If VDD should drop below VBOR at anytime, the BOR will then restart the process again.



PoWeR-up Timer

- The PWRT allows VDD to rise to an acceptable level to prevent ambiguous operation
- Operates on an internal RC oscillator
- PWRT is always be enabled when BOR is enabled
- See DC parameters for details (TPWRT)

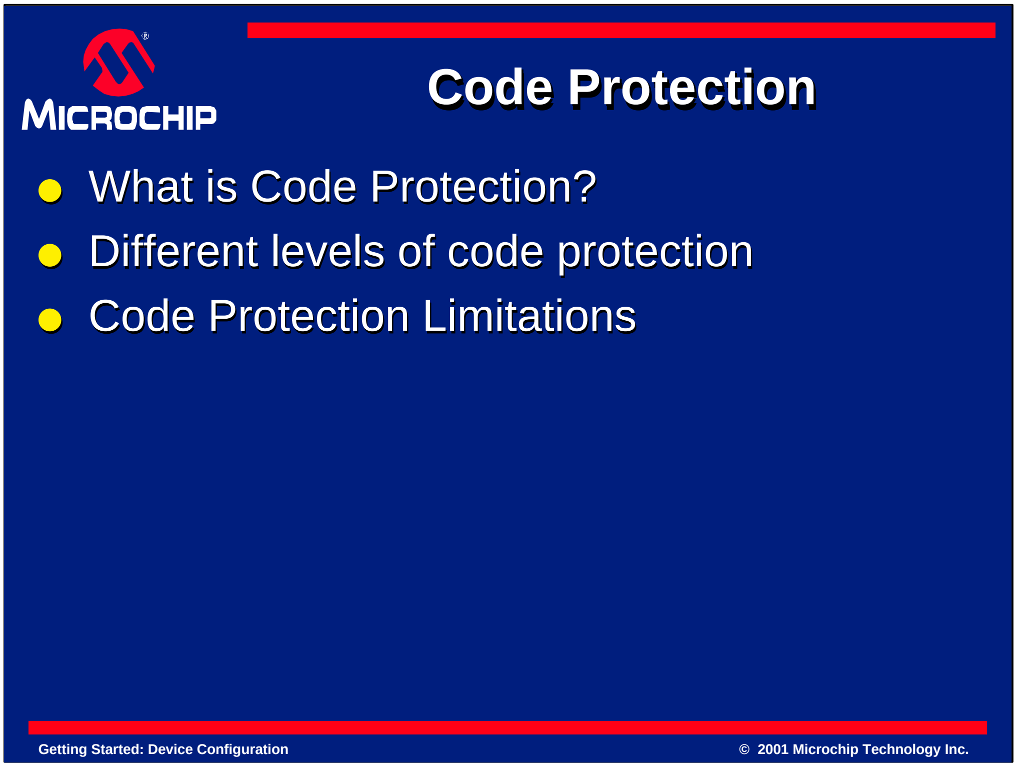
Getting Started: Device Configuration © 2001 Microchip Technology Inc.

The Power-Up Timer is also enabled through the configuration bits. The Power-Up Timer or PWRT provides a time-out period to allow the VDD to rise to an acceptable level to prevent ambiguous operation.

It operates on an internal RC oscillator that provides a nominal 72ms delay.

When Brown-Out Reset is used the Power-Up Timer is always enabled. Power-Up Timer is always enabled when Brown-Out Reset is enabled, even if Power-Up Timer is disabled. They work together to allow VDD to rise to an acceptable level after a Brown-Out Reset .

Please see your device specifications for Power-Up Timer .



Code Protection

- What is Code Protection?
- Different levels of code protection
- Code Protection Limitations

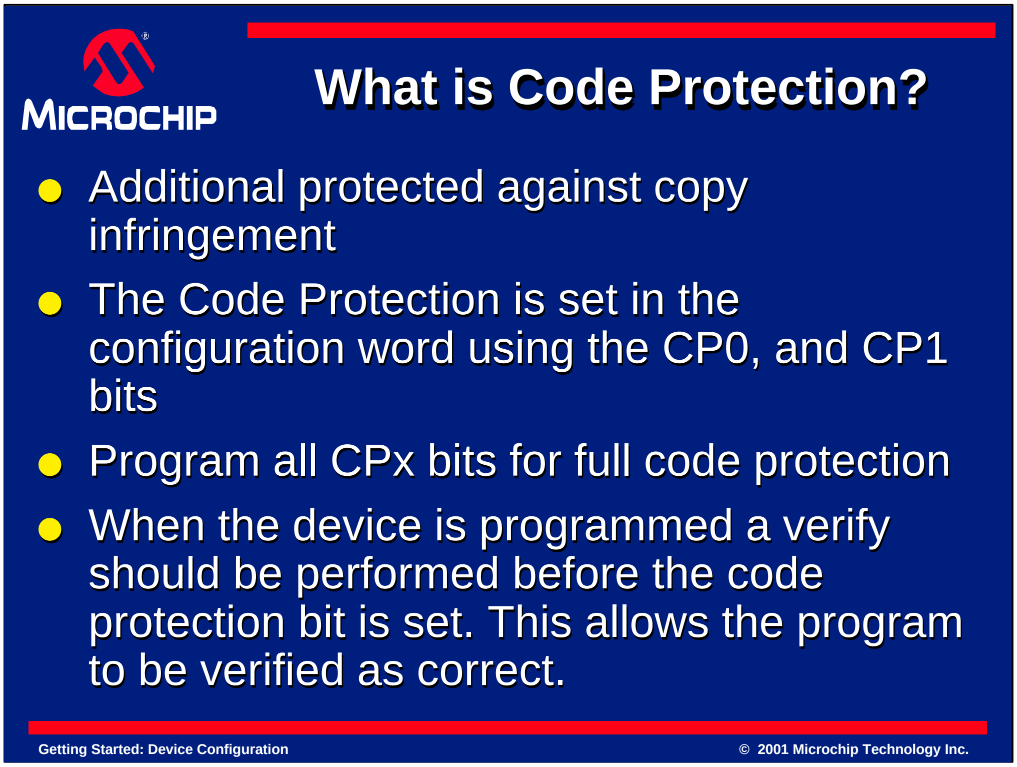
Getting Started: Device Configuration

© 2001 Microchip Technology Inc.

Code Protection is the next Configuration Mode we will review. First we will discuss what Code Protection is, how it can help you, and how it is implemented.

Then, we can look at the different levels of code protection using a PIC16F87X device for our example. With that we can see how this microcontroller can be code protected at different sections of program memory.

Lastly, we'll discuss some of the limitations of Code Protection, and in particular the limitations of Code Protection on the windowed and FLASH devices.



What is Code Protection?

- Additional protected against copy infringement
- The Code Protection is set in the configuration word using the CP0, and CP1 bits
- Program all CPx bits for full code protection
- When the device is programmed a verify should be performed before the code protection bit is set. This allows the program to be verified as correct.

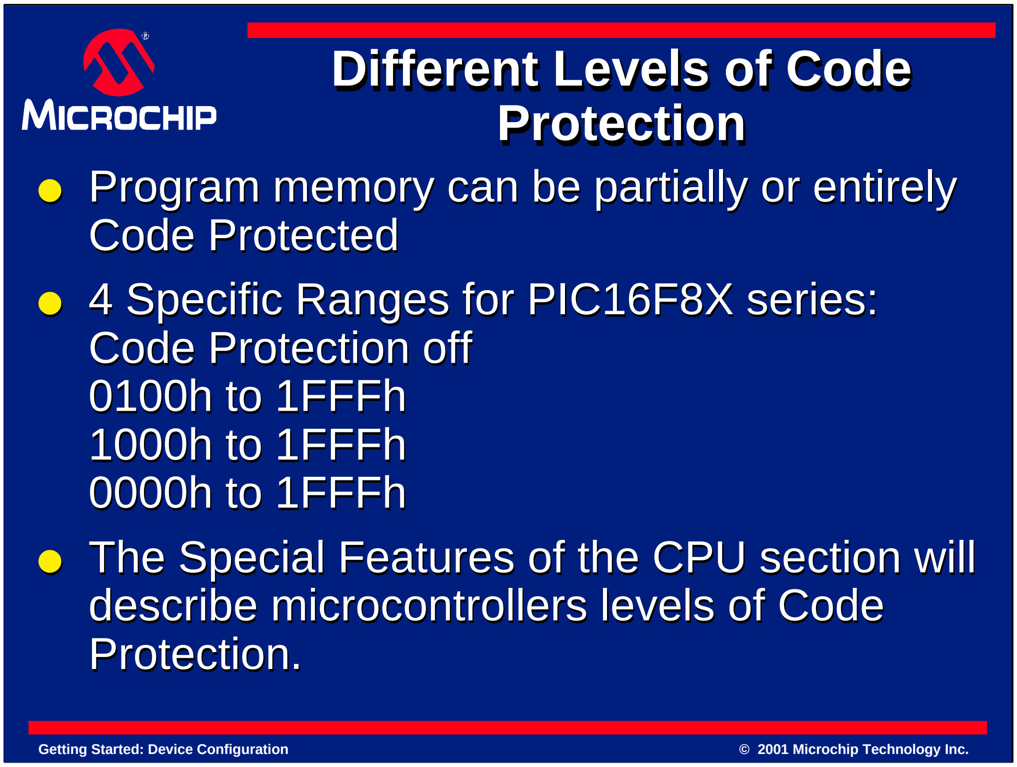
Getting Started: Device Configuration © 2001 Microchip Technology Inc.

Code Protection is a user selectable feature that provides additional protection for your code and prevents program and data EEPROM from being read out.

Code Protection is controlled by CP0 and CP1, both of which are located in the configuration word.

You’ll notice that there may be several bits labeled CPx in the configuration word. They must all be programmed in order for full protection.

When the device is programmed, a verify should be performed before the code protection bits are set. Since code protection prevents reading out the data, it is best to verify the part before setting the code protection bits.



Different Levels of Code Protection

- Program memory can be partially or entirely Code Protected
- 4 Specific Ranges for PIC16F8X series:
  Code Protection off
  0100h to 1FFFh
  1000h to 1FFFh
  0000h to 1FFFh
- The Special Features of the CPU section will describe microcontrollers levels of Code Protection.

Getting Started: Device Configuration

Each microcontroller device has different levels of code protection.

Some devices allow the entire part to be code protected, others only allow partial protection.

For example, the PIC16F87X family of devices will allow 4 different ranges of program memory to be code protected. They are:

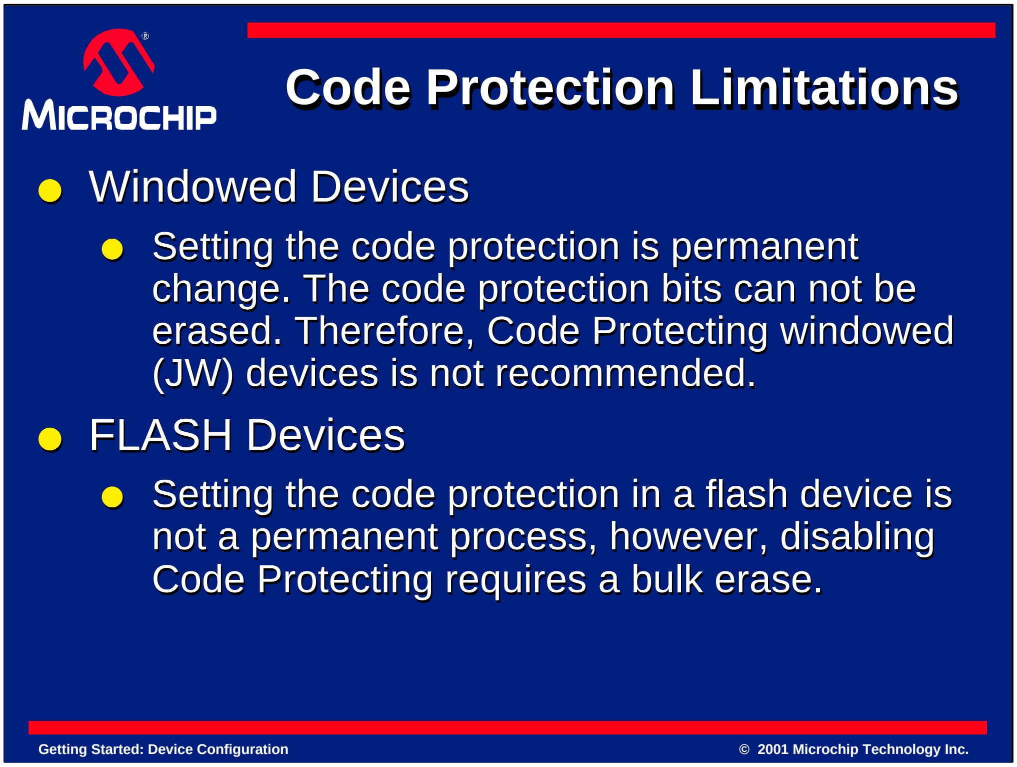
Code Protection off

0100h to 1FFFh

1000h to 1FFFh

0000h to 1FFFh

Therefore, it is best to refer to your device datasheet for code protection ranges that affect you. Code protection is described further in the data sheet under Special Features of the CPU.



Code Protection Limitations

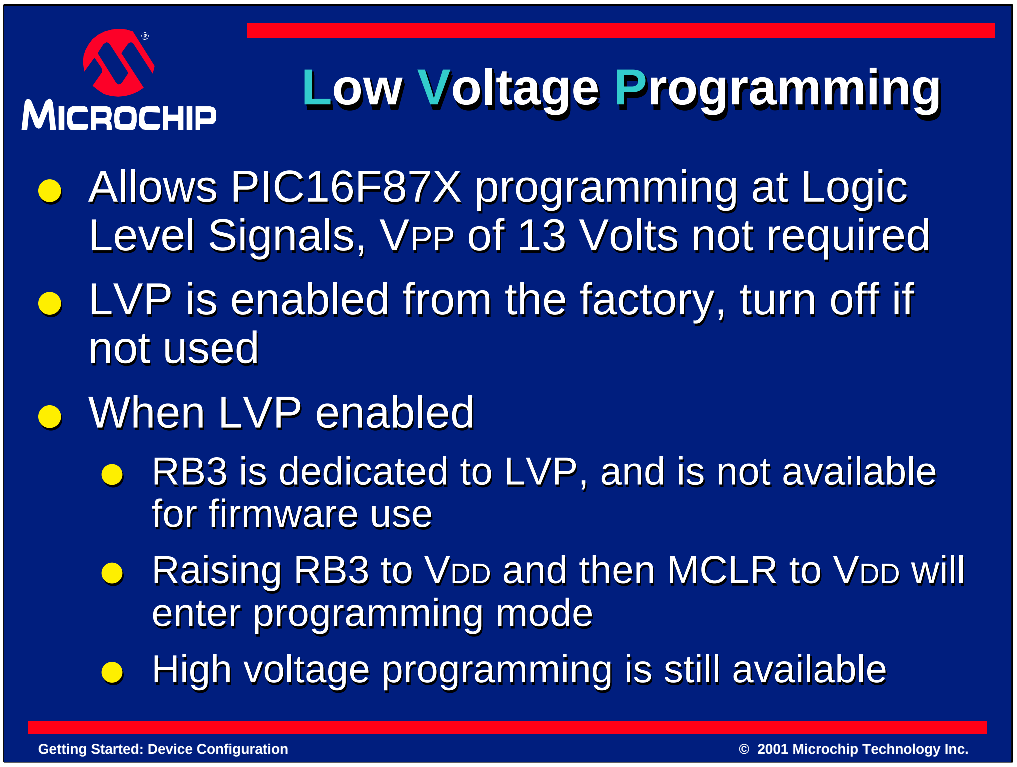
- Windowed Devices
  - Setting the code protection is permanent change. The code protection bits can not be erased. Therefore, Code Protecting windowed (JW) devices is not recommended.
- FLASH Devices
  - Setting the code protection in a flash device is not a permanent process, however, disabling Code Protecting requires a bulk erase.

Getting Started: Device Configuration © 2001 Microchip Technology Inc.

It is important to note that each type of microcontroller has code protection limitations.

For example, setting the code protection on a windowed device is a permanent change to that part. UV erasing the device will not erase the code protection bits. For this reason we do not recommend code protecting windowed devices.

FLASH microcontrollers, however, are different. Setting the code protection on a FLASH device is not a permanent change, but, disabling the code protection will require a bulk erase.



Low Voltage Programming

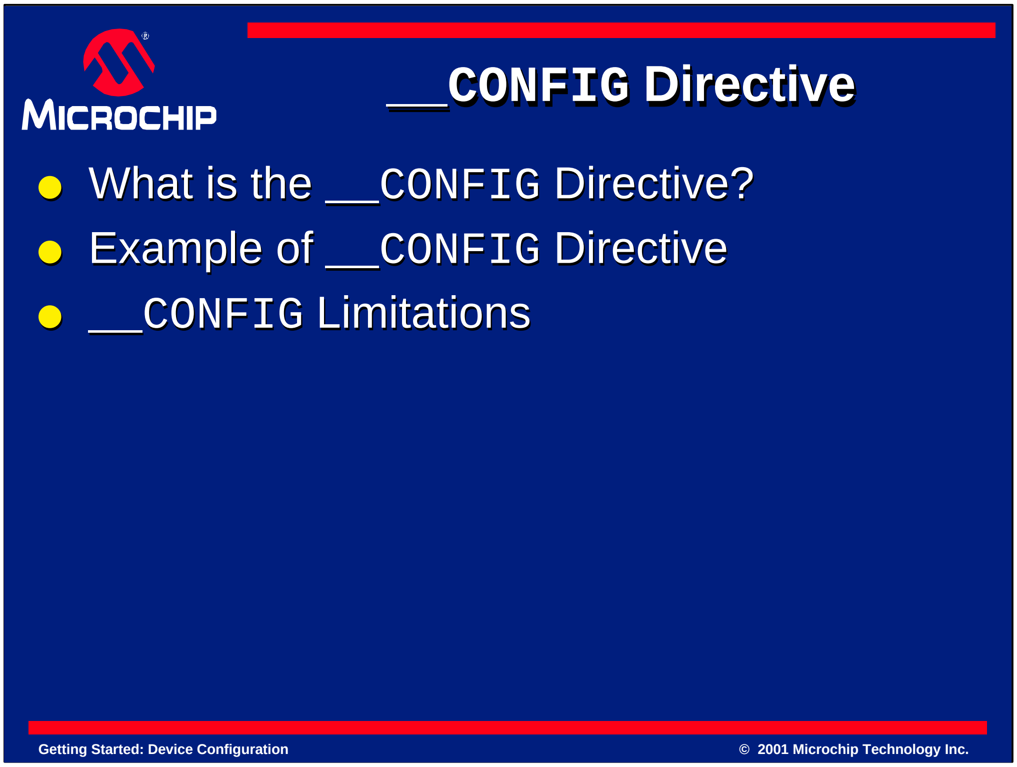
- Allows PIC16F87X programming at Logic Level Signals, VPP of 13 Volts not required
- LVP is enabled from the factory, turn off if not used
- When LVP enabled
  - RB3 is dedicated to LVP, and is not available for firmware use
  - Raising RB3 to VDD and then MCLR to VDD will enter programming mode
  - High voltage programming is still available

Getting Started: Device Configuration

Low Voltage Programming on the PIC16F87X family of microcontrollers can also be enabled through the configuration bits. The Low Voltage Programming feature allows the device to be programmed with logic level signals instead of the traditional 13 Volts on VPP.

Although Low Voltage Programming is enabled from the factory, the user can disable this feature if not used.

When Low Voltage Programming is enabled it dedicates the RB3 pin for Low Voltage Programming. Therefore, rising RB3 to VDD and then MCLR to VDD will enter programming mode. In addition, high voltage programming is still available in LVP mode.



__CONFIG Directive

- What is the __CONFIG Directive?
- Example of __CONFIG Directive
- __CONFIG Limitations

Getting Started: Device Configuration

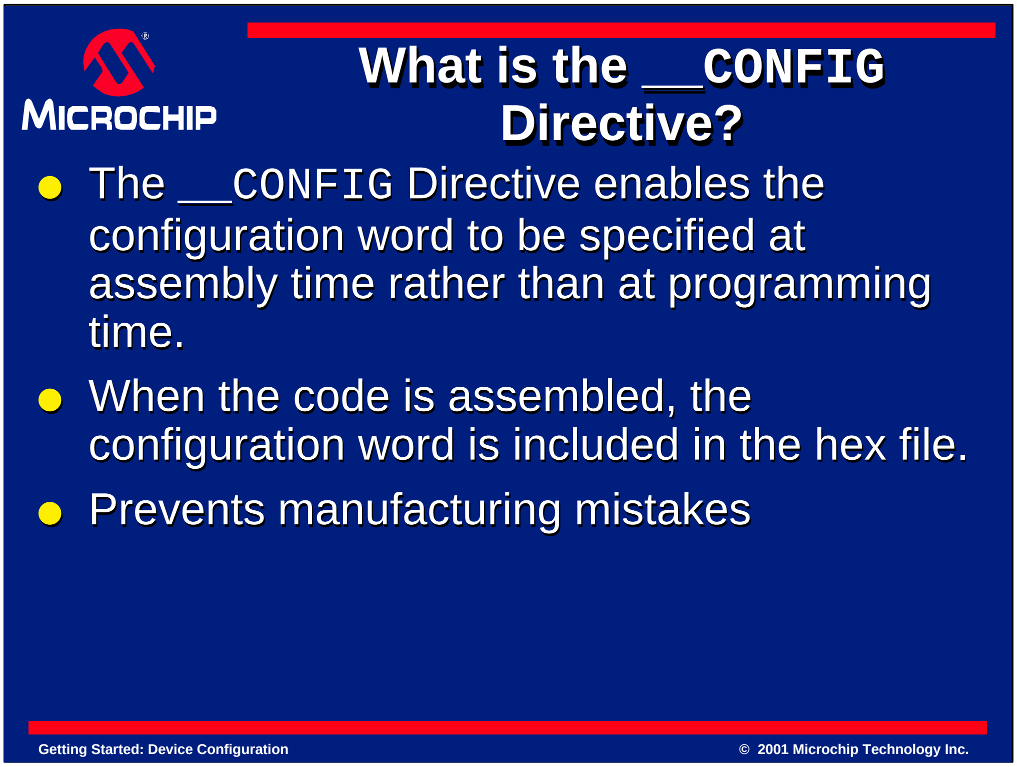
© 2001 Microchip Technology Inc.

Our last topic in this tutorial is the __CONFIG Directive.

We will explain what it is, how it is implemented, and the advantages.

Next, we will give an example using the PIC16F87X series devices, and explain each of the configuration bits.

Lastly, we'll look at some of the __CONFIG limitations. We will discuss what the __CONFIG directive can and can not do.



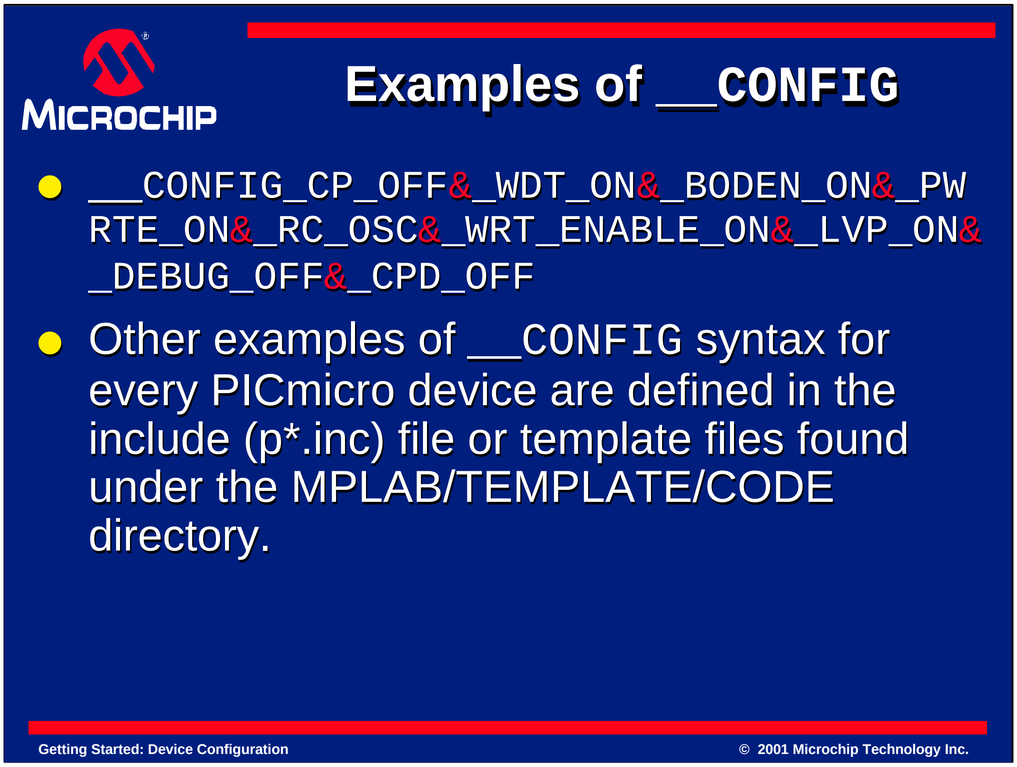
What is the __CONFIG Directive?

- The __CONFIG Directive enables the configuration word to be specified at assembly time rather than at programming time.
- When the code is assembled, the configuration word is included in the hex file.
- Prevents manufacturing mistakes

Getting Started: Device Configuration

The __CONFIG directive enables the configuration word to be present at assembly time rather than at programming time.

When the code is assembled the configuration word is included in the hex file. This reduces the chances that devices will be inadvertently programmed with the incorrect configuration settings.



MICROCHIP

Examples of __CONFIG

- __CONFIG_CP_OFF&_WDT_ON&_BODEN_ON&_PWRTE_ON&_RC_OSC&_WRT_ENABLE_ON&_LVP_ON&_DEBUG_OFF&_CPD_OFF
- Other examples of __CONFIG syntax for every PICmicro device are defined in the include (p*.inc) file or template files found under the MPLAB/TEMPLATE/CODE directory.

Getting Started: Device Configuration © 2001 Microchip Technology Inc.

This PIC16F877 example shows the __CONFIG line defined. The configuration first begins with a space or tab. If a space or tab is not present a warning message will appear stating that the __CONFIG directive is in column 1.

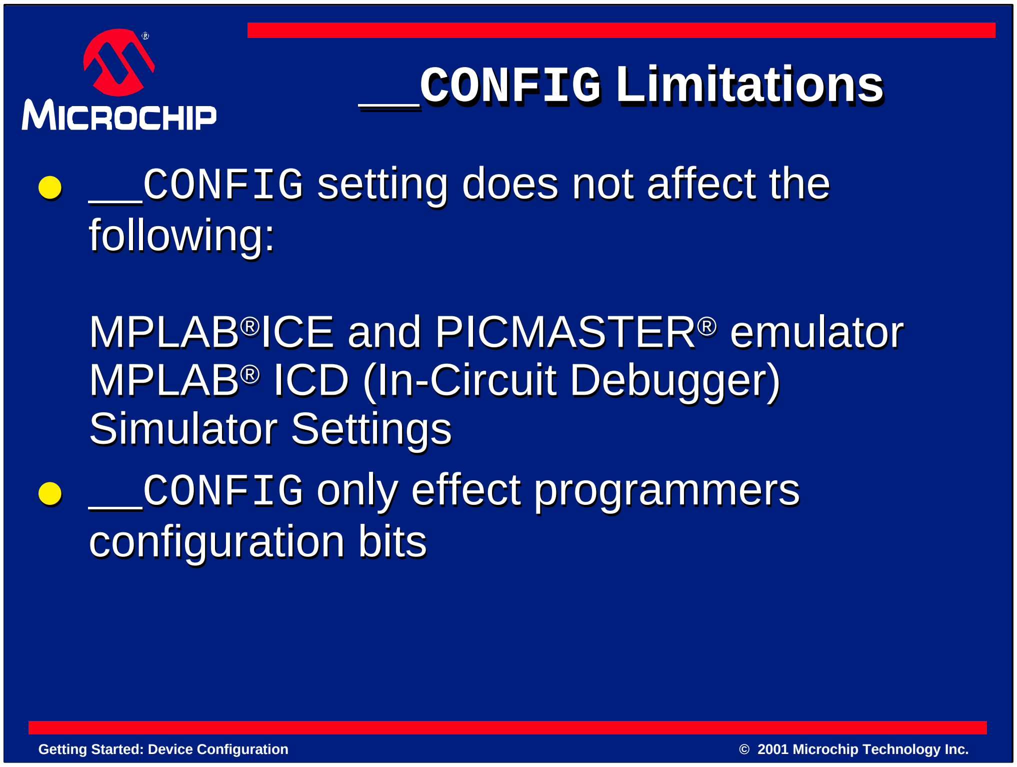
Next add two underscores “_” in front of CONFIG and then the configuration bits name. Any additional configuration bits defined in the include files will have to be separated by an ampersand “&” symbol which allows the configuration bits to be “anded” together.

In this example we have the following configuration bits. They are defined as:

_CP_OFF for Code Protection off, _WDT_ON for Watchdog Timer off, _BODEN_ON for Brown-Out Reset On, _PWRTE_ON for Power-Up Timer Enable On, _RC_OSC for RC oscillator selected, _WRT_ENABLE_ON for Flash Program Memory Write On, _LVP_ON for Low Voltage In-Circuit Serial Programming On, _DEBUG_OFF for In-Circuit Debugging Mode off, and _CPD_OFF for Data EE Memory Code Protection off

Although it is not necessary to specify all, they must all be defined correctly as stated in either the device’s “include file” (All include files start with p*.inc) or template file. The p*.inc files define each of the configuration bits, and the template files include the entire configuration word that was define in the p*.inc. Refer to these files for further information.

The include (p*.inc) files are found under the MPLAB directory, and the template files are found under the MPLAB/TEMPLATE/CODE directory.



__CONFIG Limitations

- __CONFIG setting does not affect the following:

  MPLAB®ICE and PICMASTER® emulator
  MPLAB® ICD (In-Circuit Debugger)
  Simulator Settings
- __CONFIG only effect programmers configuration bits

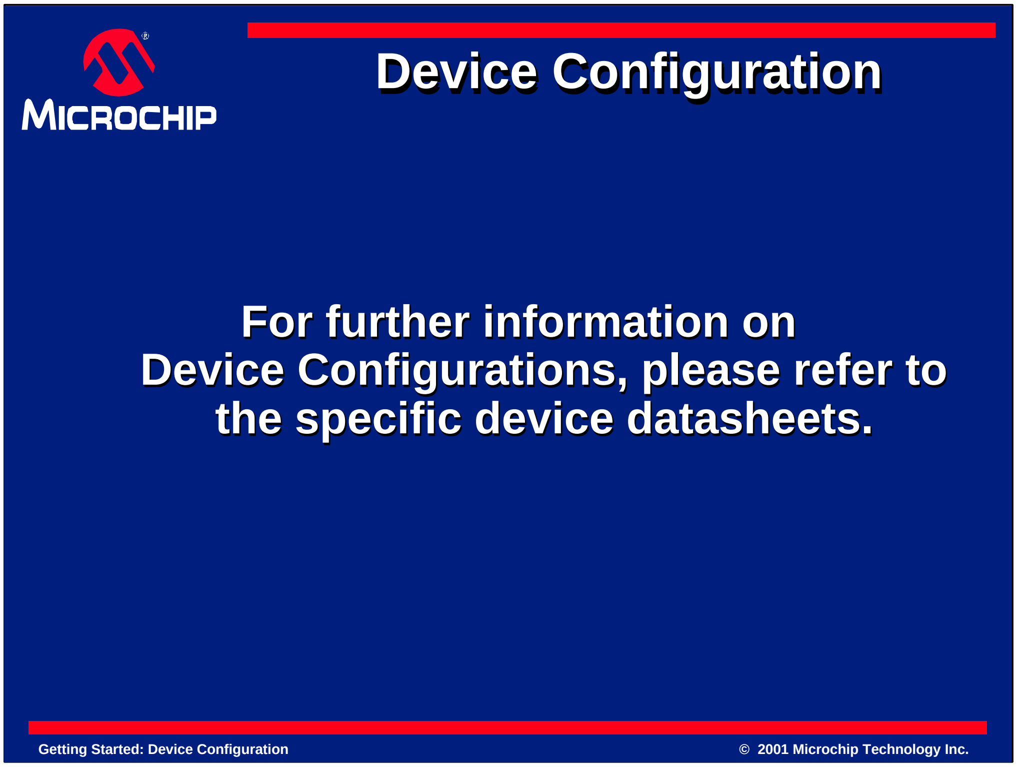
Getting Started: Device Configuration

The __CONFIG mode does have some limitations.

__CONFIG settings will not affect the following:

MPLAB® ICE and PICMASTER® emulator,
MPLAB® ICD, and simulator settings.

__CONFIG only affects the PRO MATE®, PRO MATE® II, and PICSTART® Plus configuration bits.



Device Configuration

For further information on Device Configurations, please refer to the specific device datasheets.

Getting Started: Device Configuration

© 2001 Microchip Technology Inc.

This concludes our brief look at Device Configuration. In addition to your device datasheet, you will also find more information on this topic in pdf format on the Microchip Website. Helpful documents include:

the *Mid-Range Family Reference Manual*,

application notes:

AN588 "PIC16/17 Oscillator Design Guide"

AN606 "Low Power Design using PIC16/17"

Datasheets, Application Notes, Seminar and Workshop schedules, and other helpful information can also be found on the Microchip Website.

“Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. No representation or warranty is given and no liability is assumed by Microchip Technology Inc. with respect to the accuracy of such information, or infringement of patents arising from any such use of otherwise. Use of Microchip's products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights.”