

Lab 3: Introduction to the Boe-Bot Platform

Objective:

- To become oriented with the Boe-Bot Platform
- To introduce basic programming concepts
- To demonstrate how a microcontroller can control hardware

Background

Boe-Bot:

The Boe-Bot (“Board of Education”-Bot) is a basic robot, designed by Parallax, Inc., that integrates various topics of electrical engineering together, namely robotics, circuits, and programming. It has a built-in breadboard that can be used in conjunction with a computer program. Circuits are programmed using the 16 pins that line the breadboard along the left. These pins can have two states – 0 or 1, or false and true respectively. There are also two servos that run the wheels and are connected to the rest of the robot through pins 12 and 13 – pin 12 for the right servo and pin 13 for the left.

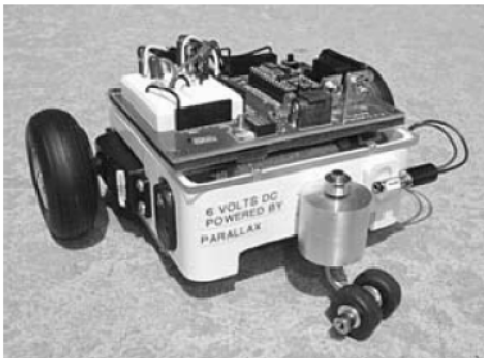


Figure 1: The Initial Boe-Bot Prototype

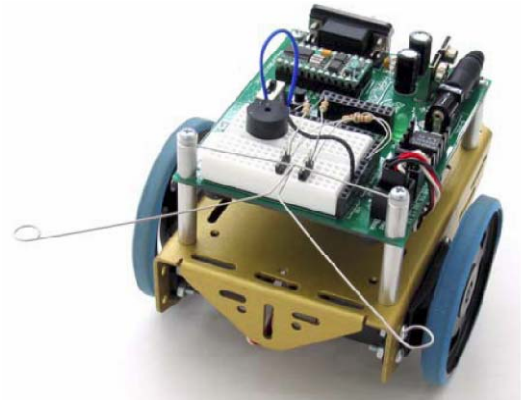


Figure 2: The Boe-Bot, outfitted with whiskers

The Boe-Bot is a major consumer of power. A Boe-Bot running continuously performing an average list of tasks will run through a pack of 4 batteries in about 30 minutes. To lessen the cost of batteries, you will be using an AC adapter to perform basic tasks like downloading programs and running some tests.

The Javelin Stamp:

The Javelin Stamp, a white computer chip located next to the USB port, controls the Boe-Bot. The chip's pins correspond to those located on the breadboard. The chip can be programmed to perform a variety of functions, the most basic of which are reading and writing to the pins (meaning reading and writing a value – 0 or 1 – to the pin). The Javelin Stamp is programmed using Java, with the main library class being the built-in CPU class. This CPU class has methods designed to control every facet of the Boe-Bot, from the breadboard to the servos. For the ones needed for this lab, see the Java Programming section.

WARNING
THE JAVELIN STAMP CAN GET QUITE HOT AFTER EXTENDED PERIODS OF USE. USE CAUTION WHEN HANDLING THE BOE-BOT.

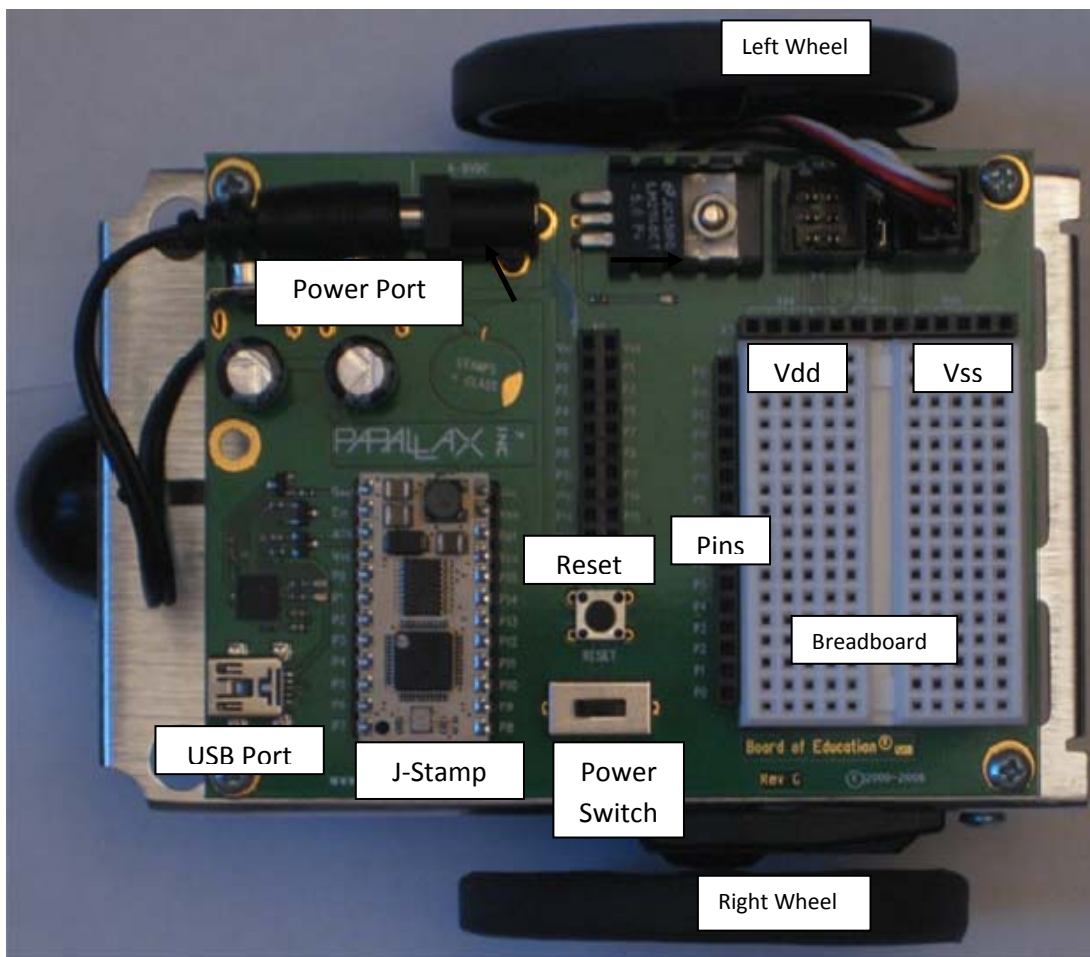


Figure 3: Boe-Bot Circuit Board

Controlling the Circuit Board:

The sixteen pins along the left side of the circuit board are the liaison between the Javelin Stamp and the breadboard. Each pin can hold one of two values, a 0 (false), or a 1 (true). These values are either dependent on the function of a given circuit (e.g. if an LED is on, then the value is 1), or is written to the pin by the Javelin Stamp. A program, using the `readPin` method in the CPU class, can read the value of the pin. This will be quite useful when you need to figure out if a light is on or if a switch is pressed. The servos are connected through a different port to pins 13 (left) and 12 (right). You will use these pins in your program when you want to control the wheels.

The Circuit Board:

The breadboard on the top of the Boe-Bot operates just like the breadboard you used in the capacitor lab before. Each row of five contacts is the same connection – if a wire and a resistor are both connected to the same row, and then they are a node in the circuit. Keep in mind that there are two columns of independent rows, divided by the empty space in the middle. The power source for the breadboard is located along the top. The first five pins, labeled ‘Vdd’, is the ‘positive’ terminal and provides 5V, while the last five pins, labeled ‘Vss’, is the ‘negative’ or ‘ground’ terminal and is considered to be 0V. These notations will be used in circuit diagrams.

Programming:

Programming the Boe-Bot is just like writing any other Java program. You will need a class, in which you will place a main method, in which your program goes. For more details on basic Java syntax please refer to the lecture notes. You can use library classes, which can contain methods that you can use in your program. The most important library class is the CPU class, which is built-in the Javelin Stamp. To use this class, you will need to import it into your program by placing `import stamp.core.CPU` at the top of your code even before the class declaration. This will instruct your program that the CPU class exists and is to be used. The following are important methods you will use in your program can be found at:

<http://www.seas.upenn.edu/~ese112/fall09/boebotResources/orientationcommands.html>

Note: The ‘pin’ variables also defined in the existing library with variable convention *pin#*, where # is the pin number. To write or read to any pins on the boe-bot you need use the following syntax i.e. `CPU.pin#` (e.g. `CPU.pin15`)

The following are helpful hints when programming the Boe-Bot:

- When writing the main method, do NOT put “String [] args” as the input to the method as you would normally do in a regular Java program– it will not work. Your main method header should be: `public static void main ()`

Javelin Stamp Integrated Development Environment (JSIDE):

In order to program the Boe-Bot, you will have to use the Javelin Stamp IDE. This IDE has the ability to edit, compile, and to download new programs and communicate with the robot. All of the RCA (Moore 101) and Ketterer (Moore 204) computers have the JSIDE installed. To install it on your personal computer (Windows only; Mac users will have to use the lab computers or dual boot or work with partner with Windows to use the JSIDE) follow the set up instruction from:

<http://www.seas.upenn.edu/~ese112/fall09/boebotResources/JSIDE.pdf>

Helpful hints:

1. If the Boe-Bot doesn't respond to your commands properly, try pressing the "RESET" button on the top a couple of times and try again. If you keep having problems, ask for help.
2. In order to download programs, the power switch must be set to '1'. Switch position will be used in the future to power the wheels.
 - If the wheels don't turn or turn in a pulsing fashion, your batteries may be dead. Ask for a new set.

Materials

- Boe-Bot unit with Javelin Stamp
- 4 AA batteries and AC Adapter
- USB cable
- Javelin Stamp IDE (from software CD)
- (2) 220Ω Resistors
- (2) LED lights (red)
- Couple of wires

Pre-Lab Questions

1. How many Input/Output (I/O) pins does the Boe-Bot have?
2. What two states or values can the I/O pins have?
3. Give definition of:
 1. Literal
 2. Variable
4. What is difference of == vs. = operator?
5. If variable p0 is of type boolean then what is the outcome of !p0
6. What is the syntax for main method for a Java Program for the Boe-Bot platform?
7. To use the existing methods and variables from the CPU library, what special syntax do need to include in your program before using method or variable? Give the command and state where in the program you put it.

Lab Instructions

You will be writing several programs in this lab. After each exercise, have a TA sign your checklist. At the top of all your programs, place the following header (as a multi-line comment), with all fields filled of course:

```
/*
 * Name(s): Include your partners name as well
 * Date:
 * Lab Section:
 */
```

****Make sure you save all your programs for submission either on external drive/usb or on your Penn seas accounts****

You can download all files you need all at once by download: [lab3.zip](#)

Part I – Using Boe-Bot programming interface

1. Open the file [HelloWorld.java](#) to compile and execute the “Hello World” program that will command the Boe-Bot to output a certain phrase (such as Hello World) back to the computer. Note the format of class structure and format of the main method syntax.
2. Compile and download (use the Ctrl+R command) the program to the Boe-Bot. An LED next to the USB port on the Boe-Bot should flash red and blue while downloading. (If you’re using Dr. Java to code, make sure you use the Javelin Stamp IDE to download)
3. After downloading, the Boe-Bot should send your phrase back to the computer in the “Messages” window. If did this not work then you program have compiler, usb port set is wrong or forgot to turn on the switch to position 1.

Part II – Practice with Conditional statements in Java

1. Download *Conditionals.java* from:
<http://www.cis.upenn.edu/~ese112/fall09/java/if.html>
Complete the program as specified.
2. Discuss the answers to the remaining questions on the same website. Put your answers as comments in the file *Conditionals.java*

Part III – Practice with Loops structures in Java

1. Go to the following website and do the exercise and the questions at:
<http://www.cis.upenn.edu/~ese112/fall09/java/controlsloops.html>
Again put your discussion questions as comments in the file `PracLoops.java`
2. Now that you're acquainted with the programming aspect, we'll move to the hardware:
 - a. Collect the following materials from the parts cabinet:
 - a. (2) 220 Ω Resistors
 - b. (2) LED lights (red)
 - c. Couple of wires
 - b. Build 2 basic LED circuits, one for each LED, following Figure 3. The P# notation refers to the I/O pins along the left of the circuit board.

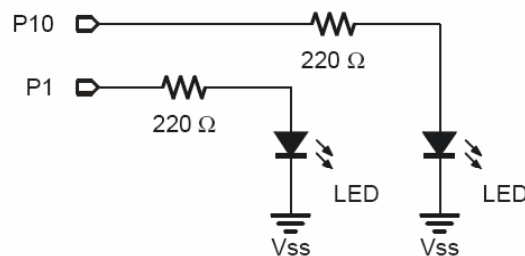


Figure 4: LED Circuit

Note: P10 and P1 will act as output for this program, i.e. the Javelin controller will output a voltage to turn off and on the LED.

Once you have built the circuit, that makes both LEDs turn ON at the same time and then OFF continuously. The period between ON and OFF is 1 second. Write your program in file called [FlashingLEDs.java](#). Note: To stop the flashing action, put the switch to 0 position.

Hints:

- i. To use a pin in a command (e.g. if you want to write '1' to pin 12), you have to use the pin variables in the CPU class. Put the import statement as discussed in the Programming section before the line above statement "public class..". Each pin has a variable, from pin0 through pin15. For example, to use pin12 in your program, you would say `CPU.pin12`.
- ii. To turn an LED on, the pin's value should be 1
- iii. To turn an LED off, the pin's value should be 0
- iv. Use the already implemented methods or commands described in the Programming section to accomplish this task. Note: for *delay* method, carefully read the units for input parameter. Hint: some calculation needs to be done to get the delay of 1 sec between ON and OFF cycle.

Part IV – Methods

1. Go to the following website and do the exercise and the questions
<http://www.cis.upenn.edu/~ese112/fall09/java/staticmethods.html>

Post-Lab Questions

There is no official report of this lab.

Submit all your Java programs from Part II-IV to Blackboard Digital Drop Box in one zipped folder using the format on the ESE112 website under the Course Information section.

Note: Questions that ask you to discuss answers can be put as comments in the files in those sections.