

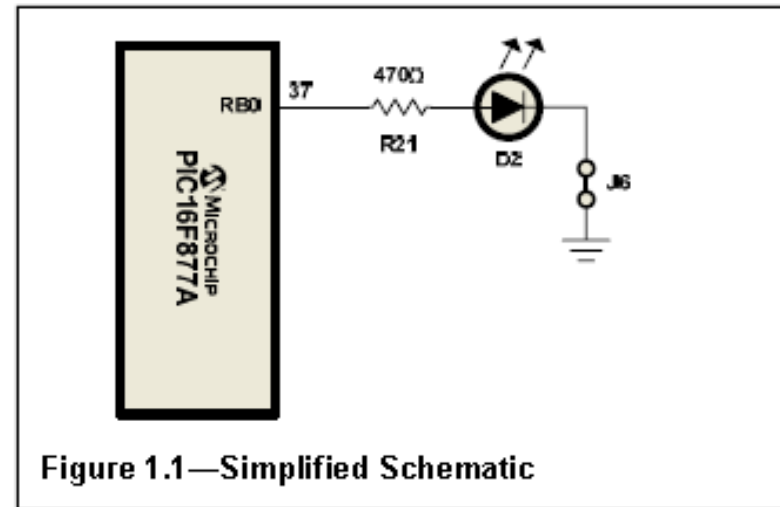
Hello World

Hello World

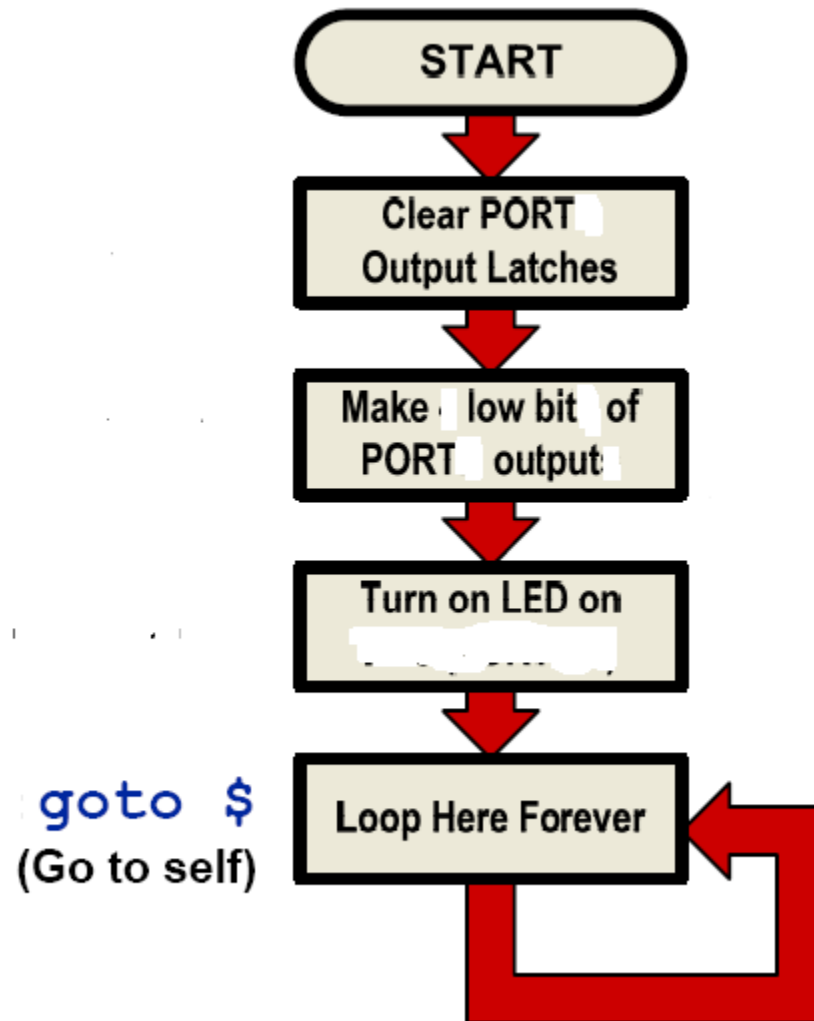
- “Hello World” is typically the first program anyone writes when faced with learning a new programming language or paradigm
- Because of the inherent limitations associated with microcontroller and their lack of a “Standard I/O” to support a screen display our “Hello World” will be limited to just lighting an LED

Lesson 1 --Hello World

- The first lesson shows how to turn on a LED. This is the PIC® microcontroller version of “Hello World” and discusses the I/O pin structures.
- The LEDs are connected to I/O pins RD0 through RD7. When one of these I/O pins drives high, the LED turns on. The I/O pins can be configured for input or output. On start-up, the default is input. The TRIS Special Function Register bits use the convention of ‘0’ for output and ‘1’ for input. We want digital output so these must be configured.



Program Structure

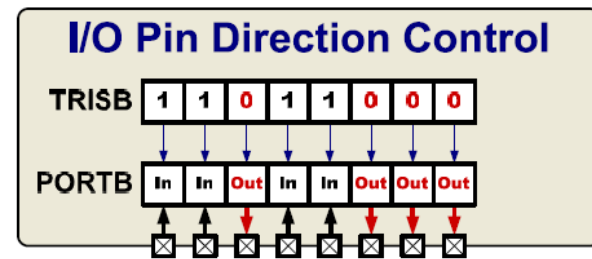


Special Function Registers

TRIS and PORT

- The LEDs are connected to I/O pins RD0 through RD7.
- When one of these I/O pins drives high, the LED turns on.
- The I/O pins can be configured for input or output.
- On start-up, the default is input.
- The TRIS Special Function Register bits use the convention of '0' for output and '1' for input. We want digital output so these must be configured.

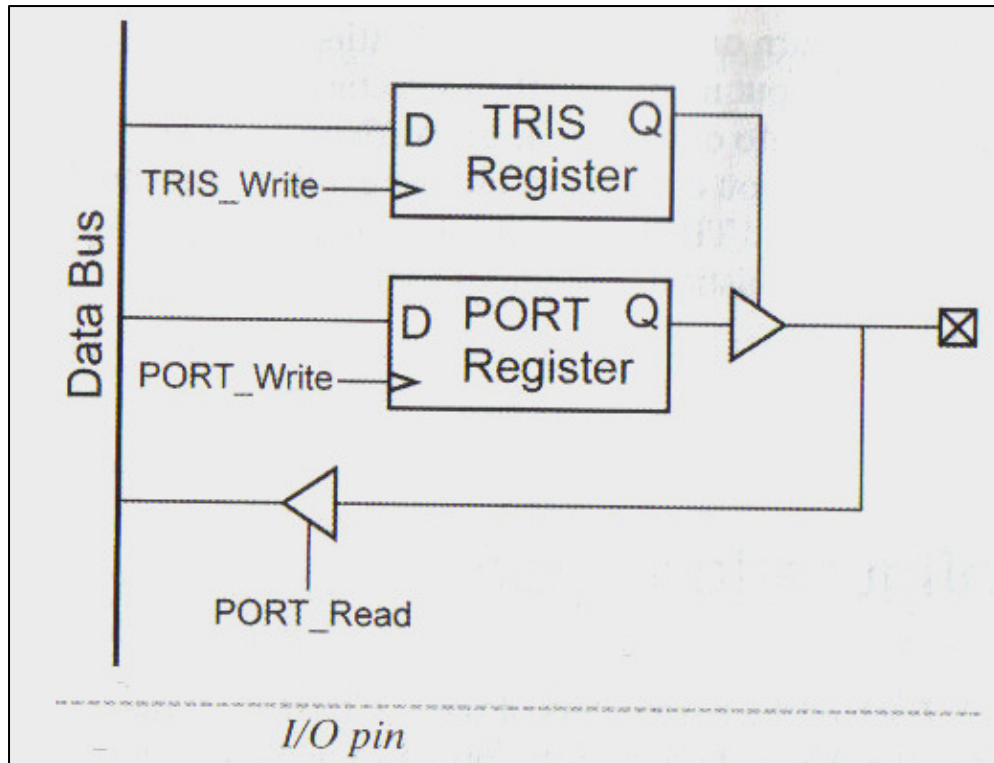
How TRISB relates to PORTB



- To make bit x of PORTB an input, set bit x in TRISB.
- To make bit x of PORTB an output, clear bit x in TRISB.

I/O PIN

Inside PIC



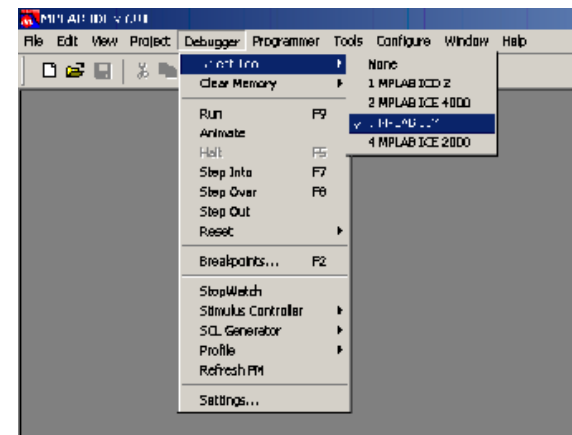
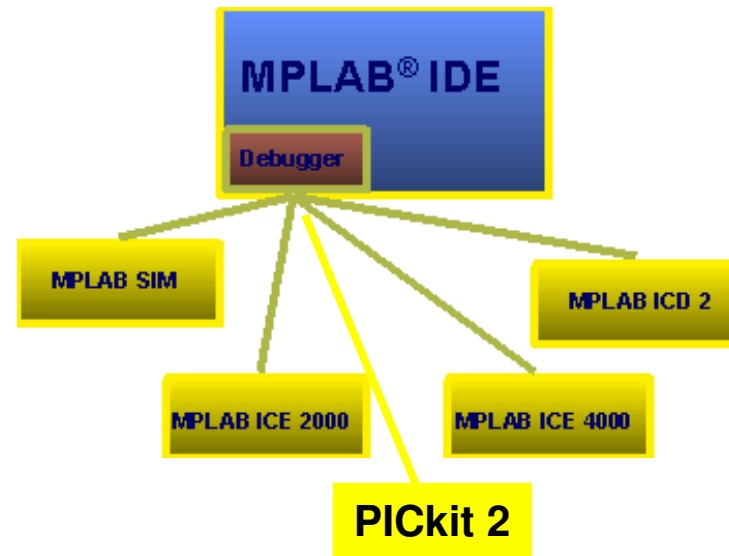
Per pin

Exercise –Execute “Hello World” using Simulator

1. Navigate:C:\EET250\16F887\Lesson 3
HelloWorld with Demo Board\HelloWorld
2. Open source code
3. You need to configure TRISD to 0x00 for
output
4. You need to then set RD7 =1
5. Select simulator
6. Enter code in designated place, build and
reset. Set brekpoint at first line of code in main
7. Open watch to PORTD and TRISD
8. Single step and note changes in these values

Debugger

- The debugger can be a **software program** that simulates the operation of the Microcontroller for testing or it can be **special instrumentation** to analyze the program as it executes in the application.
- **Simulator Debugger**
 - Usually a simulator runs somewhat slower than an actual Microcontroller, since the CPU in the PC is being used to simulate the operations of the Microcontroller. The speed of simulation depends upon the speed of the PC, the PC's operating system, how many other tasks are being run in the background, and the complexity of the simulation
- **Hardware Debuggers and Programmers**
 - A programmer simply transfers the machine code from the PC into the internal memory of the target Microcontroller. The Microcontroller can then be plugged into the application, and it will run as designed.



Using Real Time Debug

PICkit™ 2 Debug Express

- Programmer and Debugger !
- Limited to one break point
- Low Cost

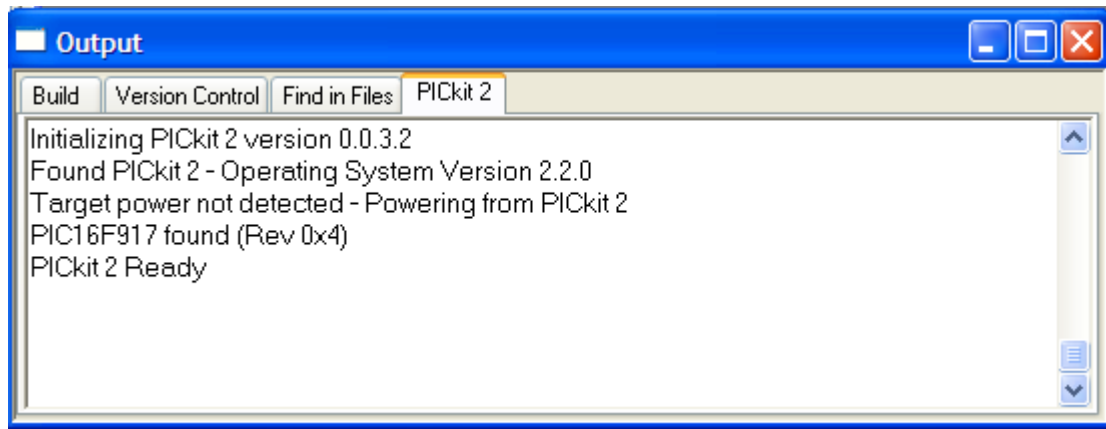
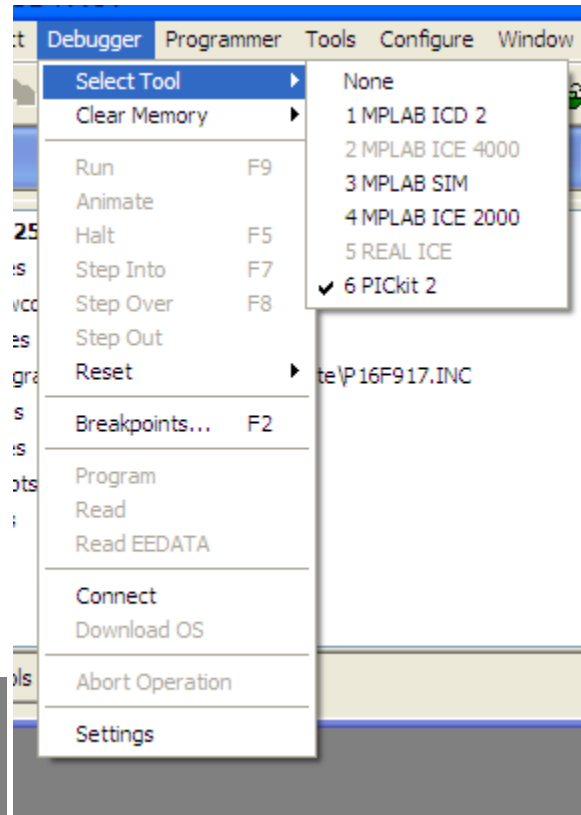
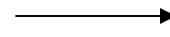


Exercise –Execute “Hello World” using PICKIT and Demo Board

1. You should already be at
:C:\EET250\16F887\Lesson 3
HelloWorld with Demo Board\HelloWorld
with project open
2. Hook USB to PICKIT2 and plug Demo
board into PICKIT2
3. Note leds on PICKIT2 indicates power
4. Follow the next steps as indicate in
slides to execute code on hardware

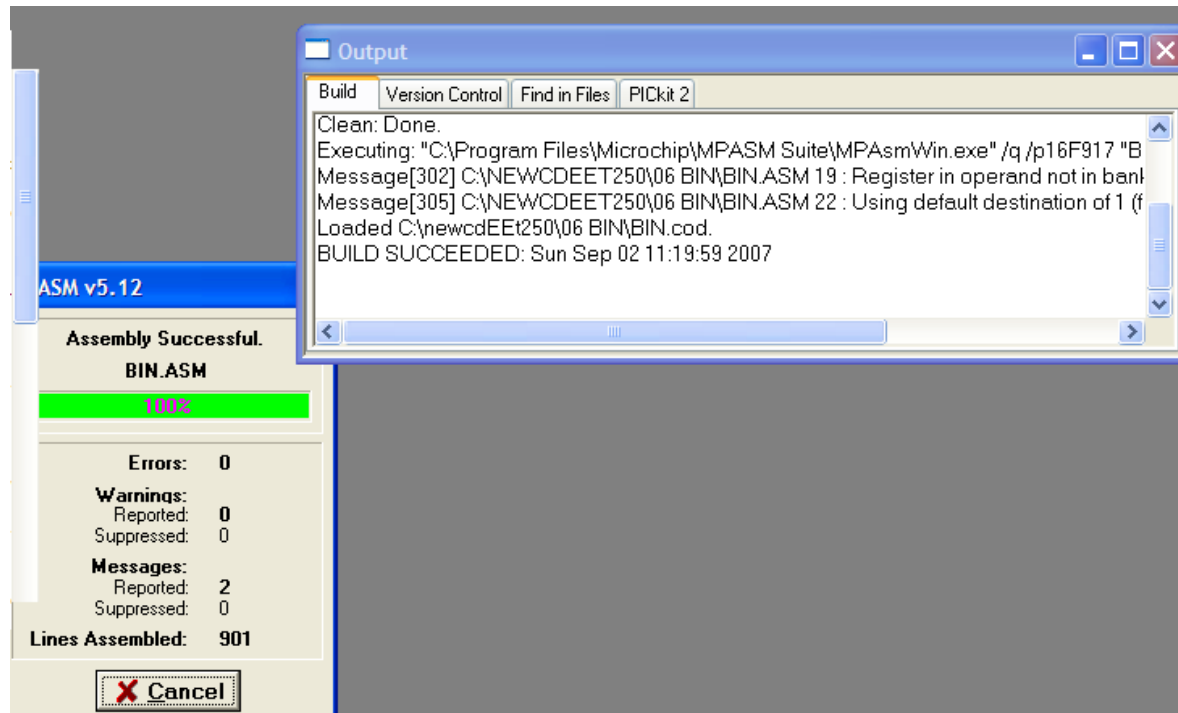
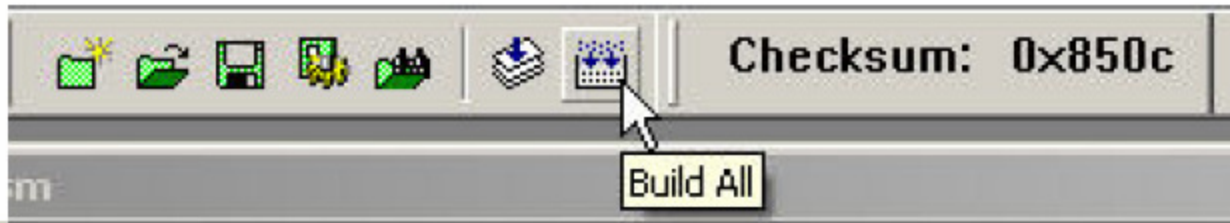
Using PICKIT2 DEBUGGER

- Plug in the PICKIT2
- Select PICKIT2 rather than MPLAB SIM
- Leave watch window up
- Output should appear as follows

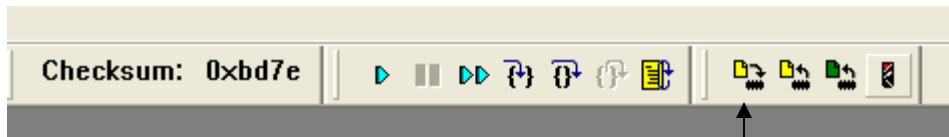


Recompile code with PICKIT2 selected

Compile Code

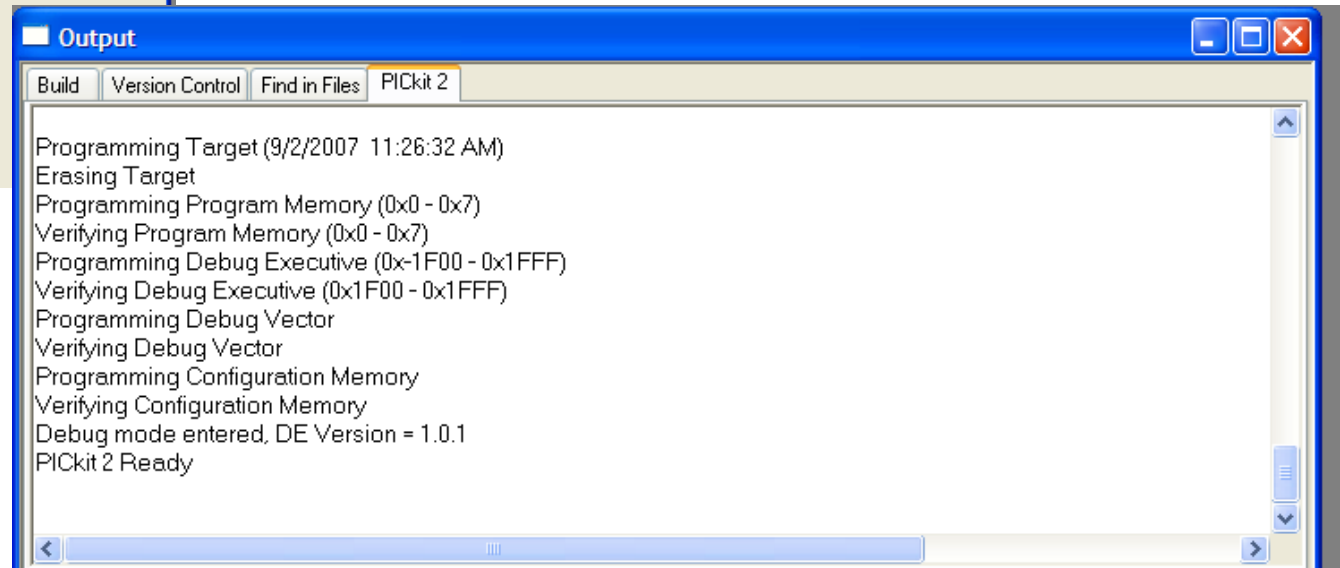
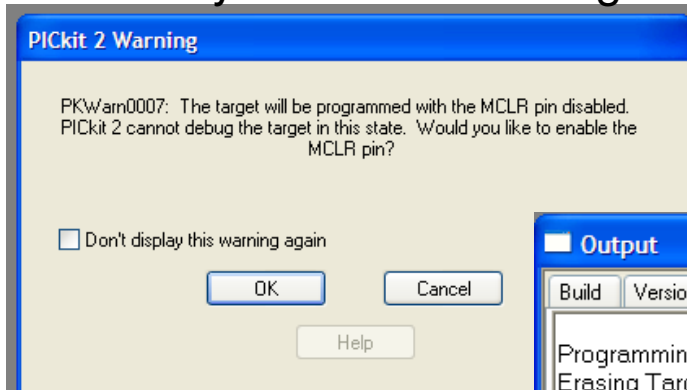


Program Demo Board PICKIT2



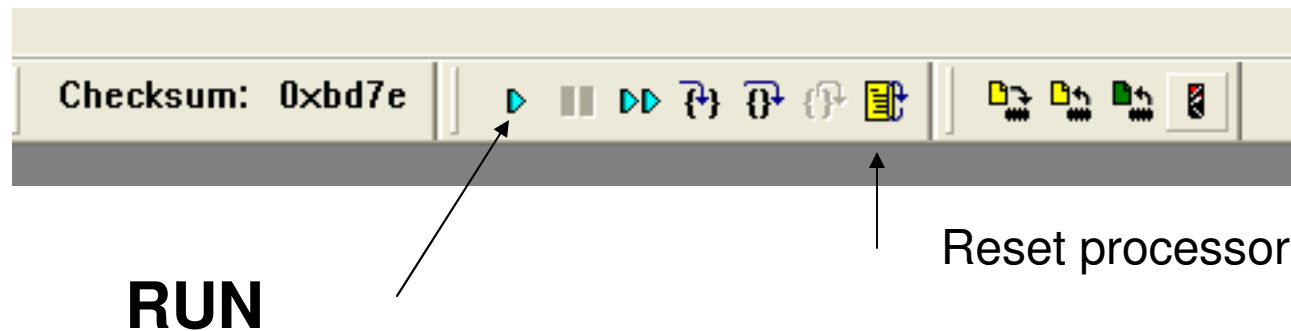
Program Target Device

Say ok to this warning



Output window should look like this.

Running program using PICKIT2



Step Through Code

