**Lesson 1: Introduction & Hello World Program**

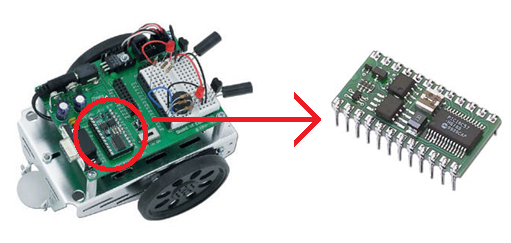
Needed

* Boe-Bot
* Computer with BASIC Stamp Editor program
* USB cable

(Put in an introduction to the Cyber Science Curriculum)

“Hello World” is a program that allows a user to establish initial communication between the Boe-Bot and the computer.

The Boe-Bot is a robotics kit with a microcontroller. A *microcontroller* is a small computer in a single integrated circuit. The primary purpose of using this type of “computer” is for low cost application.

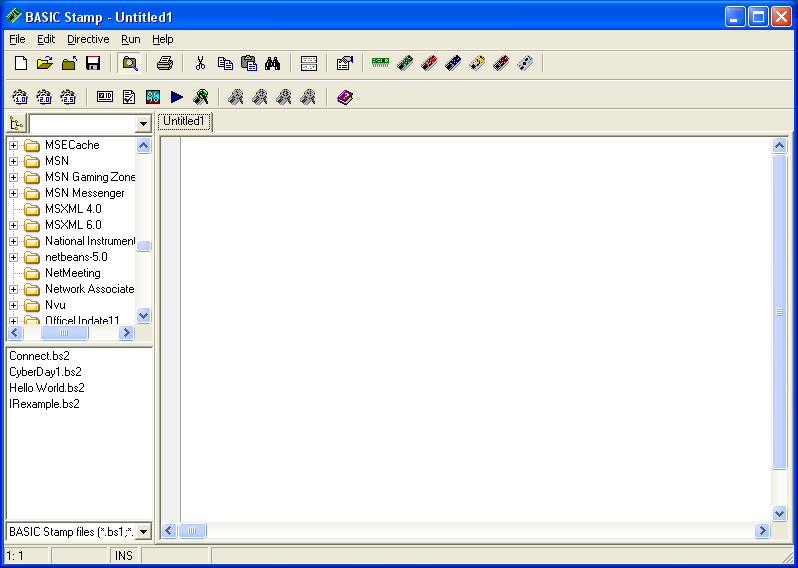


The Boe-Bot’s microcontroller is a BASIC Stamp. BASIC Stamp is actually a PIC (Peripheral Interface Controller) microcontroller. (Microchip PIC16C57)

\*For this initial program, the Boe-Bot does not have to be assembled. If you are using an unassembled Boe-Bot, you will need the board with the BASIC stamp installed and the battery pack with batteries plugged into the board. For homework the students can begin assembly of the Boe-Bots, which will be due in four class periods.

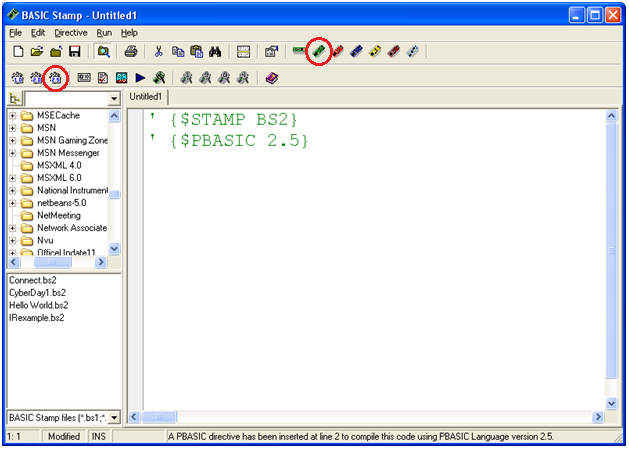
To get started programming:

1. Open the BASIC Stamp Editor program.



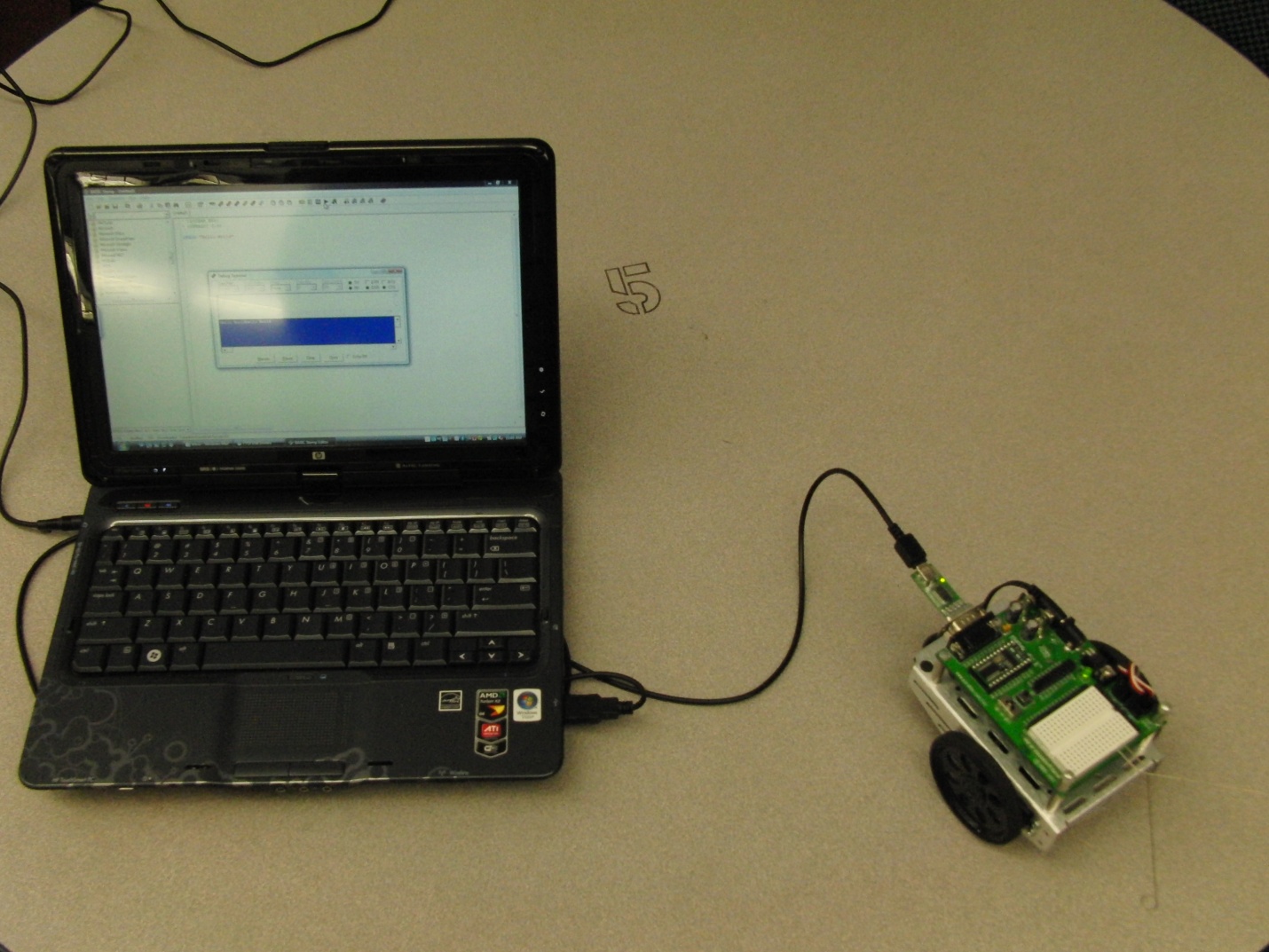
(You may want to click the “Do not show” option on the dialog boxes to eliminate the message from showing every time.)

1. Select “Stamp Mode: BS2” and “PBASIC Language: 2.5” from the buttons in the toolbar.



* Use BS2 (BASIC Stamp v. 2)
* Use PBASIC version 2.5

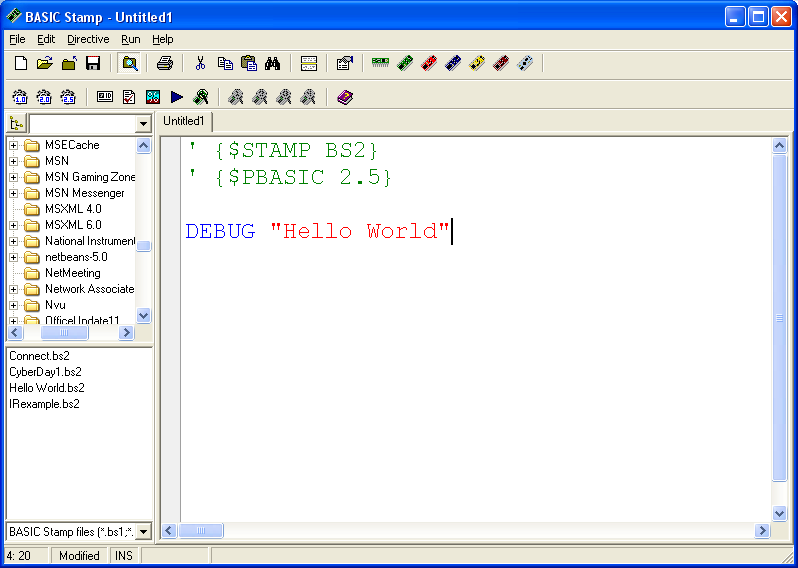
1. Connect Boe-Bot and laptop using cable.



To send the program from the computer to the Boe-Bot, press the “Run” button to send the PBASIC Program to the BASIC Stamp where it is executed. If you try to run the program at this point, there is an error message “Nothing to tokenize” because there is no program to execute.

Green lines of text in the PBASIC editor, created when typing an apostrophe at the beginning of a line of text, represent comments within the program. These lines are not sent to the Boe-Bot. They are just used for notation. This is a good programming approach because it allows the programmer to see the information without taking up program memory.

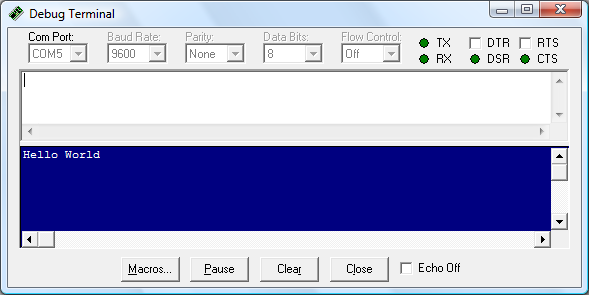
Now let’s write a simple program that will allow the Boe-Bot to send information back to the computer. The DEBUG command sends communication from the Boe-Bot to the computer and displays the message in quotes.



* DEBUG = send text to the “DEBUG” terminal
* “Hello World” = text to be displayed

Notice the colors of the text:

* Green = comments
* Blue = command recognized by PBASIC
* Red = string of characters

Now that your program is written, hit the “Run” button to send the PBASIC program to the Boe-Bot and have the Boe-Bot send the “Hello World” string back to the computer. 

As you can see, “Hello World” is a very simple program, but it shows that we can communicate between the computer and the microcontroller.

Simple Programming

Now that you understand the Boe-Bot communication through the simple hello world program, there are a few concepts we need to discuss before we get into more complex programming. A major part of programming is assigning variables. Add some information on binary systems. If you have something in a program that will vary in value, you can define that term as a variable.

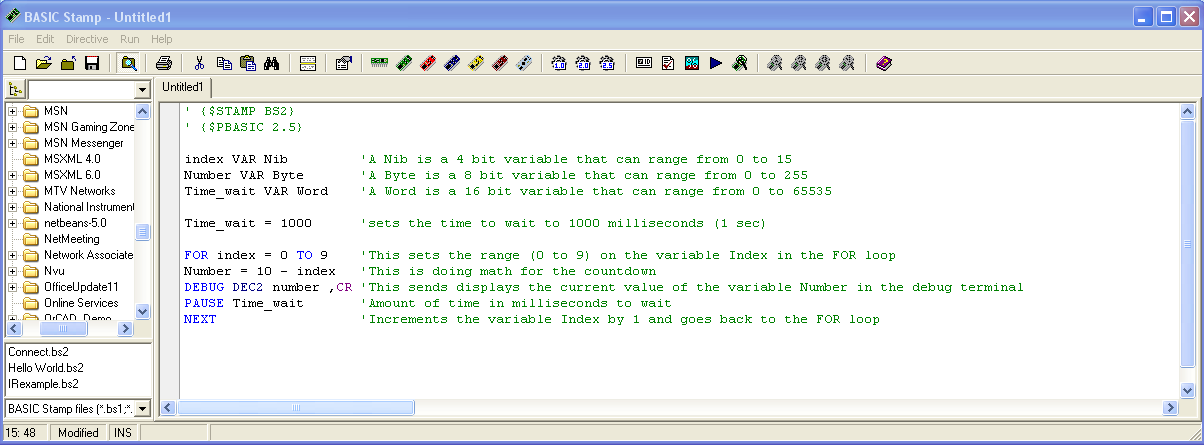
*Variable* – an assigned symbolic representation of a value that can be changing.

Think of a variable in math. You can have an equation , where m and b are constants (slope and y-intercept) but the y and x are variables that can change values. The same idea is used here in programming. You can assign a changing component in a program a variable name then define that variable to change accordingly.

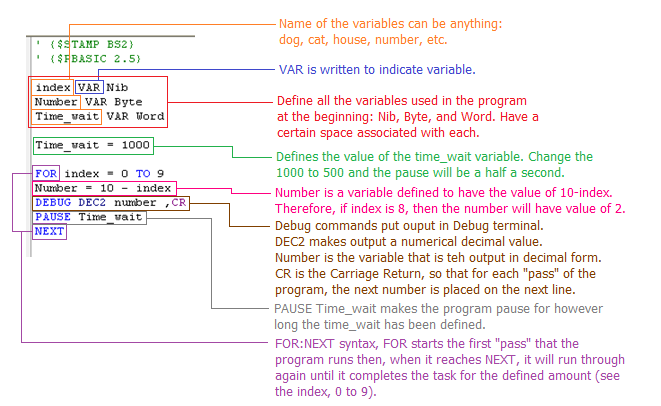
A variable can be defined as one of four capacities: bit, nib, byte, word. A bit stores up to two pieces of information, 0-1. A nib stores up to 16 pieces of information, 0-15 (2bits). A byte stores up to 256 pieces of information, 0-255 (8 bits), and a word stores up to 65,536 pieces of information, 0-65,535 (16 bits). When defining a variable you must define accordingly depending on the amount of information being stored.

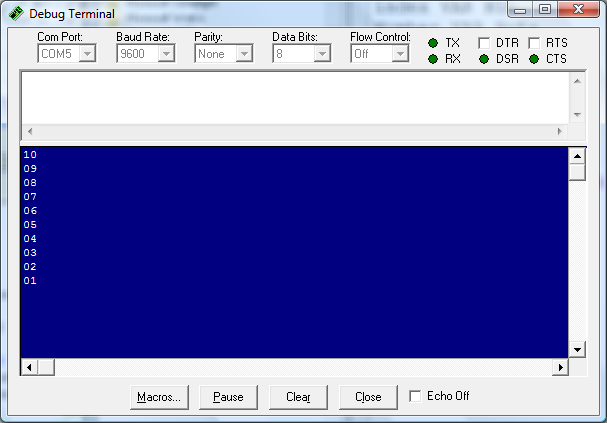
The following program illustrates the use of variables by making the Boe-Bot act as a countdown machine. Noticfe the use of a syntax called FOR:NEXT. The FOR:NEXT syntax executes a sequence of commands for a set number of times. Check out the following program which uses a FOR:NEXT syntax to perform a count down from 10 to 1. The variables being defined in this program are index, Number, and Time\_wait.

The green lines are simply comments about each line in the program and are not essential to the program’s performance.



Let’s take a closer look at each of the lines in this program in more general terms:





You can see now that program is set up to allow for easily changing the length that the program pauses where you only need to change the definition of Time\_wait.

Taking advantage of the variables, is important especially when you have long programs, for instance, what if the countdown started from 1000, with a half second pause in between each count. You can see how changing the pauses would get cumbersome.

One of the key advantages of computers is that they can reliably do repetitive tasks. Now with this new variable terminology, you can easily make the program perform such a task by simply changing the variables and a couple of lines in the code. What do you think you would change in order to count down from 1000 with a half second pause in between numbers?

