

VC-C50i

Sample Application

Visual Basic

Note: This program was developed using Visual Basic Ver.6.0 (SP5). If the setup for the sample program is performed in another environment, problems may occur in the operation of other programs.

Contents

1. Setup of Sample Program	2
1.1 When the Visual Basic 6.0 (SP5) environment is already installed	2
1.2 When the Visual Basic 6.0 (SP5) environment is not installed	3
2. Adding Functions (Zoom)	4
2.1 Procedure for adding buttons	4
3. Sample Program Functions	7
3.1 RxWait Waiting for receipt of reply command	7
3.2 GetTxCommand Loading of transmission information	7
3.3 SetRxCommand Loading of receiving data	7
4. Sample Program Receive Events	8
4.1 AckRxEvent ACK command receive event	8
4.2 NotifyRxEvent Notification command receive event	8
5. ACK and Notify Command Receive Monitoring	9
6. ActiveX Controller References	10
6.1 Creating projects	10
6.2 Installing ActiveX controllers	12
6.3 Pasting the Vcc5AXCtrlE object	13

1. Setup of Sample Program

1.1 When the Visual Basic 6.0 (SP5) environment is already installed

- (1) Copy the CD-ROM sample software and ActiveX control files to the hard disk.
- (2) Copy the Vcc5AXCtrlE.OCX file and Vcc5E.DLL file to the System directory (System32) in Windows.

In Windows 2000:

¥Winnt¥System32

In Windows XP:

¥Windows¥System32

- (3) Add the Acc5AXCtrlE.OCX file entry to the registry.

To add to the registry, open the command prompt in the Accessory menu, and then enter the following command in the command line.

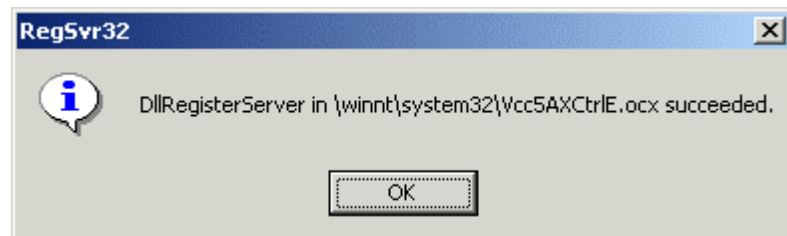
In Windows 2000:

Regsvr32 ¥Winnt¥System32¥Vcc5AXCtrlE.OCX

In Windows XP:

Regsvr32 ¥Windows¥System32¥Vcc5AXCtrlE.OCX

The message shown below is displayed when the registration is successful.



*Registry entries can be deleted by using the u parameter of Regsvr32.

In Windows 2000:

Regsvr32 /u ¥Winnt¥System32¥Vcc5AXCtrlE.OCX

In Windows XP:

Regsvr32 /u ¥Windows¥System32¥Vcc5AXCtrlE.OCX

- (4) Copy the sample software (all files in the VBSample_E folder) to the appropriate location.
The sample software is then available for usage.

1.2 When the Visual Basic 6.0 (SP5) environment is not installed

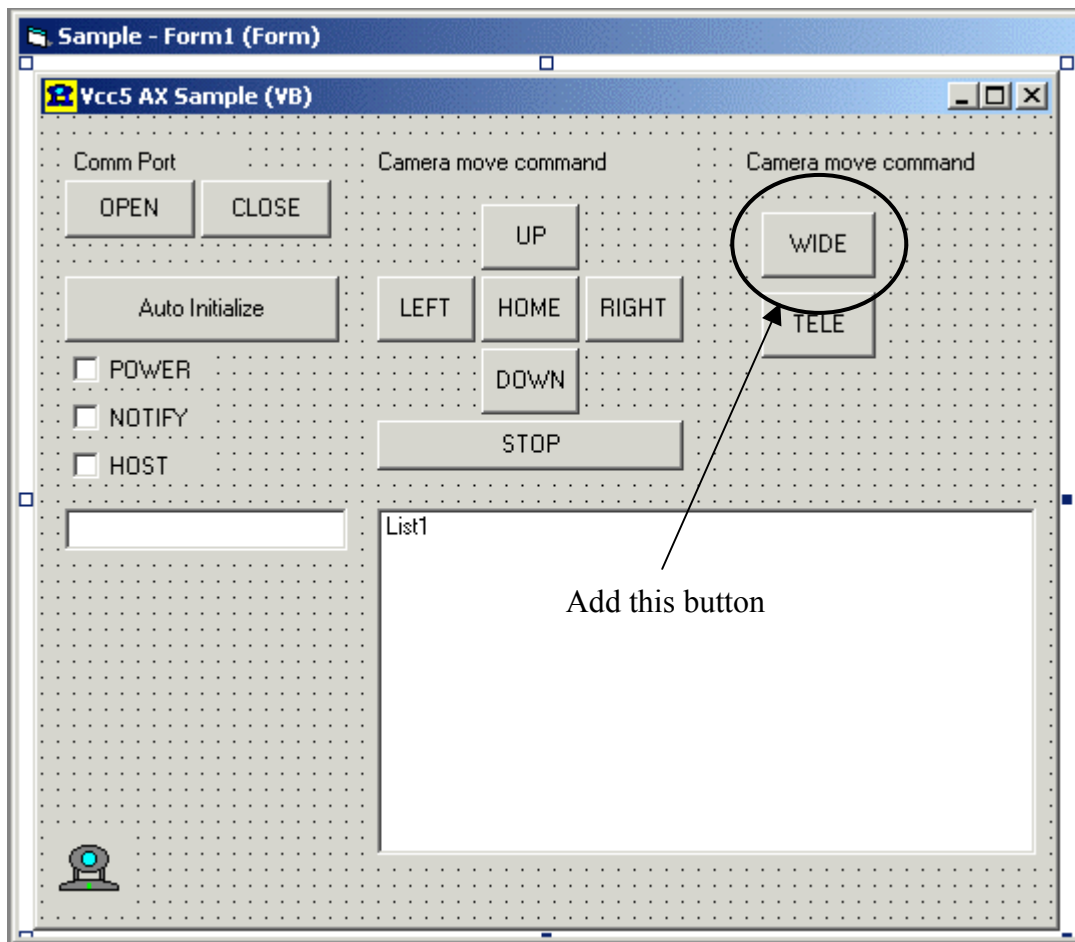
- (1) Use the setup disc to install the sample software and register the required OCX and DLL files.
- (2) Copy the Vcc5AXCtrlE.OCX and Vcc5E.DLL files to the setup directory.
- (3) The sample program can now be run.

2. Adding Functions (Zoom)

2.1 Procedure for adding buttons

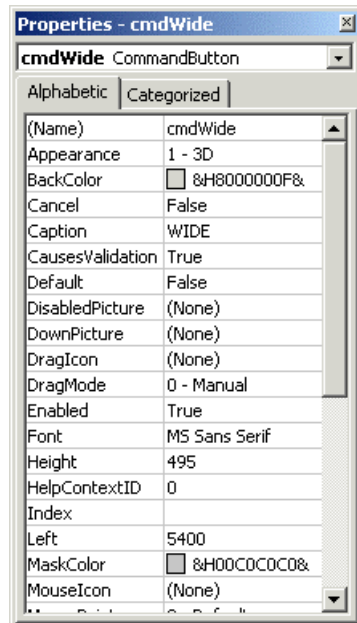
This section shows the adding of the WIDE button as an example. The user decides how to change the name and other properties of the button to be added. These user-selected items are indicated by "(User)" in the descriptions below.

- (1) Click the Command Button icon in the toolbox, and then place the Command button on the form.

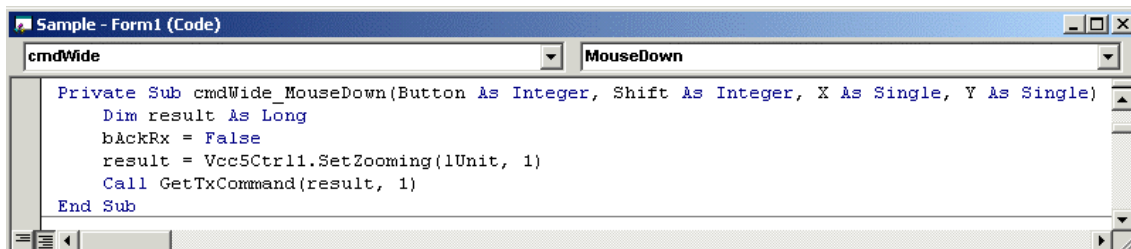


- (2) To change the button indicator, select the Caption properties under the Properties window, and then change "Command1" to "WIDE" (User).

- (3) To change the button name, select the (object name) in the Properties window, and then change "Command1" to "cmdWide" (User).



- (4) Select Code from the View menu to display the Code window.
- (5) Create the Wide start program (operation which starts when the WIDE button is pressed). Select "cmdWide" from the objects in the combo box at the top of the Code window. The cmdWide_Click event procedure is automatically created, and so be sure to delete it after completing step (6).
- (6) Select "MouseDown" from the procedures in the combo box.



- (7) The following code is written to the cmdWide_MouseDown event.

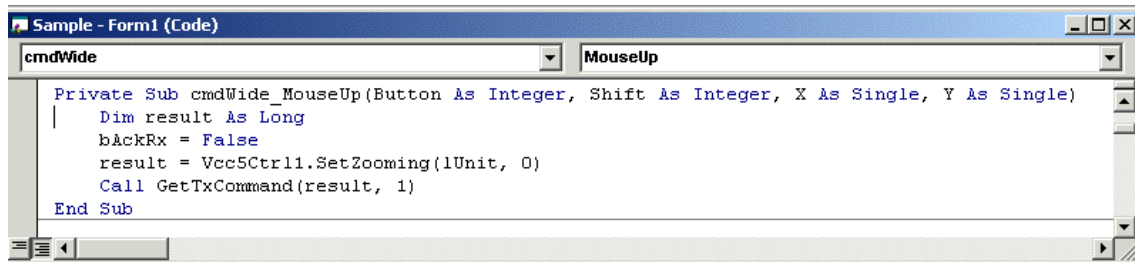
```
Private Sub cmdWide_MouseDown()
    Dim result As Long
    bAckRx = False           'ACK receive flag clear'
    result = Vcc5Ctrl1.SetZooming(1Unit, 1) 'Wide zoom command transmission'
    Call GetTxCommand(result, 1)           'Load transmission information'
End Sub
*For the Wide zoom commands, refer to the ActiveX control function manual.
```

- (8) Create the Wide stop program (operation which ends when the WIDE button is released). Select "cmdWide" from the objects in the combo box at the top of the Code window.
- (9) Select "MouseUp" from the procedures in the combo box.

(10) Copy the code for the MouseDown event and paste it to cmdWide_MouseUp.

(11) Change the Wide zoom command parameter to Stop.

result = Vcc5Ctrl1.SetZooming(1Unit, 0)'Wide zoom stop command transmission



The screenshot shows a Visual Basic code editor window titled "Sample - Form1 (Code)". The "cmdWide" control is selected, and the "MouseUp" event is chosen from the dropdown menu. The code for the MouseUp event is as follows:

```
Private Sub cmdWide_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    Dim result As Long
    bAckRx = False
    result = Vcc5Ctrl1.SetZooming(1Unit, 0)
    Call GetTxCommand(result, 1)
End Sub
```

(12) This completes the procedure for adding a button. Execute the operation to check it.

Add the function for the TELE button using the same procedure as the WIDE button.

```
Private Sub cmdTele_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    Dim result As Long
    bAckRx = False
    result = Vcc5Ctrl1.SetZooming(1Unit, 0)
    Call GetTxCommand(result, 1)
End Sub

Private Sub cmdTele_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
    Dim result As Long
    bAckRx = False
    result = Vcc5Ctrl1.SetZooming(1Unit, 2)
    Call GetTxCommand(result, 1)
End Sub

Private Sub cmdWide_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    Dim result As Long
    bAckRx = False
    result = Vcc5Ctrl1.SetZooming(1Unit, 0)
    Call GetTxCommand(result, 1)
End Sub

Private Sub cmdWide_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
    Dim result As Long
    bAckRx = False
    result = Vcc5Ctrl1.SetZooming(1Unit, 1)
    Call GetTxCommand(result, 1)
End Sub
```

3. Sample Program Functions

3.1 RxWait Waiting for receipt of reply command

This monitors the reply commands sent from the camera.

Function RxWait(ByRef bRx As Boolean) As Long

Parameter

bRx Receive command type

This is assigned by referring to the global variables bAckRx and bNotifyRx.

The bAckRx and bNotifyRx variables are set to True when the ACK command is received and when the Notification command is received, respectively.

The bAckRx and bNotifyRx variables must be set to False when a command is transmitted.

Return Value

This value is set to "1" when reception is successful.

This value is set to "0" when a timeout has occurred.

3.2 GetTxCommand Loading of transmission information

This obtains the data that was actually transmitted.

Sub GetTxCommand(ByVal lFlag As Long, ByVal lAckSw As Long)

Parameter

lFlag: Command transmission error data

When this flag is "0", the "Command Tx Error" message is displayed and the function is ended.

When this flag is a value other than "0", the transmission data is loaded.

lAckSw: After this command is transmitted, monitoring of ACK receive waiting is set.

When this value is "0", no ACK receive waiting is performed.

When this is a value other than "0", the ACK receive waiting is performed.

Return Value

None

3.3 SetRxCommand Loading of receiving data

This loads the received data generated for each event.

Sub SetRxCommand(ByVal strMsg As String, ByVal lBytes As Long, ByRef bBuff () As Byte)

Parameter

strMsg: Receive command message

This information is added to the top of the received data and a list box is displayed.

lBytes: Size of received data

bBuff: Buffer of received data array

This designates the data area loaded from the ActiveX control side.

Return Value

None

4. Sample Program Receive Events

4.1 AckRxEvent ACK command receive event

An event is generated when an ACK command is received.

While the event is being generated, the receive command can be obtained by the GetPicData function.

```
Sub Vcc5Ctrl1_AckRxEvent(ByVal wParam As Long, ByVal lBytes As Long)
```

Parameter

wParam: Receive command type

"1" is set when the ACK/NACK command is received.

lBytes: Size of received data

Return Value

None

4.2 NotifyRxEvent Notification command receive event

An event is generated when a Notification command is received.

While the event is being generated, the receive command can be obtained by the GetPicData function.

```
Sub Vcc5Ctrl1_NotifiRxEvent(ByVal wParam As Long, ByVal lBytes As Long)
```

Parameter

wParam: The next receive command type is set.

1: When a Notification command is received (header: FAh)

2: When a Remote Control Through (Remote Control ON) command is received (header: FDh)

3: When a Remote Control Through (Remote Control OFF) command is received
(header: FCh)

4: Global command result notification received (header: F8h)

5: Event Notification command received (header: FBh)

lBytes: Size of received data

Return Value

None

5. ACK and Notify Command Receive Monitoring

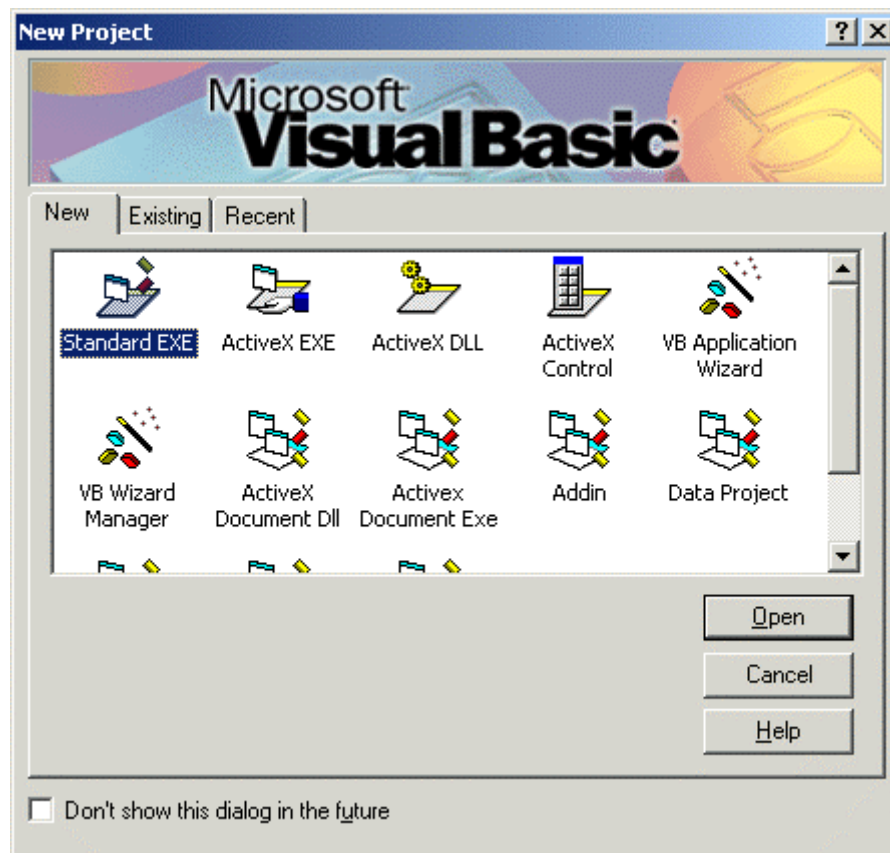
Pressing the Auto Initialize command button will automatically execute the next command transmission while confirming the reply.

1. Host mode setting ACK receive wait
2. Notification mode setting ACK receive wait
3. Camera power on Notification command receive wait

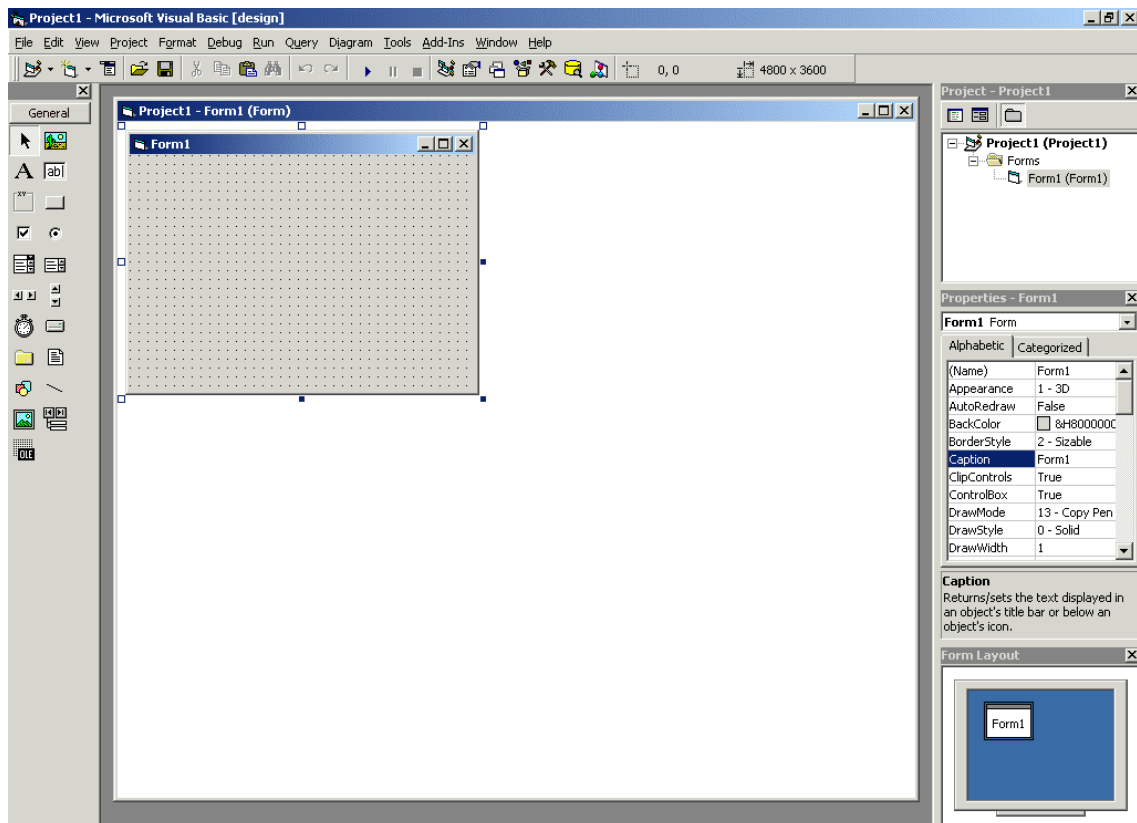
6. ActiveX Controller References

6.1 Creating projects

Start Visual Basic, and select Standard.EXE in the New Project dialogue box under the New tab.

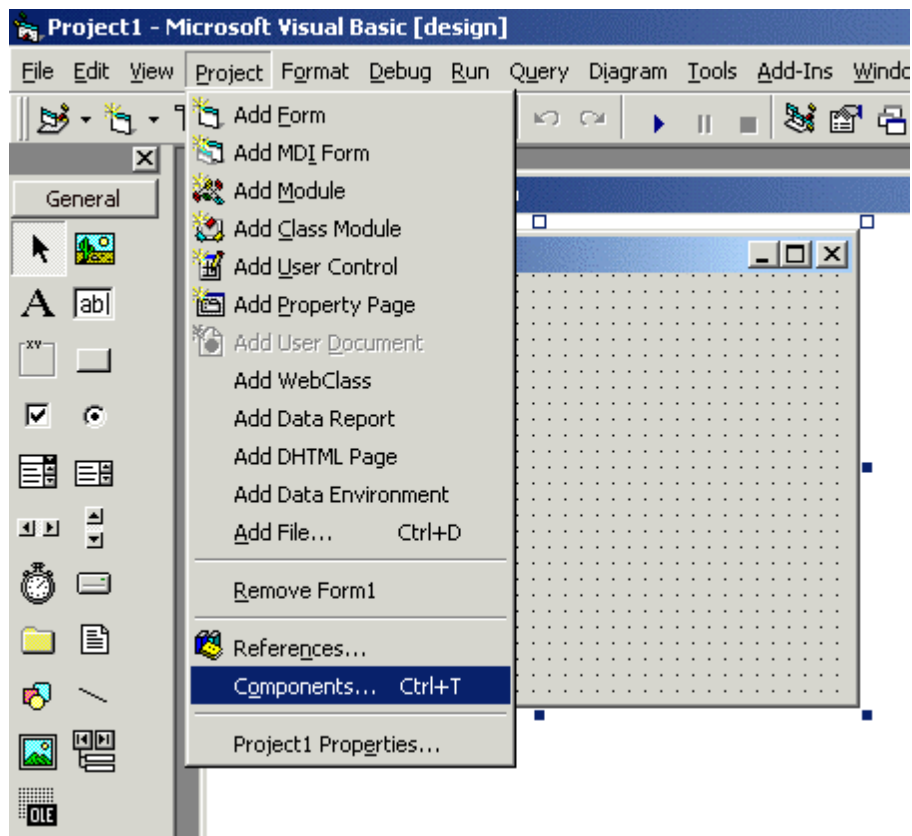


The form for Form1 is displayed in the Visual Basic window.

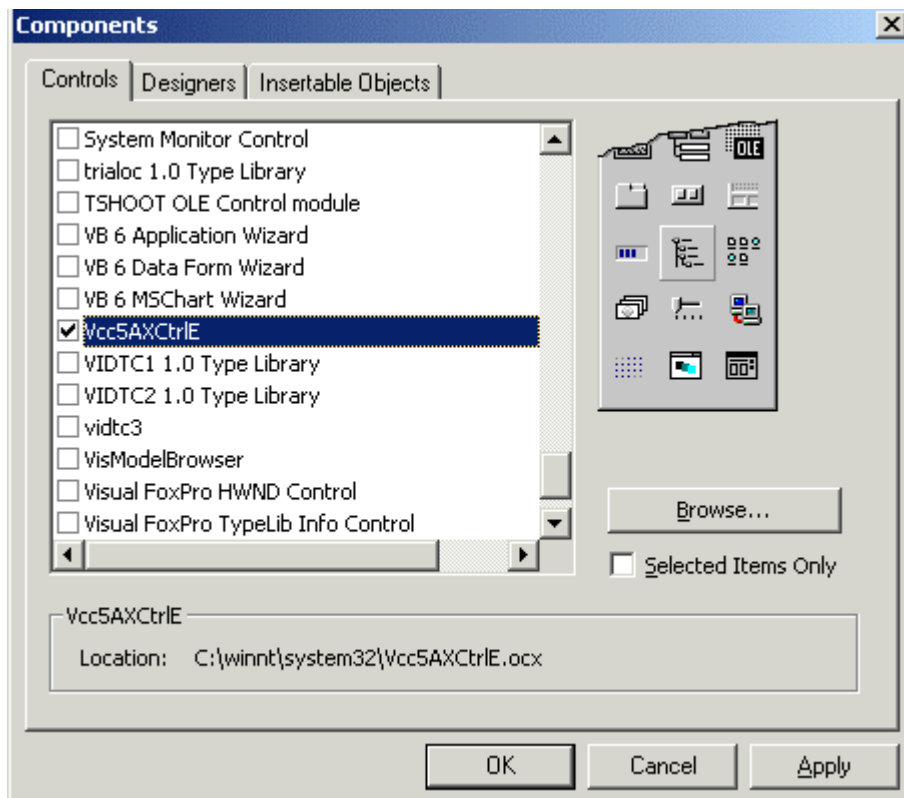


6.2 Installing ActiveX controllers

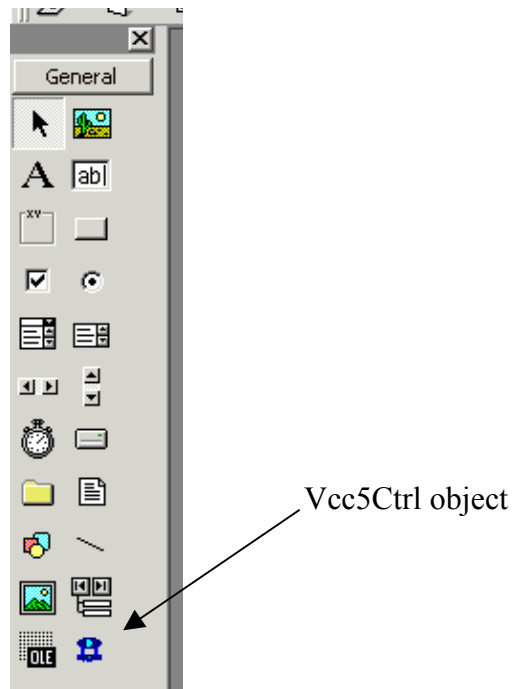
Select the Components under the Project menu in the menu bar.



Select Vcc5AXCtrlE from the component selection dialogue box, and then click OK.

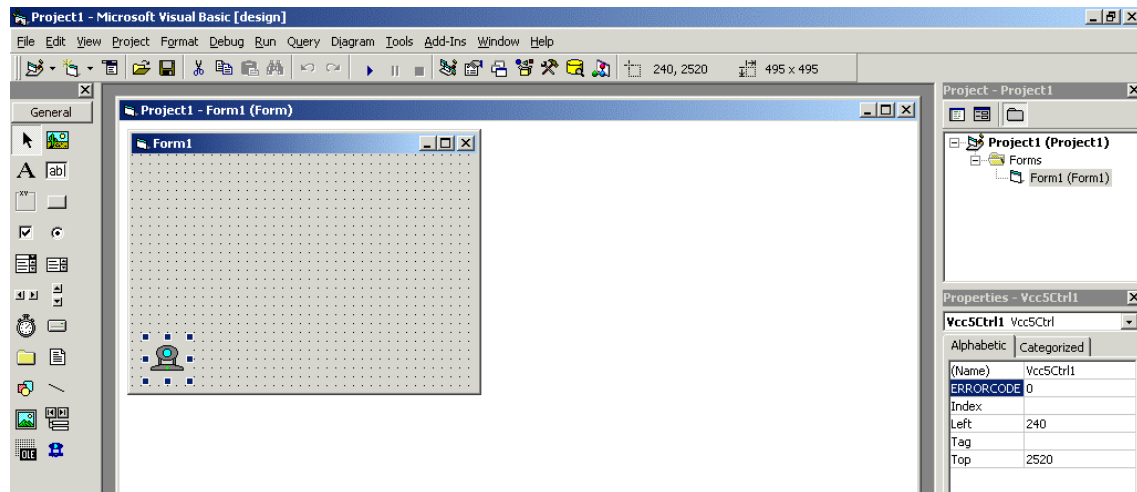


The Vcc5AXCtrlE object is displayed in the toolbox.



6.3 Pasting the Vcc5AXCtrlE object

Use the mouse to select the Vcc5Ctrl in the toolbox, and then paste to Form1.



These operations enable you to refer to the method, event, and properties for Vcc5AXCtrlE in Project1.