

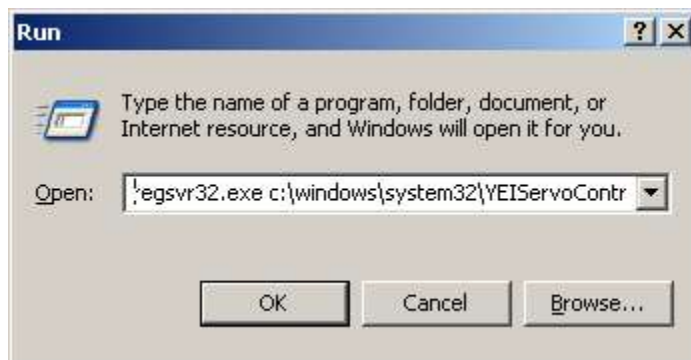
ServoCenter3.1 ActiveX Control Examples – Visual Basic 6.0

1. Overview

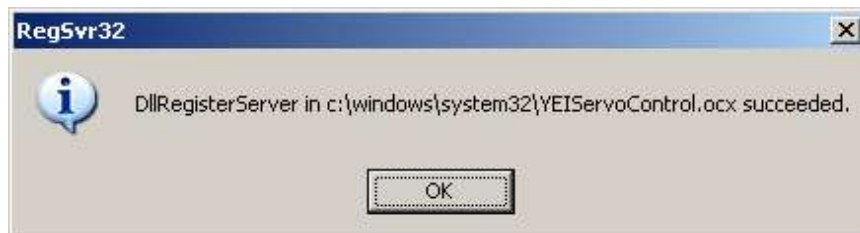
To avoid the raw serial method of programming the ServoCenter 3.1 board, the YEIServoCenter ActiveX Control can be used (in a Visual Basic 6.0 project). This Control is designed to allow the programmer to concentrate on controlling the servos without having to worry about the ServoCenter 3.1 communications protocol. Below is an explanation of how to program the ServoCenter 3.1 using the YEIServoCenter ActiveX Control.

2. Installing the ActiveX Control

1. Copy YEIServoControl.ocx from the CD to the ActiveX control directory. In Windows 95/98/ME, ActiveX controls are stored in C:\WINDOWS\SYSTEM\. In Windows 2000/XP, they are stored in C:\WINDOWS\SYSTEM32\.
2. Register YEIServoControl.ocx. To do this, click the Start button, and then select Run. Into the Run dialog box, type “Regsvr32.exe” followed by a space and then the path and filename of the ServoControl.



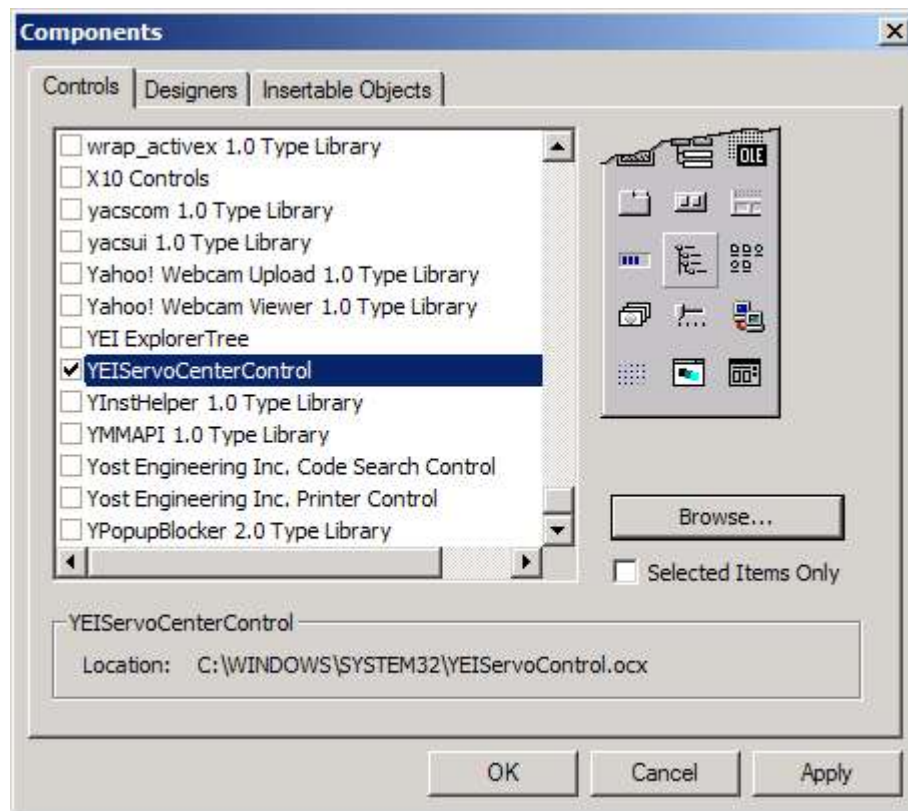
If the control is successfully registered, you will see a dialog box like this...



The Control is now installed and ready for use!

3. Adding the ServoCenter Control to a Visual Basic 6.0 project

1. Open a new project in the Visual Basic 6.0 editor.
2. Add the YEIServoControl to your project. To do this, click the “Project” menu at the top of the editor and then select “Components”. This should bring up a menu that looks like this:



If the YEIServoControl was copied to the correct directory, it should appear in the list in the dialog box. If not, it can still be added by clicking the “Browse” button and browsing to the directory where it is stored.

Once the YEIServoControl has been selected, click the “OK” button to add it to your project.

3. With the YEIServoControl now added to your project, select it from the toolbox and draw it on the form.
4. In the “Properties” pane of the VB editor, specify the communications port (**ComPort** property) by which the control will communicate with the ServoCenter 3.1 board. A **ComPort** property of 1 will signify Com1, a 2 will signify Com2, and so on up to Com4.

There are two more properties that deal with communication to the board. One is the **BoardID** and the other is **BaudRate**. The **BoardID** corresponds to the BoardID switches found on the controller board, and the **BaudRate** is the speed at which data will be sent to and from the controller board. The **BoardID** can be set from 0 to 15 and **BaudRate** can be set to 9600,14400,19200, or 38400. Make sure that the board has the correct jumper settings on JP2 and that your board's ID switches are in the correct positions to ensure successful communication.

5. There are several other entries in the “Properties” pane. One is the **ServoConfigFile** property. This property can be used to specify a file which will store servo min, max, and start information for up to 16 ServoCenter 3.1 boards. A related property is the **AutoCreateConfigFile** property. Setting this property to True will cause a file to be created to store settings, even if a name is not specified in the **ServoConfigFile** property.

The boolean property **SynchronizeBoard** determines where the values that are displayed on the ServoConfig screen are pulled from. If **SynchronizeBoard** is True, the values displayed on the ServoConfig screen are polled from the board with the current **BoardID**. If **SynchronizeBoard** is False, the control will not poll the current board, but will show the data for the current **BoardID** that is in the file that was specified by **ServoConfigFile**.

This completes the process of preparing the project for ServoCenter 3.1 use. You may now begin writing code to control the servos attached to your ServoCenter 3.1. To ensure the proper functionality of the YEIServoControl, please note that the **InitBoard** method must be invoked before attempting to access the ServoCenter 3.1 board. Once the InitBoard method has been executed, the servos attached to the ServoCenter 3.1 board can be manipulated, and information about these servos can be displayed, with the methods described in Section 4.3.4 of the ServoCenter 3.1 manual.

4. ServoCenter 3.1 OCX Methods

All of YEIServoControl's movement and set methods take integer arguments, and all of the get methods return integers, unless otherwise specified. All methods below which are followed by an asterisk (*) have a counterpart that will perform the same function, but to all servos [0-15]. For example: the QuickMove method will move a single servo to a specified position and the QuickMoveAll method will move all 16 servos to the specified position. The methods below which are followed by 2 asterisks (**) also have an All counterpart, which returns an array of integers [0-15]. Only the single servo version of each method will be fully described.

Value ranges for arguments:

ServoNum – 0 to 15

ServoPosition, ChangeInServoPosition – 0 to 200

ScaledServoPosition, PercentOfSpeed – 0 to 100

ServoMaxSpeed, PulseWidth10usUnits – 1 to 239

QuickMove Method*

The QuickMove method moves the specified servo to the specified position. This method is of the form:

```
YEIServoCtrl1.QuickMove(ServoNum, ServoPosition)
```

QuickScaledMove Method*

The QuickScaledMove method moves the specified servo to the specified scaled position. The scaled position is really a percentage of the distance between the specified servo's min and max. This method is of the form:

```
YEIServoCtrl1.QuickScaledMove(ServoNum, ScaledServoPosition)
```

ServoEnable Method*

The ServoEnable method will enable the specified servo. This method is of the form:

```
YEIServoCtrl1.ServoEnable(ServoNum)
```

ServoDisable Method*

The ServoDisable method will disable the specified servo. This method is of the form:

```
YEIServoCtrl1.ServoDisable(ServoNum)
```

SetMin Method*

The SetMin method sets the min position for the specified servo to a specified position. This method is of the form:

```
YEIServoCtrl1.SetMin(ServoNum, ServoPosition)
```

SetMax Method*

The SetMax method sets the max position for the specified servo to the specified position. This method is of the form:

```
YEIServoCtrl1.SetMax(ServoNum, ServoPosition)
```

SetStart Method*

The SetStart method sets the start position for the specified servo to the specified position. This method is of the form:

```
YEIServoCtrl1.SetStart(ServoNum, ServoPosition)
```

SetMaxSpeed Method*

The SetMaxSpeed method sets the maximum speed rating for the specified servo. This rating is normally in μs units. Multiplying ServoMaxSpeed by $10\mu\text{s}$ yields the maximum speed. This method is of the form:

```
YEIServoCtrl1.SetMaxSpeed(ServoNum, ServoMaxSpeed)
```

SetMaxCurrent Method*

The SetMaxCurrent method takes the current raw position of the specified servo and sets its max position equal to it. This method is of the form:

```
YEIServoCtrl1.SetMaxCurrent(ServoNum, ServoPosition)
```

SetMinCurrent Method*

The SetMinCurrent method takes the current raw position of the specified servo and sets its min position equal to it. This method is of the form:

```
YEIServoCtrl1.SetMinCurrent(ServoNum)
```

SetStartCurrent Method*

The SetStartCurrent method takes the current raw position of the specified servo and sets its start position equal to it. This method is of the following form:

```
YEIServoCtrl1.SetStartCurrent(ServoNum)
```

GetCurrentPos Method**

The GetCurrentPos method gets the current position of the specified servo and returns it. This method is of the form:

```
YEIServoCtrl1.GetCurrentPos(ServoNum)
```

GetMin Method**

The GetMin method gets the min position of the specified servo and returns it. This method is of the form:

```
YEIServoCtrl1.GetMin(ServoNum, ServoPosition)
```

GetMax Method**

The GetMax method gets the max position of the specified servo and returns it. This method is of the form:

```
YEIServoCtrl1.GetMax(ServoNum, ServoPosition)
```

GetStart Method**

The GetStart method gets the start position of the specified servo and returns it. This method is of the form:

```
YEIServoCtrl1.GetStart(ServoNum, ServoPosition)
```

GetMaxSpeed Method**

The GetMaxSpeed method gets the max speed of the specified servo and returns it. This method is of the form:

```
YEIServoCtrl1.GetMaxSpeed(ServoNum, ServoPosition)
```

MoveRaw Method*

The MoveRaw method moves the specified servo to the specified position at the specified percent of the max speed for that servo. This method is of the form:

```
YEIServoCtrl1.MoveRaw(ServoNum, ServoPosition, PercentOfSpeed)
```

MoveRawCW Method*

The MoveRawCW method moves the specified servo in the Clockwise direction in the amount specified by ChangeInServoPosition at the specified percent of the max speed for that servo. This method is of the form:

```
YEIServoCtrl1.MoveRawCW(ServoNum, ChangeInServoPosition,  
PercentOfSpeed)
```

MoveRawCCW Method*

The MoveRawCCW method moves the specified servo in the Counter-Clockwise direction in the amount specified by ChangeInServoPosition at the specified percent of the max speed for that servo. This method is of the form:

```
YEIServoCtrl1.MoveRawCCW(ServoNum, ChangeInServoPosition,  
PercentOfSpeed)
```

MoveScaled Method*

The MoveScaled method moves the specified servo to the specified scaled position at the specified PercentOfSpeed. The scaled position is really a percentage of the distance between the specified servo's min and max. This method is of the form:

```
YEIServoCtrl1.MoveScaled(ServoNum, ScaledServoPosition,  
PercentOfSpeed)
```

MoveScaledCW Method*

The MoveScaledCW method moves the specified servo to the specified scaled position in a Clockwise direction at the specified PercentOfSpeed. The scaled position is really a percentage of the distance between the specified servo's min and max. This method is of the form:

```
YEIServoCtrl1.MoveScaledCW(ServoNum, ScaledServoPosition,  
PercentOfSpeed)
```

MoveScaledCCW Method*

The MoveScaledCCW method moves the specified servo to the specified scaled position at the specified PercentOfSpeed. The scaled position is really a percentage of the distance between the specified servo's min and max. This method is of the form:

```
YEIServoCtrl1.MoveScaledCCW(ServoNum, ScaledServoPosition,  
PercentOfSpeed)
```

SetPulseWidthMin Method

Servos require a pulse to move the servo to a specified location. The pulse width min is set to (PulseWidth10usUnits * 10 μ s), so passing 1 sets the PulseWidthMin to 10 μ s. This method is of the form:

```
YEIServoCtrl11.SetPulseWidthMin(PulseWidth10usUnits)
```

SetPulseWidthMax Method

Servos require a pulse to move the servo to a specified location. The pulse width max is set to (PulseWidth10usUnits * 10 μ s), so passing 100 sets the PulseWidthMax to 1000 μ s. This method is of the form:

```
YEIServoCtrl11.SetPulseWidthMax(PulseWidth10usUnits)
```

GetSettings Method

The GetSettings method returns a string containing information about the status of all of the servos. This information can be corrupted if you're sending data at the same time that you are trying to get data. This method is of the form:

```
YEIServoCtrl11.GetSettings()
```

CommitSettings Method

The CommitSettings method takes the current min values, max values, and other settings and stores them in the EEPROM on the ServoCenter 3.1 board. These settings will then be loaded the next time the board is restarted. This method is of the form:

```
YEIServoCtrl11.CommitSettings()
```

RestoreFactory Method

The RestoreFactory method restores the settings to the Factory Settings. This method is of the form:

```
YEIServoCtrl11.RestoreFactory()
```

ResetAsStartup Method

The ResetAsStartup method does a software restart of the system. It takes the stored settings and sets the current settings to them. This method is of the form:

```
YEIServoCtrl11.ResetAsStartup()
```

GetVersion Method

The GetVersion method returns a string that contains the copyright information and version of the board. This method is of the form:

```
YEIServoCtrl11.GetVersion()
```

ShowServoConfig Method

The ShowServoConfig method displays the configuration dialog box for the board. This method is of the form:

```
YEIServoCtrl11.ShowServoConfig()
```

InitBoard Method

The InitBoard method initializes the COM port and loads settings for the servos. This method is of the form:

```
YEIServoCtrl1.InitBoard()
```

5. Sample Code

The following code examples illustrate the use of the YEIServoControl in Visual Basic 6.0.

Here is an example of how to initialize your ServoCenter 3.1 Board:

```
Private Sub Form_Load()  
    YEIServoCtrl1.InitBoard      'initialize the control  
    YEIServoCtrl1.BoardID = 0    ' set board id to 0  
End Sub
```

Here is an example of how to move a servo by clicking on a button:

```
Private Sub cmdMoveServo_Click()  
    Call YEIServoCtrl1.QuickMove(0, 100)  
    'Move servo 0 to raw position 100  
End Sub
```

Here is an example of how to show the servo configuration dialog box:

```
Private Sub cmdShowConfig_Click()  
    YEIServoCtrl1.ShowServoConfig      'show config dialog  
End Sub
```

6. Additional Information

This code is available in the ServoCenterOCXDemo.vbp file in the Examples directory of the ServoCenter3.1 CD or online at www.YostEngineering.com/ServoCenter.