Programming to Receive Joystick Input in Windows

One of the steppng stones to utilizing a computer for data acquisition and device control is the ability to receive input from its ports. After becoming familiar with programming for a PC's ports, you can start to design programs and hardware to meet the needs of your applications.

MOTIVATION AND AUDIENCE

The focus of this tutorial is to demonstrate a method for reading joystick input from a computers game port using Visual Basic in a Windows based application. This tutorial will teach you:

- ***The basics of the game port.***
- ***How to write code to receive input from a joystick.***
- ***How to utilize data received from the joystick.***

To do this, it is assumed that you already:

- ***Are familiar with programmng in Visual Basic.***

The rest of the tutorial is presented as follows:

- **Parts List and Sources**
- **The Game Port**
- **Programming**
- **Final Words**

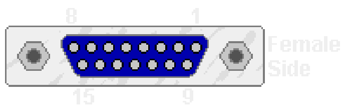
PARTS LIST AND SOURCES

All that is required for this tutorial are the following:

- ***A PC with a game port running Windows 95 or 98.***
- ***A Microsoft Visual Studio or other Visual Basic compiler.***
- ***A 2 axis, 2 button joystick that utilizes the game port.***

THE GAME PORT

The game port was originally designed to relive the erial port of devices like joysticks and MIDI devices. It eventually became the standard port for interfacing with a joystick or other game controller. The address for the game port is 0201 (Hex). The following diagram shows the pins for the game port and there respective functions:



Bit	Meaning
1	Joystick A, X Axis
2	Joystick A, Y Axis
3	Joystick B, X Axis

Pin No.	Function	Pin No.	Function
1	+ 5 Vdc	9	+ 5 Vdc
2	Joystick A, Button 1	10	Joystick B, Button 1
3	Joystick A, X Axis	11	Joystick B, X Axis

4	Joystick B, Y Axis
5	Joystick A, Button 1
6	Joystick A, Button 2
7	Joystick B, Button 1
8	Joystick B, Button 2

4	Ground	12	Ground
5	Ground	13	Joystick B, Y Axis
6	Joystick A, Y Axis	14	Joystick B, Button 2
7	Joystick A, Button 2	15	+ 5 Vdc
8	+ 5 Vdc		

The axis pins generally receive analog input from a pot on the joystick. The button pins receive a digital input. This information is primarily background, and will not greatly affect the actual programming.

PROGRAMMING

Start Visual Basic. At the Window that appears, choose "Standard EXE" as shown in Figure 1.

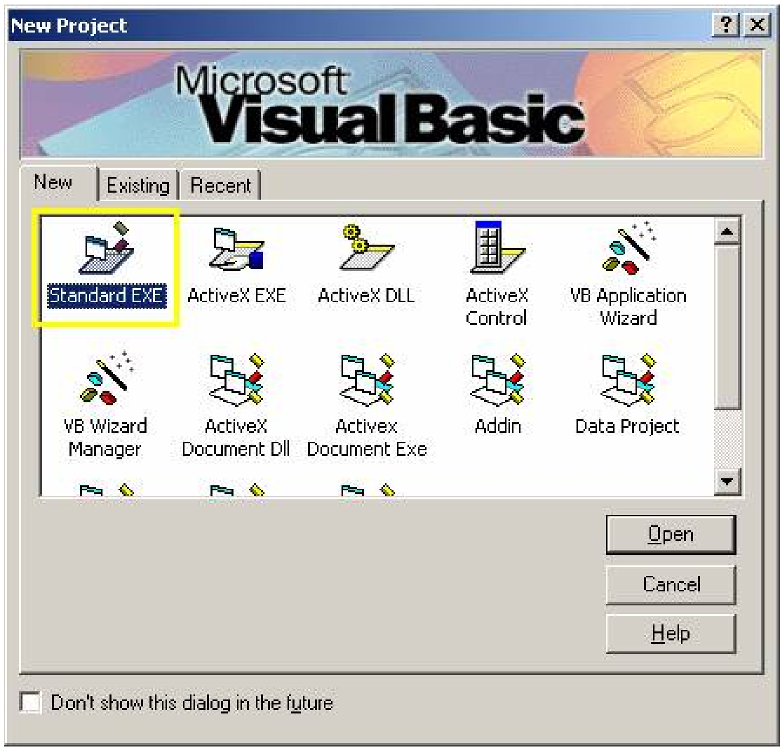
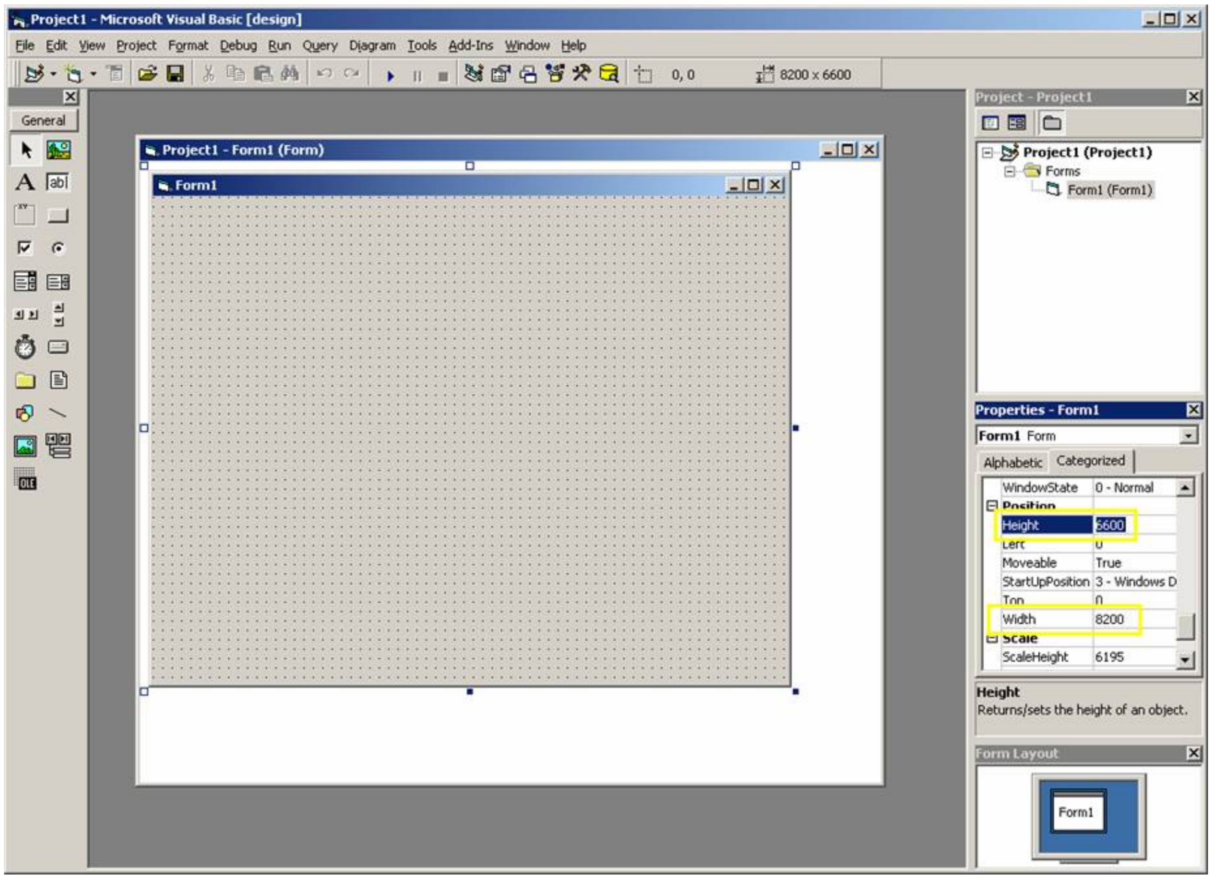


Figure 1

This should bring you to a screen with a blank form. The program we will be constructing will be a form with a dot that moves around in it based in input from the joystick. In the properties box on the right, change the width of the form to 8200 and height to 6600 as shown in Figure 2.

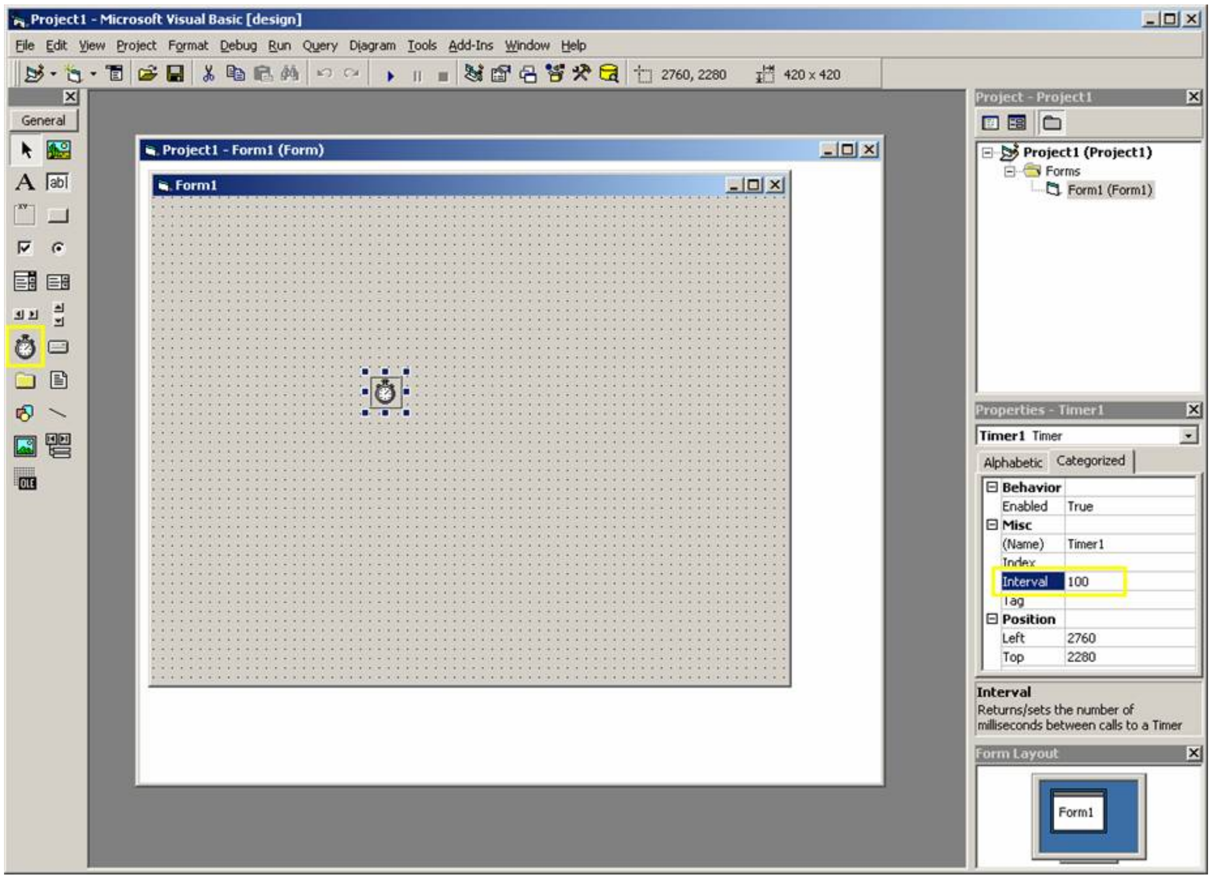


WindowState	0 - Normal
Position	
Height	6600
Left	0
Moveable	True
StartUpPosition	3 - Windows D
Top	0
Width	8200
Scale	
ScaleHeight	6195

Height
Returns/sets the height of an object.

Figure 2

From the toolbox on the left, select the timer (clock icon). Click in the form and drag to place a clock icon on the form. This adds a timer to the form to perform repeated tasks. Select the timer icon on the form. In the properties window on the right on the right, change the interval to 100.

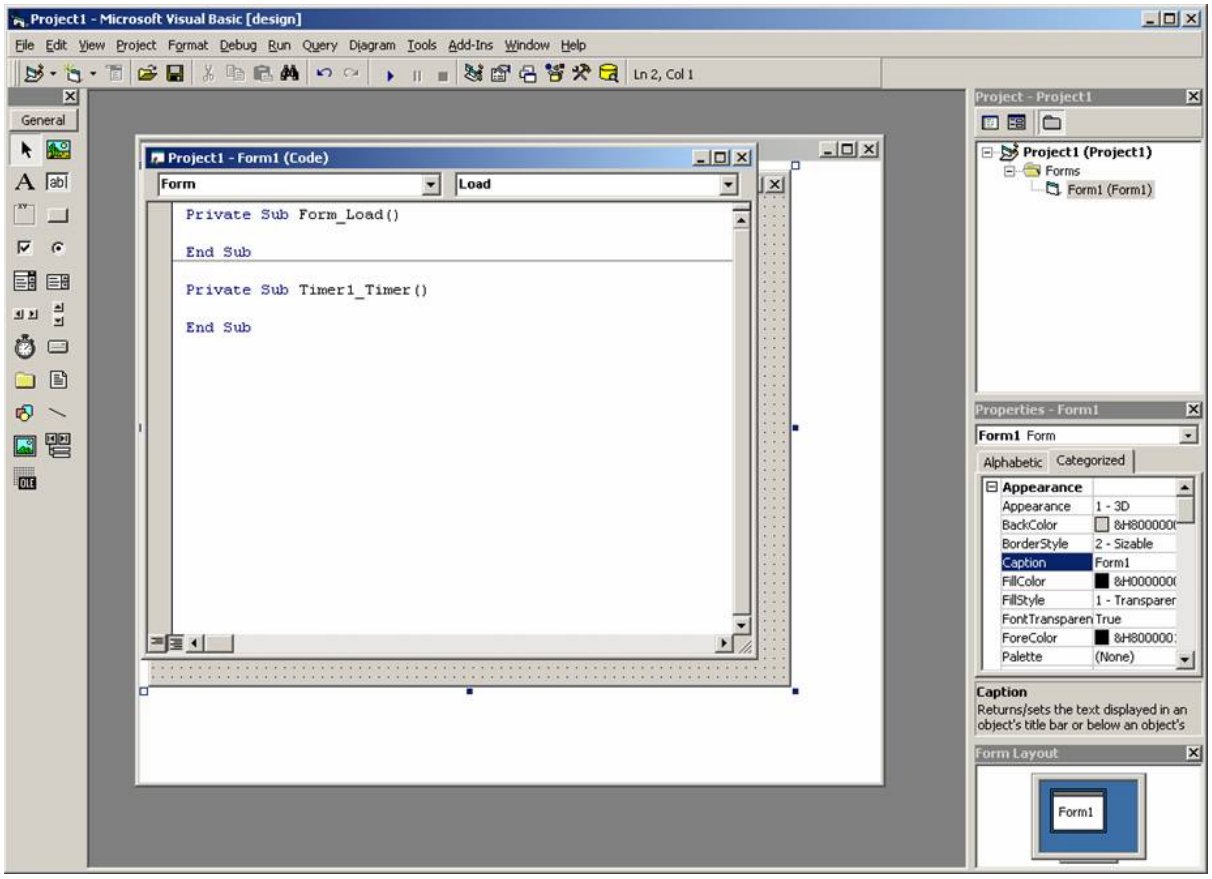


⊟ **Behavior**	
Enabled	True
⊟ **Misc**	
(Name)	Timer1
Index	
Interval	100
Tag	
⊟ **Position**	
Left	2760
Top	2280

Interval
Returns/sets the number of milliseconds between calls to a Timer

Figure 3

Now right click the form and select "View Code". This is where we will enter our code for the form. From the drop down list at the top of the window, select form. This is the area to enter code for when the form is originally loaded. You should have a screen that looks similar to Figure 4.



```
Private Sub Form_Load()

End Sub

Private Sub Timer1_Timer()

End Sub
```

Appearance	
Appearance	1 - 3D
BackColor	&H8000000(
BorderStyle	2 - Sizable
Caption	Form1
FillColor	&H0000000(
FillStyle	1 - Transparer
FontTransparen	True
ForeColor	&H800000:
Palette	(None)

Caption
Returns/sets the text displayed in an object's title bar or below an object's

Figure 4

The following is the code for the form. It will be explained in detail later.

```
Dim x, xg As Long
Dim y, yg As Long
Dim start As Long
Dim status, i As Integer
Dim PortBits(8) As Integer
Dim PortNum As Long
Dim r As Long
Dim temp As Long
Dim xmax As Long
Dim xmin As Long
Dim xratg As Double
Dim ymax As Long
Dim ymin As Long
Dim yratg As Double
Dim formh As Long
Dim formw As Long
Dim ji As JOYINFOEX
Dim caps As JOYCAPS     ' joystick capabilities
Dim rc As Long
Private Sub Form_Load()

    ji.dwSize = Len(ji)
    ji.dwFlags = JOY_RETURNALL

    ' Get the current joystick data
    rc = joyGetPosEx(JOYSTICKID1, ji)

    If (rc <> 0) Then
        MsgBox "Couldn't detect the joystick"
        End
    Else
        temp = MsgBox("Hold the joystick to the right and click 'OK'", vbOKOnly, "Calibrate X")
        rc = joyGetPosEx(JOYSTICKID1, ji)
        xmax = ji.dwXpos
        temp = MsgBox("Hold the joystick to the left and click 'OK'", vbOKOnly, "Calibrate X")
        rc = joyGetPosEx(JOYSTICKID1, ji)
        xmin = ji.dwXpos
        temp = MsgBox("Hold the joystick down and click 'OK'", vbOKOnly, "Calibrate Y")
        rc = joyGetPosEx(JOYSTICKID1, ji)
        ymax = 32600 'ji.dwYpos
        temp = MsgBox("Hold the joystick up and click 'OK'", vbOKOnly, "Calibrate Y")
        rc = joyGetPosEx(JOYSTICKID1, ji)
        ymin = 0 'ji.dwYpos
    End If

    formw = 8200
    formh = 6300

    xratg = formw / (xmax - xmin)
    yratg = formh / (ymax - ymin)

    r = 50

End Sub

Private Sub Timer1_Timer()

   ' Initialize struct
   ji.dwSize = Len(ji)
   ji.dwFlags = JOY_RETURNALL

   ' Get the current joystick data
   rc = joyGetPosEx(JOYSTICKID1, ji)

   x = ji.dwXpos
   y = ji.dwYpos

   If ji.dwButtons = 1 Then
   r = r + 50
   Else
        If ji.dwButtons = 2 Then
                If r >= 100 Then
                        r = r - 50
                 End If
         End If
   End If

   xg = Round((x - xmin) * xratg)
   yg = Round((y - ymin) * yratg)

   Form1.Circle (x, y), r

End Sub
```

VARIABLE DECLERATIONS

The first portion of code goes above the "From_Load()" sub. This declares all the variables to be used in the program. Their meaning will become evident as we progress through the code.

```
Dim x, xg As Long
Dim y, yg As Long
Dim start As Long
Dim status, i As Integer
Dim PortBits(8) As Integer
Dim PortNum As Long
Dim r As Long
Dim temp As Long
Dim xmax As Long
Dim xmin As Long
Dim xratg As Double
Dim ymax As Long
Dim ymin As Long
Dim yratg As Double
Dim formh As Long
Dim formw As Long
Dim ji As JOYINFOEX
Dim caps As JOYCAPS     ' joystick capabilities
Dim rc As Long
```

FORM LOAD

The next portion of code defines a script that will run as soon as the form is started (in other words, when the program first starts). Does a calibration to set up constants to be used later.

```
Private Sub Form_Load()

    ji.dwSize = Len(ji)
    ji.dwFlags = JOY_RETURNALL

    ' Get the current joystick data
    rc = joyGetPosEx(JOYSTICKID1, ji)

    If (rc <> 0) Then
        MsgBox "Couldn't detect the joystick"
        End
    Else
        temp = MsgBox("Hold the joystick to the right and click 'OK'", vbOKOnly, "Calibrate X")
        rc = joyGetPosEx(JOYSTICKID1, ji)
        xmax = ji.dwXpos
        temp = MsgBox("Hold the joystick to the left and click 'OK'", vbOKOnly, "Calibrate X")
        rc = joyGetPosEx(JOYSTICKID1, ji)
        xmin = ji.dwXpos
        temp = MsgBox("Hold the joystick down and click 'OK'", vbOKOnly, "Calibrate Y")
        rc = joyGetPosEx(JOYSTICKID1, ji)
        ymax = 32600 'ji.dwYpos
        temp = MsgBox("Hold the joystick up and click 'OK'", vbOKOnly, "Calibrate Y")
        rc = joyGetPosEx(JOYSTICKID1, ji)
        ymin = 0 'ji.dwYpos
    End If
```

The beginning of this code initializes the joystick variables. It tells the joystick function to return all flags (status of the joystick, max and min values, etc.). It then goes through a standard error handling process to ensure that the joystick was detected. If the joystick was detected, a series of message boxes appear. Each message box asks that the user hold the joystick to an extreme and click "ok". When the user clicks "ok", the value for the respective access is read and stored in a variable.

```
    formw = 8200
    formh = 6300

    xratg = formw / (xmax - xmin)
    yratg = formh / (ymax - ymin)

    r = 50

End Sub
```

The last part of the form load code sets up two ratios. These ratios relate the joystick value returned and to a position within the form. These will be used to scale the input from the joystick. Finally, the radius of the dot is initialized to 50.

UTILIZING INPUT FROM THE JOYSTICK

The remainder of the code is a simple application that places a circle on the form who's position is directed by the joystick. The circle changes color and size based on which buttons are pressed.

```
Private Sub Timer1_Timer()

   ' Initialize struct
   ji.dwSize = Len(ji)
   ji.dwFlags = JOY_RETURNALL

   ' Get the current joystick data
   rc = joyGetPosEx(JOYSTICKID1, ji)

   x = ji.dwXpos
   y = ji.dwYpos
```

It should first be noted that this code is contained within the timer sub. This will cause the code to be run with every cycle of the timer (and thereby read a new joystick position and redraw the dot). The joystick is initialized and the postion from the joystick is read and stored in the variables x and y.

```
   If ji.dwButtons = 1 Then
   r = r + 50
   Else
        If ji.dwButtons = 2 Then
                If r >= 100 Then
                        r = r - 50
                 End If
         End If
   End If
```

Based on which button is hit, the radius of the circle is either increased or decreased, but the radius is never allowed to drop below 50.

```
   xg = Round((x - xmin) * xratg)
   yg = Round((y - ymin) * yratg)

   Form1.Circle (x, y), r

End Sub
```

The input from the joystick is scaled and rounded to the nearest integer. This will give us a position we can use on the form. A circle is then drawn at this position, with the appropriate radius.

FINAL WORDS

After completing this tutorial you should be familiar with the the gameport, be able to program in Visual Basic to receive input from a joystick and apply that input to an application.

If you have questions about this tutorial you can email me at **Keithicus@drexel.edu**.