Controlling a Boe-bot using Visual Basic 2008 Express Edition – Jan 24 2009

Please note that this is not a tutorial on how to use VB 2008, or how to program a Parallax Boe-bot (BB). It is simply a guide on how to create a bridge between the two. It is assumed that you know some essentials for programming on either side. If you would like to brush up on your VB skills, a great tutorial can be found here:

http://www.vbtutor.net/

The BB manual can be also be downloaded at the following link:

http://www.parallax.com/Portals/0/Downloads/docs/books/edu/Roboticsv2_2.pdf

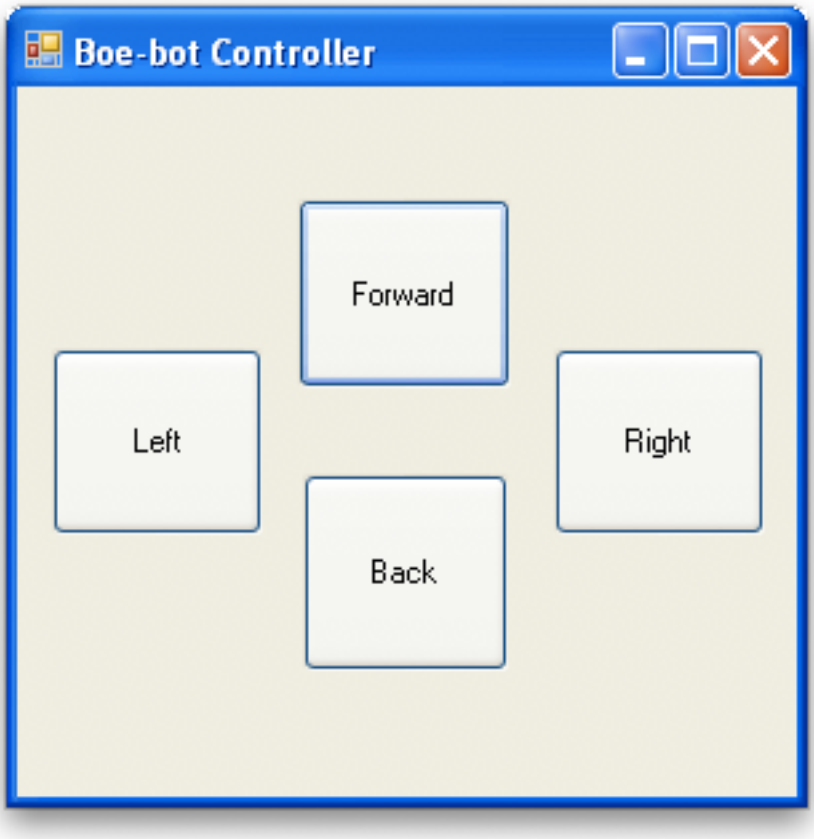
Lets begin on the BB side. After the robot is fully assembled and tested we start by writing subroutines for moving forward, backwards, turning left and right. If you set up the BB using the manual above the left servo is connected to the pin 13, and right to pin 12. This will result with the subroutines resembling the following:

```
FORWARD:
   PULSOUT 13, 850
   PULSOUT 12, 650
RETURN

LEFT:
   PULSOUT 13, 650
   PULSOUT 12, 650
RETURN

RIGHT:
   PULSOUT 13, 850
   PULSOUT 12, 850
RETURN

BACK:
   PULSOUT 13, 650
   PULSOUT 12, 850
RETURN
```

Before we connect the BB to the VB program we need to write a program to take serial data from the user and drive the robot. The easiest was to do this is with the debug window. The program simply reads in a single character using the DEBUGIN command, and stores it to a variable. Then, using the SELECT…CASE command we can associate the letters “W”, “A”, “S” and “D” to moving forward, left, back, and right (Just like in a video game!). Here is the resulting program:

```
char  VAR Byte

DO
  DEBUGIN STR  char\1
  SELECT  char
    CASE "W": GOSUB FORWARD
    CASE "A": GOSUB LEFT
    CASE "S": GOSUB BACK
    CASE "D": GOSUB RIGHT
  ENDSELECT
LOOP
```

Our next step is the design the VB interface to drive your BB. Adding four buttons to your controller can easily do this. It should resembles this:



In order to talk to the BB a serial port needs to be inserted into the VB program. The serial port button can be found under the components tab in the toolbox. After adding it, the settings should not need to be changed except to change the COM port your BB is connected to. Before the VB program can talk to the BB, the serial port needs to be opened. Start by double clicking on the background of your VB program, this will switch it to the code side. The curser will enter the function called Form1_load. Just type SerialPort1.Open() and the COM port will open when the program opens.

To send characters to the BB you have to start by double-clicking on each button, this will jump you to the code side and into the rub routine that runs when each button is pressed. For example, double click on the forward button, and enter the phrase SerialPort1.Write("W"). This will write

the character “W” whenever the forward button is pressed. You can now continue with the other three buttons.

So the BB can hear the VB program, only one line of code needs to be changed. Instead of using DEBUGIN, we will change to SERIN. The dedicated COM port when going through the programmer for the BB is 16. The standard baud rate through VB is N9600. While using the SERIN command the baud rate constant is 16468. After changing to the DEBUGIN command the program should resemble the following:

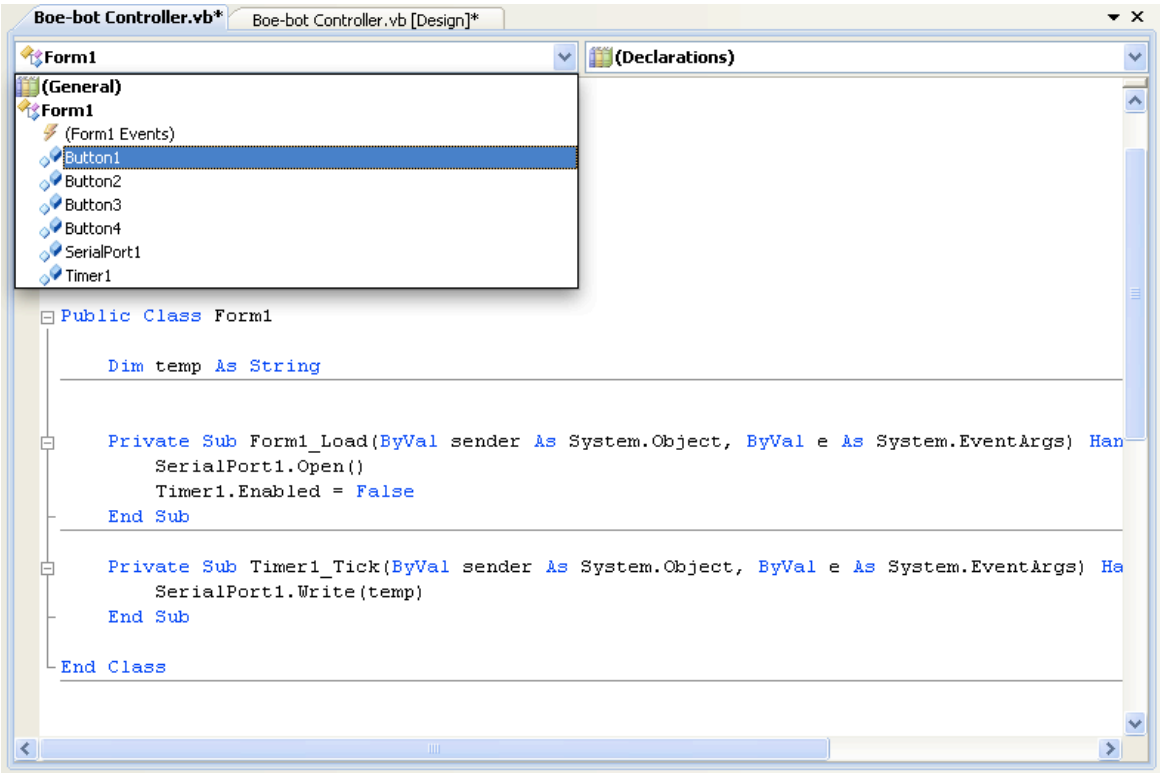
```
char  VAR Byte

DO
  SERIN 16,  16468, [STR char\1]

  SELECT  char
    CASE "W": GOSUB FORWARD
    CASE "A": GOSUB LEFT
    CASE "S": GOSUB BACK
    CASE "D": GOSUB RIGHT
  ENDSELECT
LOOP
```

Now when you run the VB program and click on the buttons the BB should scoot around accordingly.

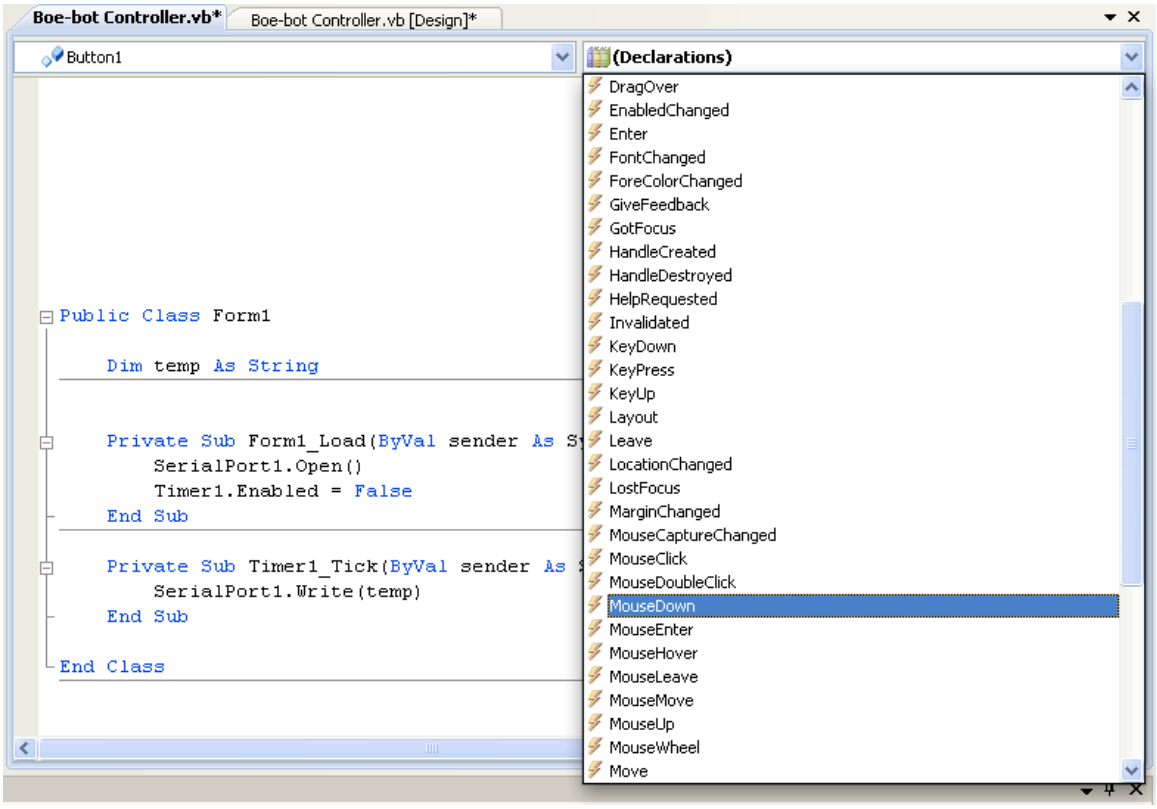
If you would like to make BB run a little smoother, I find a timer to be helpful. You start by adding a timer to the VB program. Since the refresh rate of servos is 20ms, you need to change the interval setting from 100 to 20. Now double click on the timer icon and to enter the code side.

You can start by deleting all of the other functions except the Form1_Load, and Timer1_Tick. At the top of the program declare a string variable. I used the variable name “temp.” Now, under the function Timer1_Tick, you can enter SerialPort1.Write(temp). Now, whenever the timer is enabled the COM port will send a character to the BB. This will repeat every 20ms until the timer is disabled. All we have to do now is send a character to that temp variable and enable the timer!

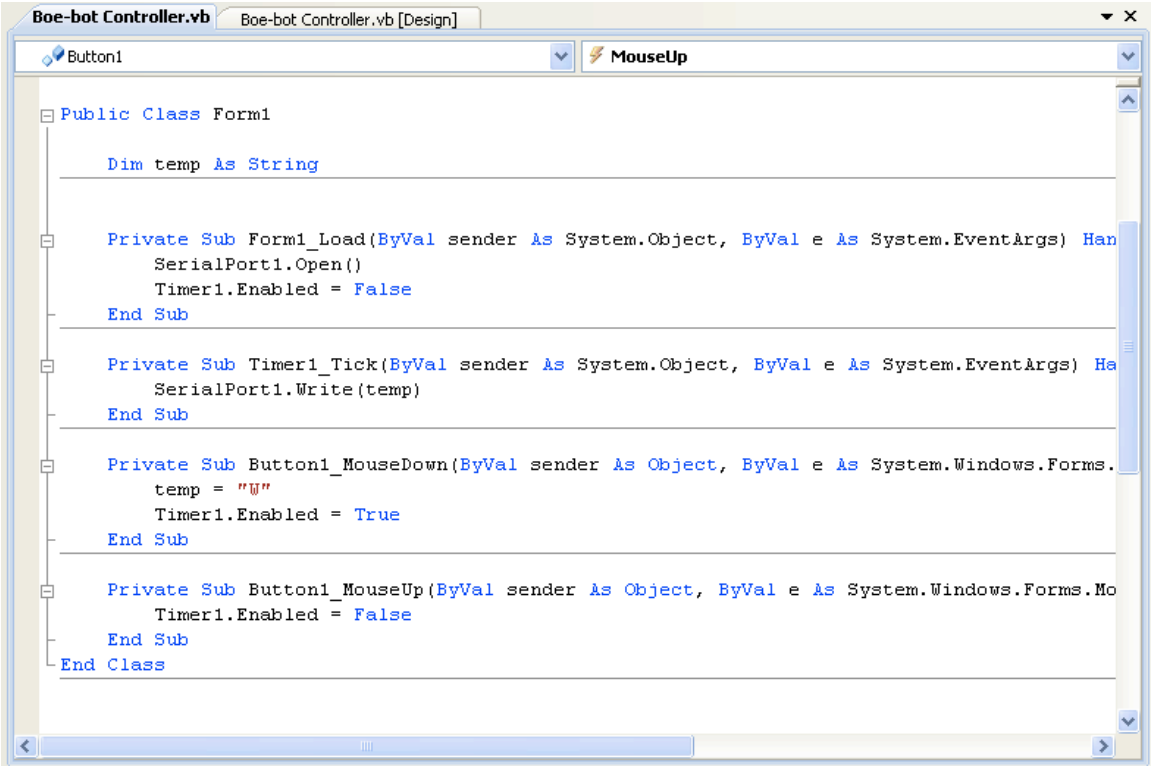
One the code side, we can select the corresponding button we want to work with from the “Class name” drop down menu. I will start with the forward button, which in my program is Button1.



Now, under the "Method Name" drop down menu, we can select MouseDown.



In the new subroutine that pops up we will set the “temp” variable equal to “W”, and enable the timer. Using the same previous method, we can make another subroutine for the mouse is lifted back up. All if needs to do is disable the timer. Your program should now resemble this:



You should now test your program to make sure it will move the BB forward and stop accordingly. Once one button is working properly it will take no time at all to add the other three.

Once the other three buttons are incorporated you are complete! Some other fun things to try are using an RF transmitter and receiver pair along with another BB board, or Sumo-bot board. Adding a few more lines of code with let you drive the BB wirelessly. You can also find Bluetooth modules sold by several companies that will allow you to connect your BB directly from your laptop. I actually just bought one yesterday, and hope to try it out soon!