Universal Device Manual

Standardized Computer Control Documentation

About the Universal Device Manual

The NCD Universal Device Manual (UDM) was created as a common reference guide for NCD products released after January, 2007. This manual covers many products, highlighting the universal standards and differences in our products. The first portion of this manual covers the similarities and standards that all products share. The final portion of the manual highlights the specific details of each product. From specific commands to custom wiring connections, the individuality of each product is fully detailed in this guide.

Manual Index

Basic Wiring and Communication to NCD Devices

Supported Devices

This section of the manual describes individual products, including connection diagrams and a detailed command summary for each device. All products released in 2007 and later will be referenced in this manual. Products released in previous years will continue to have individual product manuals for each device. Product dimensional drawings are ONLY available on our web site at www.iorelay.com.

Mirror Modules

Warranty
NCD Warrants its products against defects in materials and workmanship for a period of 5 years. If you discover a defect, NCD will, at its option, repair, replace, or refund the purchase price. Simply return the product with a description of the problem and a copy of your invoice (if you do not have your invoice, please include your name and telephone number). We will return your product, or its replacement via UPS Ground service anywhere within the continental United States. Customers outside the continental US 48 States will be responsible for all shipping charges.

This warranty does not apply if the product has been modified or damaged by accident, abuse, or misuse.

30-Day Money-Back Guarantee
If, within 30 days of having received your product, you find that it does not suit your needs, you may return it for a refund. NCD will refund the purchase price of the product, excluding shipping/handling costs. This guarantee does not apply if the product has been altered or damaged.

Copyrights and Trademarks

Disclaimer of Liability
NCD is not responsible for special, incidental, or consequential damages resulting from any breach of warranty, or under any legal theory, including lost profits, downtime, goodwill, damage to or replacement of equipment or property, and any costs or recovering, reprogramming, or reproducing any data stored in or used with NCD products.

Technical Assistance
Technical questions should be directed to National Control Devices Technical Support Staff. For current contact information, please visit www.iorelay.com and click “Contact Information” on the left side. Technical questions submitted via e-mail are answered frequently throughout the business day. Technical support is also available by calling (417) 646-5644.

NCD Contact Information

Mailing Address:
National Control Devices
P.O. Box 455
Osceola, MO 64776

Telephone:
(417) 646-5644

FAX:
(866) 562-0406

Internet:
www.controlanything.com
www.controleverything.com
Www.iorelay.com

Sending Commands to NCD Devices

Most NCD devices are capable of sending and receiving data via RS-232 serial communications and are compatible with just about any computer or microcontroller ever produced, including the Macintosh, Basic Stamp, Windows, DOS, Linux, Unix, or any other operating system that supports serial communications.

Regardless of the system you are using, you will need access to a programming language that supports program control of the serial port on your system.

A terminal program is not suitable for controlling NCD devices. Commands should be sent using ASCII character codes 0-255 rather than ASCII characters (A, B, C etc.). See "ASCII Codes vs. Characters" on this page.

Most systems require you to open the appropriate serial port (COM port) prior to sending or receiving data.

Serial Communications Requirements
Your programming language should support commands for sending and receiving data from the serial port. It is important to familiarize yourself with these instructions for reliable operation. Different NCD devices support different baud rates, typically from 1200 Baud to 115.2K Baud. NCD Devices require 8 data bits, 1 stop bit, and no parity.

Because there are so many different ways to send and receive data from various languages on various platforms, we will provide generic instructions that can be easily converted to your favorite language.

For example, if this manual says "Send ASCII 254", the user will need to translate this instruction into a command that is capable of sending ASCII character code 254.

To Send ASCII 254 from Visual Basic, you will use the following line:

MSComm1.Output = Chr$(254)

In Qbasic, you can send ASCII 254 using the following line of code:

Print #1, Chr$(254);

Note that sending ASCII character code 254 is NOT the same as sending ASCII characters 2, 5, and 4 from a terminal program. Typing 2, 5, and 4 on the keyboard will transmit three ASCII character codes (3 bytes of data).

Sending HEX Values
For some programming languages, HEX values are preferred. ASCII values 0 to 255 found in this manual can be translated into hex values from $00 to $FF.

You may want to visit www.controlanything.com for the latest software and programming examples.

Programming examples are much more extensive for Visual Basic 6 users than for any other programming language. If you are not a VB programmer, you may consider looking at the VB6 source code, as it is easily translated into other popular languages.

ASCII Codes vs. Characters

The differences between ASCII codes and ASCII characters tend to generate a lot of confusion among first-time RS-232 programmers. It is important to understand that a computer only works with numbers. With regard to RS-232 data, the computer is only capable of sending and receiving numbers from 0 to 255.

What confuses people is the simple idea that the numbers 0 to 255 are assigned letters. For instance, the number 65 represents the letter A. The number 66 represents the letter B. Every character (including numbers and punctuation) is assigned a numeric value. This standard of assignments is called ASCII, and is a universal standard adopted by all computers with an RS-232 serial port.

ASCII characters codes can be clearly defined as numbers from 0 to 255 as it pertains to NCD products.

ASCII characters however are best defined as letters, A, B, C, D, as well as punctuation, !@#$%, and even the numbers 0-9.

Virtually all programming languages permit you to send ASCII in the form of letters or numbers. If you wanted to send the word "Hello" out the serial port, it is much easier to send the letters H, e, l, l, and o than it is to send the ASCII character codes that represent each letter.

For the purposes of controlling NCD devices however, it is much easier to build a numeric command set. Especially when communicating to devices that have numeric parameters, such as the E3C command set.

Put simply, it is easier to control NCD devices using ASCII character codes 0 to 255 than it is to use ASCII characters A, B, C, D, etc. In addition, NCD devices are far more efficient in processing data because there are fewer bytes to parse with a numeric system as opposed to a character based system.

Because terminal programs are ASCII character based, it may be difficult to generate the proper series of keystrokes that would be necessary to activate a particular function. Therefore, terminal programs are not suitable for controlling NCD devices. In a real world control application, a terminal program would not likely be used to control NCD devices anyway. Therefore, a programming language that supports the transmission and reception of ASCII character codes 0 to 255 is highly recommended.

NOTE: Some terminal programs allow you to send raw ASCII character codes. These terminal programming features are suitable for testing NCD devices.

Regardless of your programming background, the provided Visual Basic 6 source code is very easy to understand and will likely resolve any communication questions you may have. VB6 programming examples may be viewed in any text editor.

COM Ports and NCD Device Connection to your PC

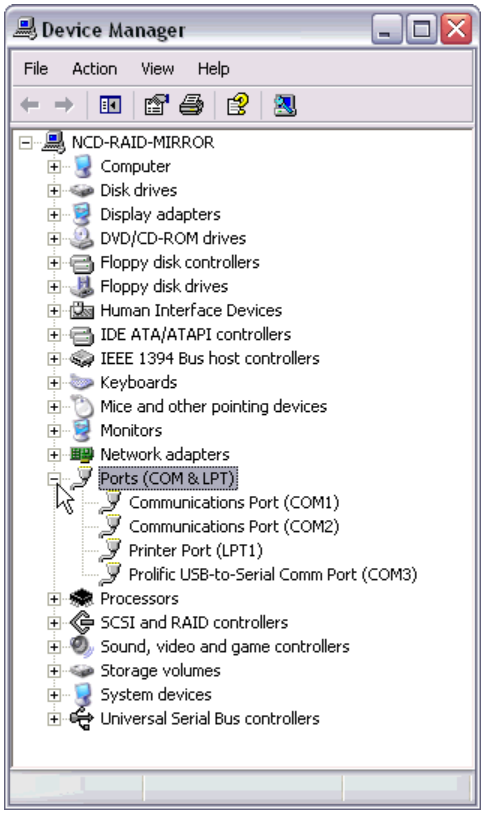
NCD Devices communicate with a computer via the COM port (also known as a serial port). Some computers do not have COM ports. For these computers, a USB to Serial Adapter should be used. Once installed, the USB Adapter will add a COM port to your PC.

Any software that is developed or used will access the serial port by referencing a COM port. If your computer has built in serial ports, the available COM ports are usually COM1 and COM2. In some cases, your computer will assign COM3 and COM4 as the available serial ports.

Adding a COM port via the USB to Serial Adapter usually assigns the COM port to COM3 or higher. In some cases, you can assign a different COM port to the USB adapter by simple plugging the adapter into a different USB connector on your PC.

It is very important to know what COM ports your system has available. To do this, follow these simple steps:

1) Click the Start button in Windows.
2) Select Control Panels.
3) Double Click on the "System" Icon.
4) Click the "Hardware" tab at the top of the window.
5) Click the "Device Manager" button.
6) The following window will open.
7) Click the "+" sign next to "Ports" to view the available COM ports on your system.



On our system, COM1 and COM2 are built into the motherboard. COM3 is also available as a USB to Serial adapter.

There are no special software requirements you will need to account for when using a USB to Serial Adapter. Most of today's adapters emulate COM ports flawlessly.

Refractor: The Mirror Module Setup Utility

We have developed a program that MUST be used to configure the Mirror Module. This program, titled "Refractor" is a highly interactive user interface that contains built in instructions on how to use every possible function. For this reason, the Refractor software will remain largely undocumented in a printed manual. Refractor does NOT allow the user to make any errors in settings. As various options are chosen, certain elements of the user interface will appear and disappear to prevent misconfiguration.

Refractor Hardware Requirements

Refractor expects to communicate to a Mirror Module interfaced to your computer via serial communications. A Quick Start kit as shown on page 15.A is required for proper communication between the computer and the Mirror Module. In addition, the PGM jumper should be installed and the RS-232/OC-232 jumper should be set to the RS-232 position. After these jumpers have been set, connect the power adapter to the Quick Start kit. You should see the LED on the Mirror Module flashing. You are now ready to run the Refractor Software.

Refractor Software Requirements

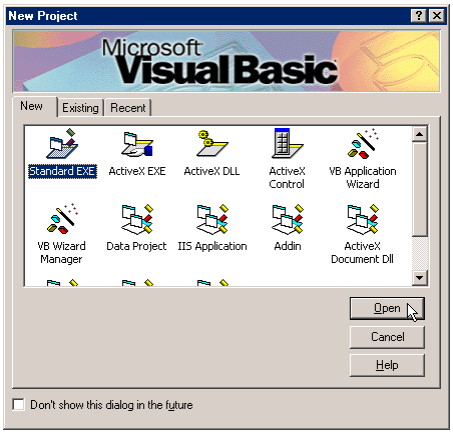
Refractor MUST be installed on a Windows XP Home or XP Professional system. We do not have a release for any other operating system, including older operating systems. In addition, Refractor REQUIRES Service Pack 2, and will not run without it. For best results, a fully updated computer, including .NET framework, should be installed on your computer when running any of our software.

Using Visual Basic 6 Professional with NCD Devices

Visual Basic is ideally suited for controlling NCD devices. We use Visual Basic 6 Professional, however, it is possible to use other versions of Visual Basic. This section of the manual describes the steps that must be taken to setup a Visual Basic Program to use NCD devices.

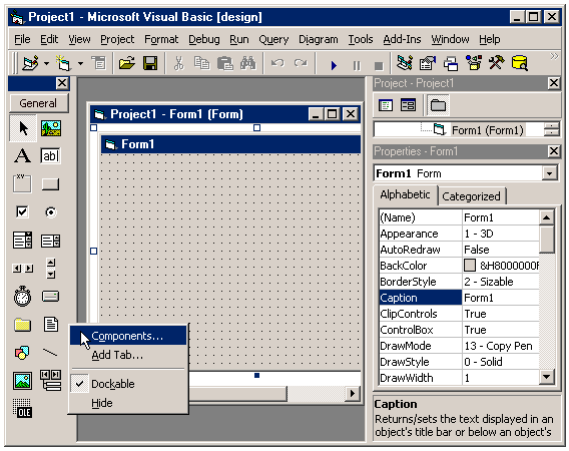
Visual Basic stocks a powerhouse of features that make it one of the most appealing and easy languages to use. Advanced users will find VB is ideally suited for intensely complex Multi-Media, Database, Mathematical, Video, and Imaging applications. If you would like to get an idea of just how powerful Visual Basic can be, please visit www.vbxtras.com and have a look at the hundreds of corporations who are developing extensions that add powerful capabilities to this amazing language.
The following guide will teach ANYONE how to operate ANY of our products using Visual Basic. The examples provided on this page were developed for Visual Basic 6 Professional. As you probably know, there are many versions of VB. The only version that is vastly different is .NET. We will discuss VB.NET later in this manual. But for now, if you are using any version of Visual Basic (other than .NET), follow these directions.

Step 1: Create a New Project

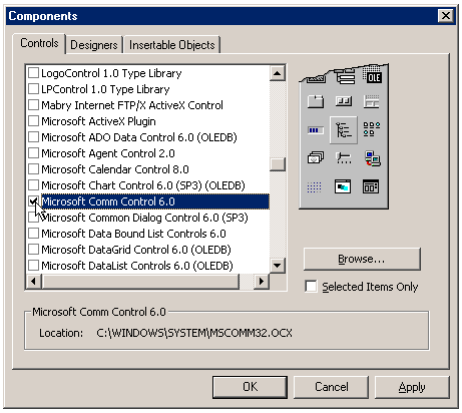


Start Visual Basic and Click on "Open".

Step 2: Add Serial Communications to your Program

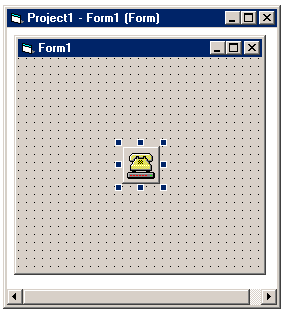


Right Click on the Tool Bar and Select "Components".



The Dialog Box shows at left will open. Scroll down and put a check next to "Microsoft Comm Control 6.0". If it does not appear in this list, you will need to use the NCD ActiveX control. Skip to the NCD ActiveX section of this manual on page 6.

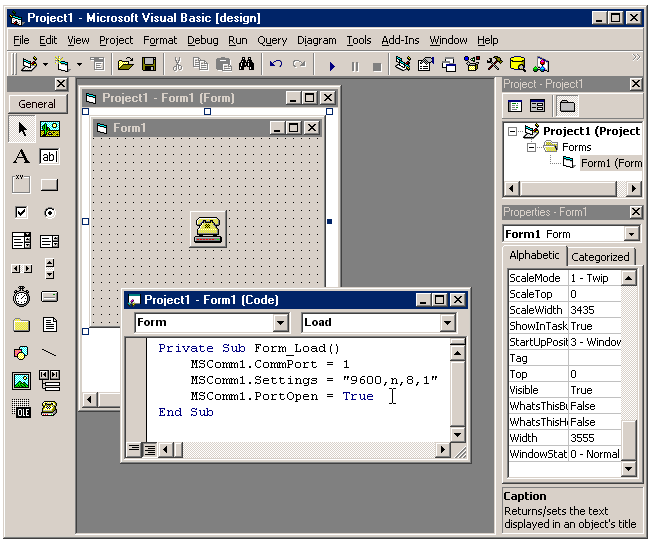
Your toolbox should now have the "Telephone" icon, which means serial communication can now be added to your project. Double Click the Telephone Icon to add Serial Communications to your program.



Step 3: Add Serial Communications to your Project

The Telephone Icon will appear in the middle of your form. Next, Double Click in the Gray area to to the telephone icon. This will open the "Form Load" window.

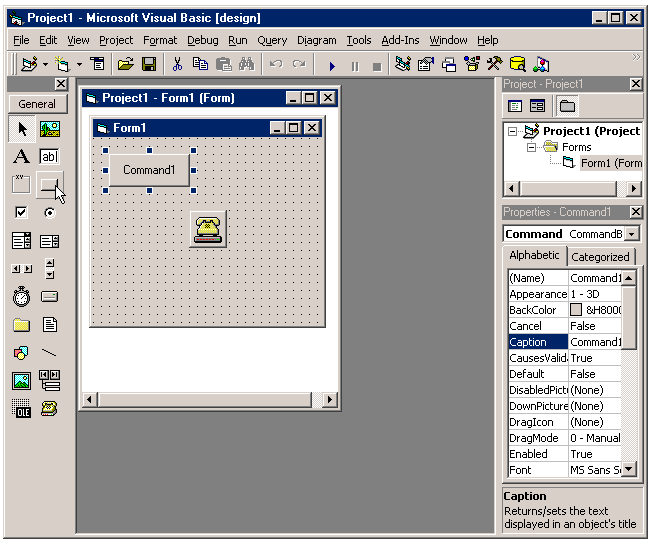
Step 4: Tell your Program to Open the Serial Port



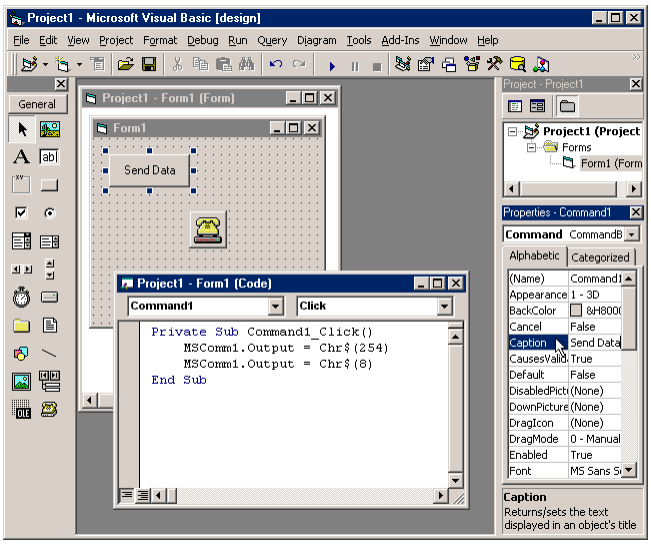
Type the commands shown in the "Form Load" window above. When you first run your program, these are the first lines of code that will be executed. These lines of code configure the COM port and communication speed. The CommPort setting may need to be changed to an available COM port on your computer. Once the port is configured, the PortOpen command is used to begin communication from your program to any of our devices. Once you have typed these lines of code, you can close the "Form Load" window.

Using Visual Basic 6 Professional with NCD Devices

Step 5: Send Data out the Serial Port to the NCD Device



In your Toolbox, double click the “Command Button: (as shown near the mouse pointer in the picture above). This will add a “Button: to your program. Double Click on the “Command1” Button you just created.



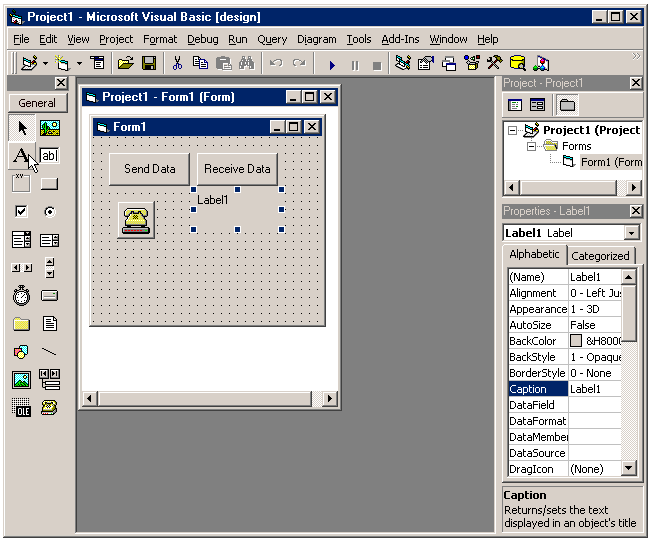
Double Clicking on the "Command1" button will open the "Command1 Click" window, which holds the lines of code that will be executed every time the user pushes the button. Type the lines of code shown above. These lines of code will activate the first relay on the R8x Pro series relay controller. You may substitute these lines of code for lines of code shown in the product manuals for our devices.

Also note the position of the pointer in the photo above. You can change the text on the button by changing the caption property in the "Properties" window. Change the caption to read "Send Data" as shown above.

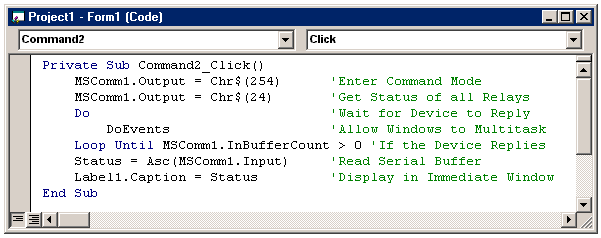
You are now ready to run your program. Simply click the Start button to run your program (small arrow pointing to the right on the upper tool bar).

When the “Send Data” button is clicked, data will be sent from Visual Basic to the relay controller. You should hear the first relay click on the relay controller and the corresponding LED should be lit. Note that commands are different for different devices, please reference the product you are controlling for the correct command sequence.

Step 6: Retrieve Data from the Device to your Computer In some cases, you may want to read data from an NCD device and process the result. This is particularly useful if you want to know the On/Off status of a relay, or if you want to read an A/D value from one of our devices that supports A/D conversion. The example shown below demonstrates reading data from our device and display the result in a Visual Basic program. We will use the R8x Pro to read the On/Off status of a relay.



Create a new button and label it "Receive Data". Create a Label by double clicking the "A" in the toolbox (shown near the pointer). The button will be used to send a request for the status of all 8 relays on an R8x Pro Relay Controller. The status will appear as a number from 0-255 "Caption" to the "Label".



Double click your new button, "Receive Data". The "Command2 Click" window will appear. Type the code shown above. When this button is clicked, a command will be sent to the device, requesting data from the controller. The program then monitors the receive buffer on your computer. If data is received from the device, it will be stored in the receive buffer. The program will wait indefinitely for data to appear in the receive buffer on your computer.

The DoEvents command is VERY important in Visual Basic. This command allows windows to Multi-Task. Without this command, your computer would appear to lock up while waiting for data to appear in the receive buffer. In actuality, your computer will not be locked up at all, instead, all windows tasks will not be serviced and all CPU power will be devoted to this portion of the program. You should ALWAYS use the DoEvents command when waiting for data from the serial port.

The controller will send a byte of data back to your computer. This byte of data is always a numeric value from 0-255. The byte of data will travel from our controller into the receive buffer of your computer. Once received, the Do/Loop portion of the program will detect the data. The next line of code reads the byte of data from the receive buffer on your computer using the MSComm1.Input command. The ASC portion of this command is used to convert the returned data into a numeric value from 0-255. The Label1.Caption=Status is used to display the returned value to the user.

Conclusion

These are the basic requirements for sending and receiving data using Visual Basic and a typical NCD device. NCD product manuals will have details regarding the commands the device is capable of accepting, as well as commands that send data back to the user.

Using the NCD ActiveX Control with Visual Basic

Regardless of the programming language you are using, the NCD ActiveX has become a preferred software component for using NCD devices. The NCD ActiveX is universally compatible with all NCD products, providing serial communications to any programming language that supports ActiveX controls. The NCD ActiveX has a complete library of commands built specifically for controlling NCD devices. In addition, it has generic communication functions, making it 100% compatible with ANY future NCD device we design.

.NET Compatibility Notice

The NCD ActiveX control can be used with .NET. However, the NCD ActiveX is NOT 100% compatible. You may experience problems closing the serial port, compilation problems, or other glitches. We have successfully and reliably used the NCD ActiveX control with .NET, but not without some odd behavior problems. We plan to release a .NET version of the NCD ActiveX control at some point in the future. Our web site will be updated with a .NET compatible version of the NCD ActiveX control.

What is the NCD ActiveX Control?

The NCD ActiveX control is best described as a very small, very efficient software component. The NCD ActiveX is not a program you can run. It is better described as an unfinished program that cannot work by itself. The NCD ActiveX control does two things notably well: 1) Enables any programming language to speak to NCD Devices via serial communications. 2) Provides a set of "Macros" for communicating to NCD devices using a more friendly English like structure.

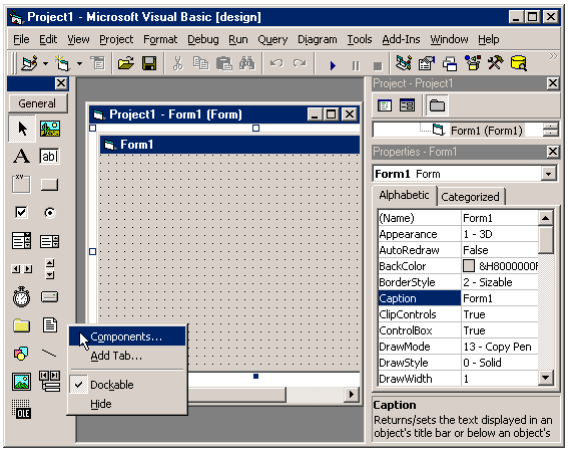
The only important requirement with regard to the NCD ActiveX control is that it MUST be used in conjunction with a programming language that supports ActiveX controls. Compatible programming languages include Visual Basic, Visual C++ and many other programs and languages.

NCD ActiveX Tutorial

Visual Basic is our preferred language for communicating to the NCD product line. This tutorial will show how to use the NCD ActiveX control with NCD devices. Before beginning this tutorial, you must download and install the NCD ActiveX control from our web site. Make sure you close all open applications BEFORE installing the NCD ActiveX. The following examples demonstrate the NCD ActiveX under VB6 Pro, but other versions of Visual Basic, including Standard and Learning editions are fully compatible with the NCD ActiveX control.

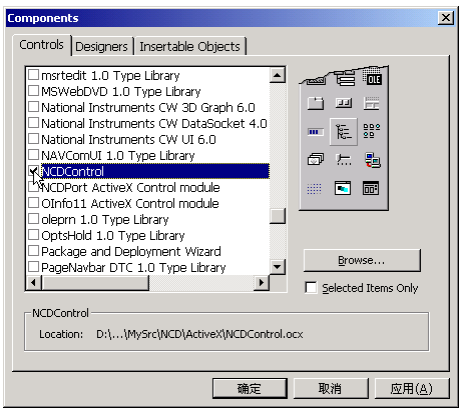
Step 1: Create a New Project

Start Visual Basic and Select Open.

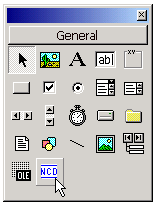


Step 2: Add NCD Control to your Program

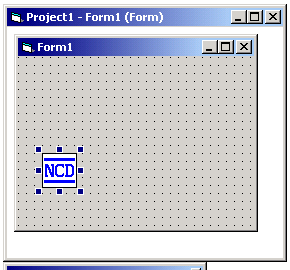
Right Click on the Tool Bar and Select "Components"



The Dialog Box (left) will Open. Scroll Down and put a check next to "NCD Control";. If it does not appear in the list, then you will need to install the NCD control. It can be get from www.controlanything.com. Next, Select "OK". This adds NCD Control to your toolbox.

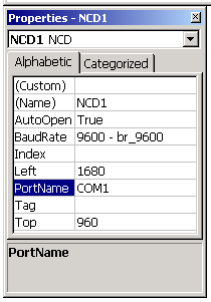


Your toolbox should now have the NCD icon on which there are text of NCD, which means NCD Control can now be added to your project. Double Click the NCD Icon to add NCD Control to your program.



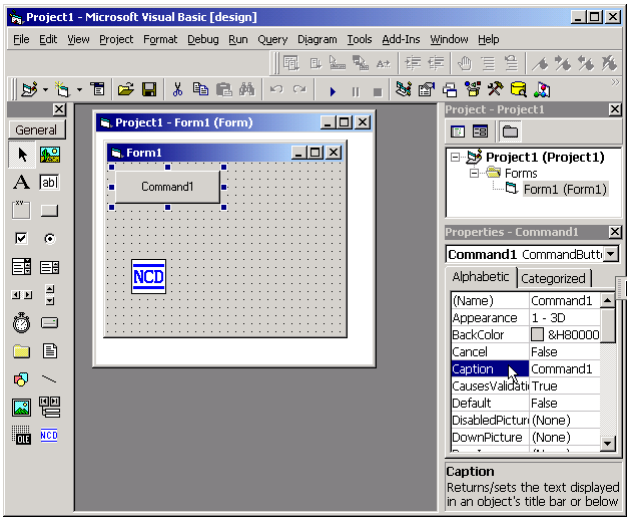
Step 3: Add NCD Control to your Project

The NCD Icon will Now Appear in the middle of your form. Keep the control selected.



Step 4: Set the Properties of NCD Control

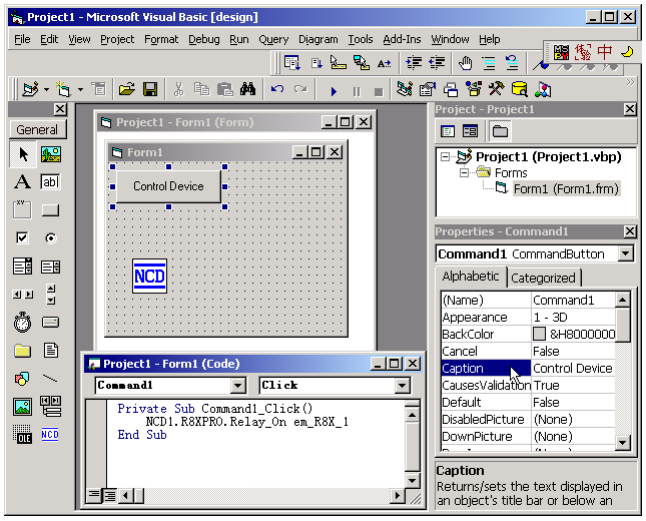
Use the properties window to set the COM port and the Baud rate. Make sure the NCD device is connected to the selected COM port and the device is set to the same baud rate.



Step 5: Control NCD Device with NCD Control

In your Toolbox, double click the "Command Button" (as shown near the mouse pointer above). This will add a "Button" to your program. Double Click on the "Command1" Button you just created.

Using the NCD ActiveX Control with Visual Basic



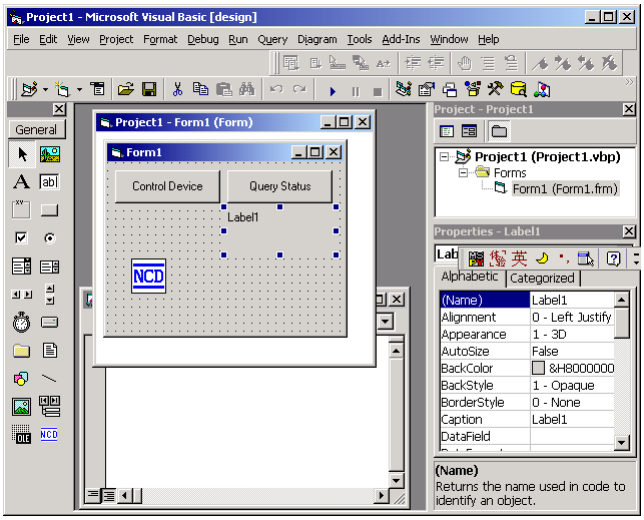
Double Clicking on the "Command1" button will open the "Command1 Click" window, which holds the lines of code that will be executed every time the user pushes the button. Type the lines of code shown above. These lines of code will activate the first relay on the R8x Pro series relay controller. You may substitute this lines of code for other method supplied in NCD ActiveX Control Manual (included with the ActiveX download).

Also note the position of the pointer in the photo above. You can change the text on the button by changing the caption property in the "Properties" window. Change the caption to read "Control Device " as shown above.

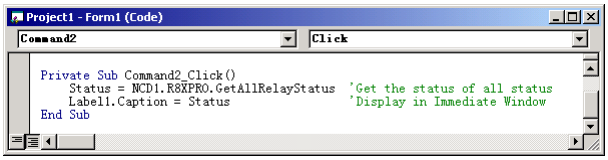
You are now ready to run your program. Simply click the Start button to run your program (small arrow pointing to the right on the upper tool bar).
That is all that is needed to control our products from Visual Basic. In no time, you will be able to create several buttons capable of controlling anything from your desktop computer.

Step 6: Retrieve Status from the Device to your Computer

In some cases, you may want to read status from an NCD device and process the result. For example, you may want to know the On/Off status of a relay, or if you want to read an A/D value from one of our devices that supports A/D conversion. The example shown below demonstrates reading relay status from R8x Pro and display the result in a Visual Basic program.



Create a new button and label it "Query Status". Create a Label by double clicking the "A" in the toolbox (shown near the pointer). The button will be used to read the status of all 8 relays on an R8x Pro Relay Controller. The status will appear as a number from 0-255 "Caption" to the "Label" for a successful reading, and appear as -1 for failed reading.



Double double click your new button, "Query Status". The "Command2 Click" window will appear. Type the code shown above. When this button is clicked, the program will read the status from device.The correct status is from 0 - 255. A return value of -1 represents a failure of reading status.

The Label1.Caption=Status is used to display the returned value to the user.

Conclusion

These are the basic requirements for controlling a NCD device with Visual Basic using the NCD ActiveX control. NCD ActiveX Control manual will have details regarding methods the device is capable of accepting, as well as function that query status of device.

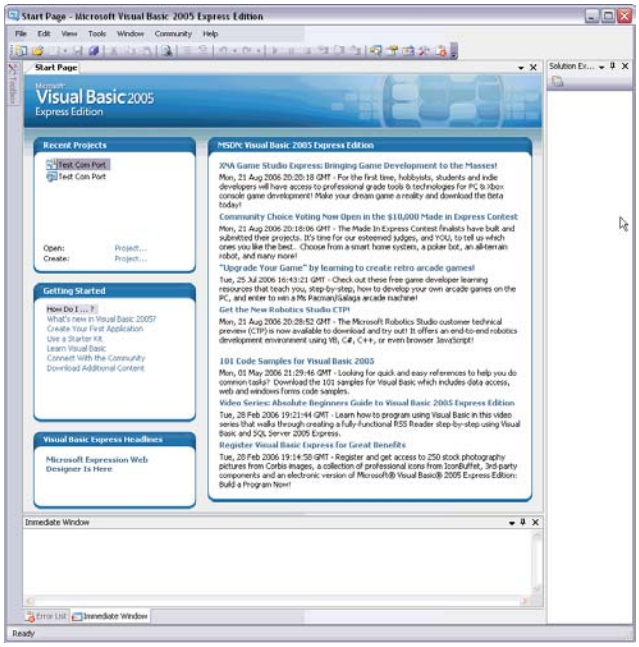
Serial Communications with Visual Basic 2005 Express Edition

In 2005, Microsoft released Visual Basic 2005 Express Edition. This version of VB is FREE and FULLY FUNCTIONAL. Though there are some limitations to the Express Edition, for the purposes of learning about computer control, it is an excellent place to get started. At the time of writing, you could download your free copy from this page: http://msdn.microsoft.com/vstudio/express/vb/

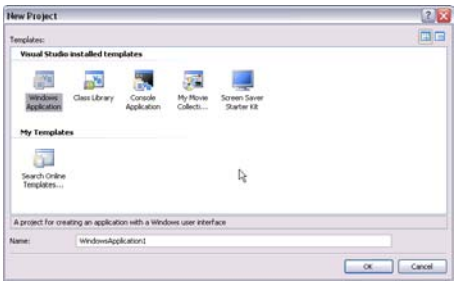
Click the large "Download Now" box on the right side of the screen.

Follow the installation instructions carefully. You will be required to register your copy with Microsoft to get your activation key (this only takes a few minutes).

Once installed, load VB 2005 Express Edition. You should see a window open similar to the one shown:

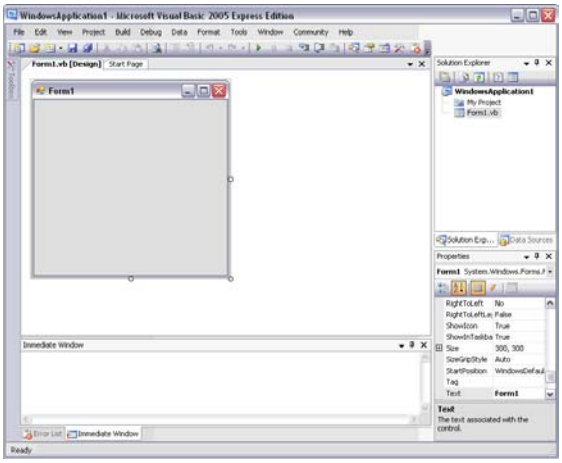


From the "File" pull-down window, choose "New Project". The following window will open:

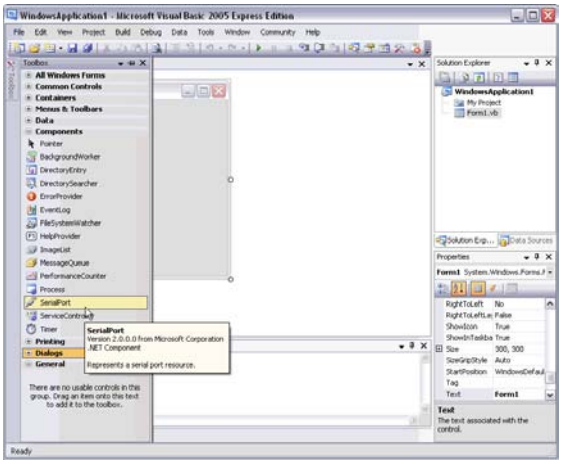


Choose "Windows Application" and click on "OK".

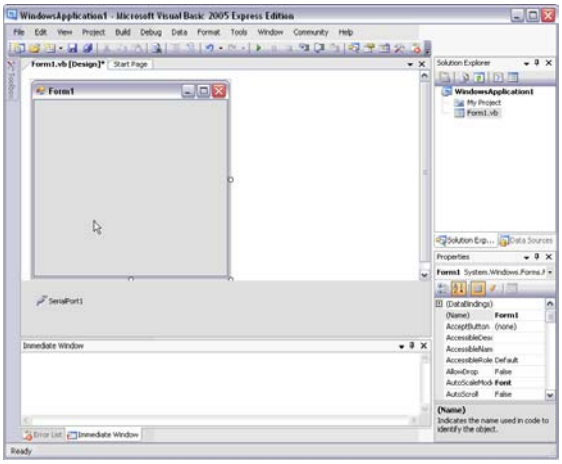
The following window will open:



Click on the "Toolbox" tab on the left side of the application window. A large list of tools will appear. You may want to make use the "+" and "-" boxes to the left of the various categories. Open the "Components" category and double click on SerialPort. This will add serial communications to your application.

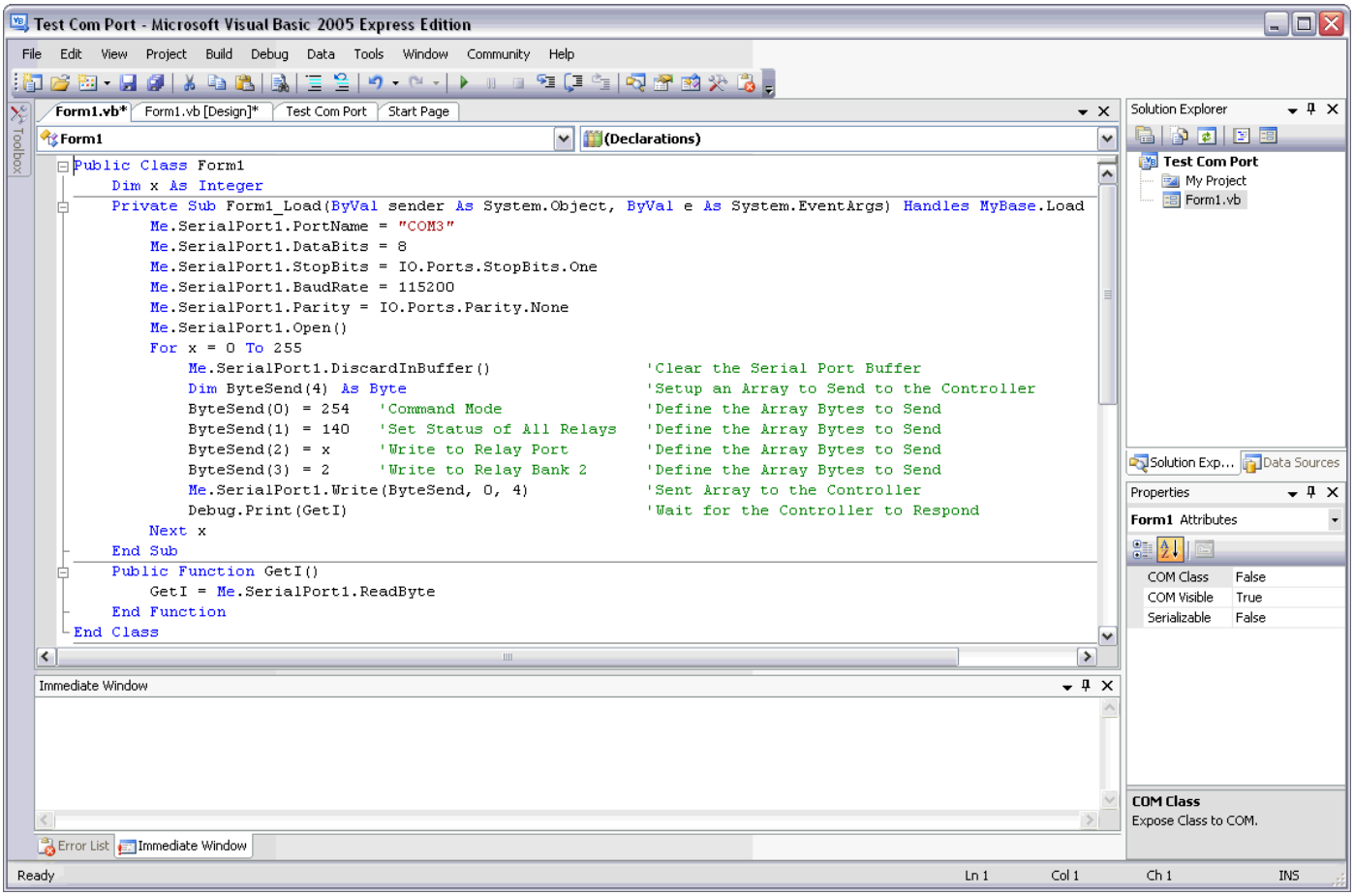


A gray bar will appear below the form. In the gray bar, the "SerialPort1" control will be shown. See the picture below:



Double Click on the Form (where the mouse pointer is in the above picture). This will allow you write your own program that can speak to NCD products.

Serial Communications with Visual Basic 2005 Express Edition



The above program was written to speak to the ProXR series relay controllers. Here is a detailed explanation of the program:

```
Dim x As Integer
```

This line is used to "setup" a variable. In this case, the variable x is setup. X will be used later in the program to set the status of the relays. X will hold a value of 0 to 255. When X = 0, all the relays will be off. When X = 255, all relays will be on. Every value in between will set the status of the relays in the equivalent binary patter of X.

```
Me.SerialPort1.PortName = "COM3"
Me.SerialPort1.DataBits = 8
Me.SerialPort1.StopBits = IO.Ports.StopBits.One
Me.SerialPort1.BaudRate = 115200
Me.SerialPort1.Parity = IO.Ports.Parity.None
Me.SerialPort1.Open()
```

The above lines of code are used to configure the serial port. In this case, we are using COM3 at the maximum allows baud rate of 115.2K baud. After these lines have been executed, the serial port will be open and ready for communication.

A for/next loop is used to count through all possible relay status data for a single relay bank:

```
For x = 0 To 255
```

The code in between the for/next loop is executed 256 times.

```
Next x
```

The following line is used to clear the serial buffer.

```
Me.SerialPort1.DiscardInBuffer()
```

Data is sent out the serial port one array at a time. This method is notably different from previous versions of Visual Basic. For the purposes of this example, we will be sending a command that contains 4 bytes of data, therefore, we must define an array of at least 4 bytes.

```
Dim ByteSend(4) As Byte
```

Next, data in the array must be defined to communicate to the device. In this case, the commands that set the status of all relays for the ProXR series controllers are used:

```
ByteSend(0) = 254   'Command Mode
ByteSend(1) = 140   'Set Status of All Relays
ByteSend(2) = x     'X Sets the Status of Relays
ByteSend(3) = 2     'Set Status of Relays in Relay Bank 2
```

Send the array out the serial port to the controller:
ByteSend is the array to send.
0 is the beginning of the relay.
4 sets the number of bytes in the array to send.

```
Me.SerialPort1.Write(ByteSend, 0, 4)
```

Wait for the controller to finish. The controller will send back an 85 when the command has finished execution. The 85 will be shown in the Immediate window. If the Immediate window is not visible, select the "Debug" pull-down menu, Choose "Windows", Choose "Immediate".

```
Debug.Print(GetI)    'Wait for the Controller to Respond
```

The GetI function is used to read data from the serial port (from the device).

```
Public Function GetI()
  GetI = Me.SerialPort1.ReadByte
End Function
```

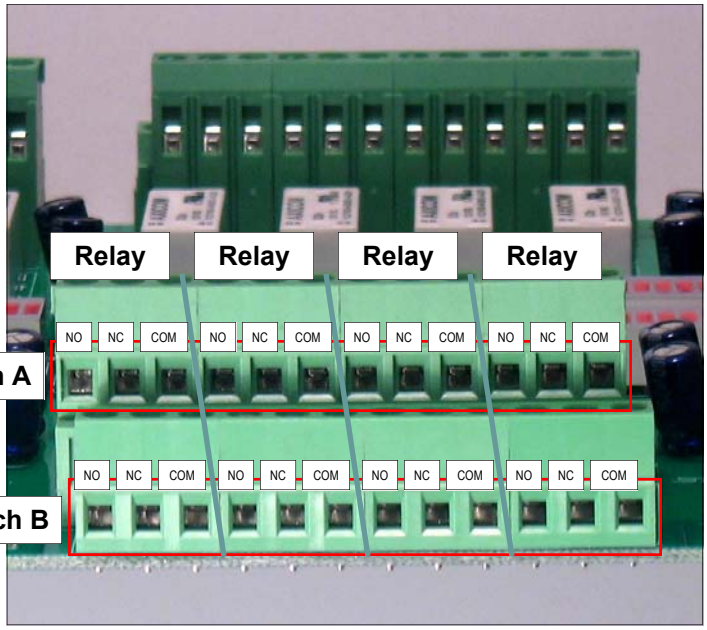
Basic Electronics Knowledge:

SPDT Relays
DPDT Relays
High Currents and Inductive Relays
FETS and FET Interface

Connecting 1-Amp DPDT Relays (White Relays) 12.A

Connector Pinouts for 1-Amp DPDT Series Controllers
Any time you see one of our relay controllers with a green 24-Position terminal block and small white relays labeled AXICOM on the board, you can assume the wiring connections shown to the right will always apply. Always view the connector from the angle shown in the picture to the right. This will ensure connections are consistently made to 1A DPDT series controllers.

A DPDT relay has two separate switches inside each relay. These switches are labeled Switch A and Switch B in the diagram at right. Switches A and B are completely separated from each other, there is no electrical connection between these switches. When a DPDT relay is activated, both switches A and B "activate" or more appropriately change position at the same time. In the diagram at right, each connection is labeled NO, NC and COM.



COM: Common
This terminal is connected to NC when relay is off.
This terminal swings over to NO when the relay is on.

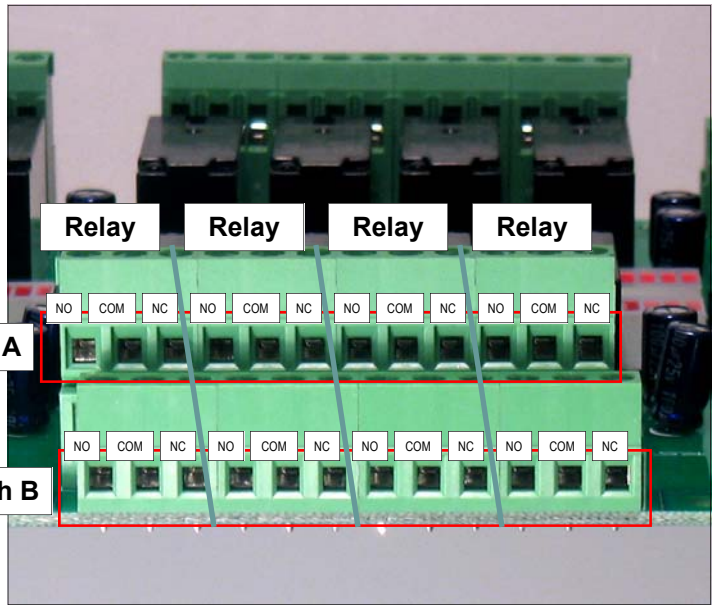
NO: Normally Open
This terminal has no connection with the relay is off.
This terminal is connected to COM when the relay is on.

NC: Normally Close
This terminal is connected to COM when the relay is off.
This terminal has no connection with the relay is on.

Connecting 3-Amp/5-Amp DPDT Relays (Black) 12.B

Connector Pinouts for 3-Amp and 5-Amp DPDT Series Controllers
Any time you see one of our relay controllers with a green 24-Position terminal block and black relays on the board, you can assume the wiring connections shown to the right will always apply. Always view the connector from the angle shown in the picture to the right. This will ensure connections are consistently made to 3A and 5A DPDT series controllers.

A DPDT relay has two separate switches inside each relay. These switches are labeled Switch A and Switch B in the diagram at right. Switches A and B are completely separated from each other, there is no electrical connection between these switches. When a DPDT relay is activated, both switches A and B "activate" or more appropriately change position at the same time. In the diagram at right, each connection is labeled NO, NC and COM.



COM: Common
This terminal is connected to NC when relay is off.
This terminal swings over to NO when the relay is on.

NO: Normally Open
This terminal has no connection with the relay is off.
This terminal is connected to COM when the relay is on.

NC: Normally Close
This terminal is connected to COM when the relay is off.
This terminal has no connection with the relay is on.

Connecting 5-Amp/10-Amp SPDT Relays 13.A

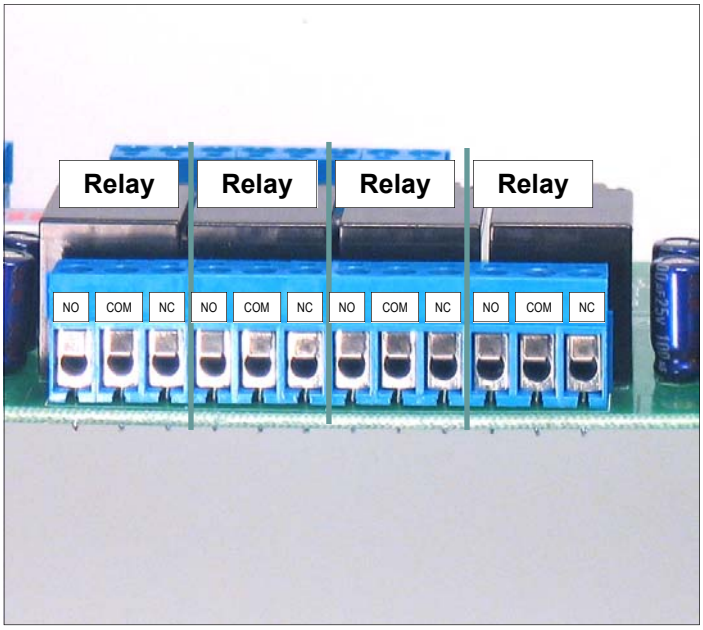
Connector Pinouts for 5-Amp / 10-Amp SPDT Series Controllers
Any time you see one of our relay controllers with blue 12-Position terminal blocks, you can assume the wiring connections shown to the right will always apply. Always view the connector from the angle shown in the picture to the right. This will ensure connections are consistently made to 5 and 10-Amp SPDT series controllers.

A SPDT relay has one switch inside each relay. When a SPDT relay is activated, the switch changes position. In the diagram at right, each connection is labeled NO, NC and COM. Please see the comments below to better understand how connections are made.

COM: Common
This terminal is connected to NC when relay is off.
This terminal swings over to NO when the relay is on.

NO: Normally Open
This terminal has no connection with the relay is off.
This terminal is connected to COM when the relay is on.

NC: Normally Close
This terminal is connected to COM when the relay is off.
This terminal has no connection with the relay is on.

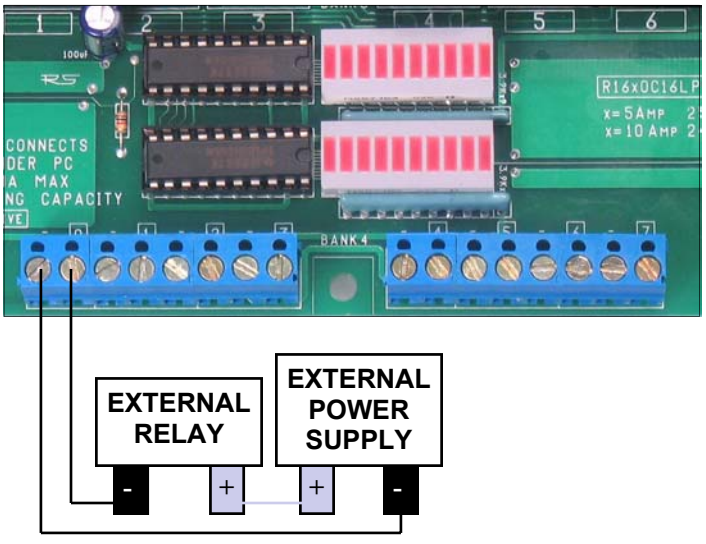


Connecting O.C. Outputs to External Devices 13.B

Connecting Open-Collector Outputs
WARNING: DO NOT CONNECT OC OUTPUTS WHILE POWER IS APPLIED TO THE CONTROLLER. THIS WILL DAMAGE THE OUTPUT DRIVER CHIP.
Any time you see one of our ProXR Series relay controllers with blue terminal blocks (8 or 16-Position), you can assume the wiring connections shown to the right will always apply. Always view the connector from the angle shown in the picture to the right. This will ensure connections are consistently made to the OC Outputs.

Open Collector outputs are ideal for adding external relays, small lights, LEDs, and other small loads under 150ma at 12VDC. OC outputs work by connecting an external device to ground. For instance, to connect an external relay to an open collector output, you will connect one of the two power leads directly to the OC output. The other relay power lead should be connected to a power supply. The ground of the power supply should be shared with the Ground leads on the OC output.

When the OC output is activated, the relay will activate by making a connection between the relay and ground. A voltage should always be present on the relay (except during connection).



Repeat the circuit as shown above for every pair of outputs. On some controllers, the two sets 8-position blue terminal blocks are together, forming a single blue 16-position terminal block. Connections are the same for these controllers. Don't forget, this circuit will share the grounds of your external power supply with the ground and RS232 ground of the controller. NEVER CONNECT EXTERNAL DEVICES WHILE THE CONTROLLER IS POWERED UP.

The E3C Command Set: Software Control of Multiple NCD Devices

The E3C command set allows you to control up to 256 NCD devices from a single serial port. It is OK to mix different types of devices, as long as the devices are E3C compliant.

How does E3C Work?
First of all, each device must be assigned a device number from 0 to 255. Most devices MUST be set to "Configuration Mode" to program the E3C device number. This prevents the device number from being accidentally changed under normal use of the device command set. However, some devices may handle this function differently. Please consult later sections of this manual that pertain to your specific device for E3C device number programming.

E3C stands for Enabled 3-Wire Communication. Put simply, when you first power up your computer and all the devices attached to the serial port, all devices will respond to your commands.

Using the E3C command set, you can specify which devices will listen and which devices will ignore your commands. Note that E3C commands are never ignored by any device, regardless of the commands you send to the controller.

The number to the left of each command indicates the ASCII character code that must be sent to issue the command. All commands must be preceded with ASCII character code 254 to place the device in command mode. See examples at right.

The E3C Command Set

254, 248 Enable All Devices:
Tells all devices to respond to your commands.

254, 249 Disable All Devices:
Tells all devices to ignore your commands.

254, 250 Enable a Selected Device:
Tells a specific device to listen to your commands.

254, 251 Disable Selected Device:
Tells a specific device to ignore your commands.

254, 252 Enable Selected Device Only:
Tells a specific device to listen to your commands, all other devices will ignore your commands.

254, 253 Disable a Selected Device Only:
Tells a specific device to ignore your commands, all others will listen.

E3C Visual Basic Programming Examples

The E3C command set is easily used from any programming language that supports serial communication. The following Visual Basic 6 Example source code demonstrates subroutines that can be used to control which devices will listen and which devices will ignore your commands.

Some devices will acknowledge your commands by sending ASCII character code 85 back to the host compute. E3C commands are never acknowledged.

Sample Code: The E3C Command Set

```
'Enable All E3C Devices
MSComm1.Output = Chr$(254)   'Enter Command Mode
MSComm1.Output = Chr$(248)   'E3C Enable All Device Command

'Disable All E3C Devices
MSComm1.Output = Chr$(254)   'Enter Command Mode
MSComm1.Output = Chr$(249)   'E3C Disable All Device Command

'Enable A Specific E3C Devices, Other Devices will be unchanged
MSComm1.Output = Chr$(254)   'Enter Command Mode
MSComm1.Output = Chr$(250)   'E3C Disable Specific Device Command
MSComm1.Output = Chr$(Device) 'Device Number that will be Disabled

'Disable A Specific E3C Devices, Other Devices will be unchanged
MSComm1.Output = Chr$(254)   'Enter Command Mode
MSComm1.Output = Chr$(251)   'E3C Disable Specific Device Command
MSComm1.Output = Chr$(Device) 'Device Number that will be Disabled

'Disable All E3C Devices Except (Device)
MSComm1.Output = Chr$(254)   'Enter Command Mode
MSComm1.Output = Chr$(252)   'E3C Disable All Device Except Command
MSComm1.Output = Chr$(Device) 'Device Number that will be Active

'Enable All E3C Devices Except (Device)
MSComm1.Output = Chr$(254)   'Enter Command Mode
MSComm1.Output = Chr$(253)   'E3C Enable All Device Except Command
MSComm1.Output = Chr$(Device) 'Device Number that will be Inactive

Note:
NCD ActiveX user can use the following methods to simplify programming
(please review pages 6 and 7 for details):

NCD1.E3C.DisableAllDevices
NCD1.E3C.DisableAllDevicesExcept Device
NCD1.E3C.DisableSpecificDevice Device
NCD1.E3C.EnableAllDevices
NCD1.E3C.EnableAllDevicesExcept Device
NCD1.E3C.EnableSpecificDevice Device

'The following NCD ActiveX methods are not supported by all NCD devices.

NCD1.E3C.ProgramDeviceNumber Device
NCD1.E3C.RecallDeviceIdentification
NCD1.E3C.RecallDeviceNumber

Device = a Device Number from 0-255
```

Quick Start Kit Wiring Standards

NCD Devices support only a few ways to connect a device to a computer. The most common way is to use a QS12 quick start kit. However, there are other forms of connection available to some devices. Please review the photos for connecting individual devices to a computer. Examine your device and compare it to the photos on this page to determine the wiring convention used to connect your device to a computer.

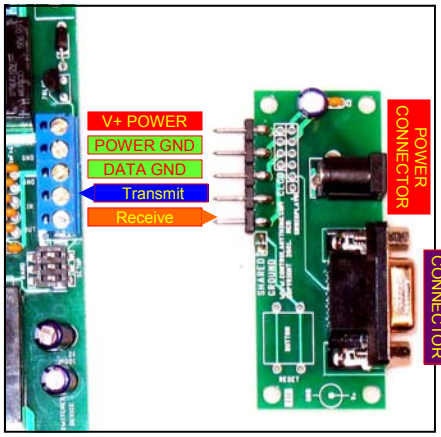
Quick Start Kits include a Power Supply (5VDC for QS5 models, 12VDC for QS12 models), Serial Cable, and RSIO Adapter. The RSIO adapter is slightly different for some versions of quick start kits as noted below.

NOTE: The Serial Cable Supplied with the Quick Start Kit is wired as a simple Serial Extension Cable with a DB9 Female Connector on one end and a DB9 Male connector on the other end.

QS5/QS12 Quick Start Kit 15.A

NOTE: RSIO Serial Adapter Includes 5 Prongs

The QS5 and QS12 quick start kits are the most widely used standards for connecting a single NCD device to a computer. This simple connection system provides +5 or +12 Volts of DC power, RS-232 Data Input, RS-232 Data Output, and the appropriate ground lines required for reliable communications and power (sometimes shared). Look for the 5 Position blue connector on your device and match it to the photo below. The prongs go into the 5 position terminal block (blue connectors). Each of the 5 screws should be tightened for reliable operation.



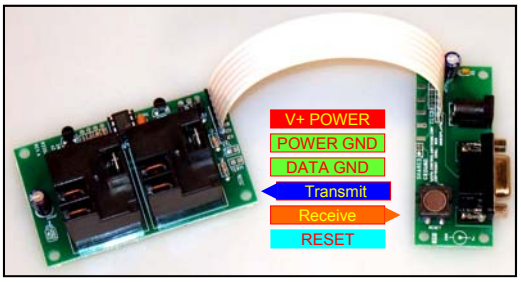
Power is supplied to the NCD device through the power connector on the Quick Start Kit.

Data flows from your computer into the device via the Transmit data line (blue). Data flows from the device to the computer via the Receive data line (orange).

QS5-F6/QS12-F6 Quick Start Kit 15.B

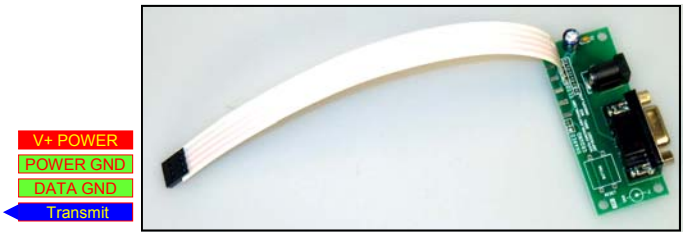
NOTE: RSIO Serial Adapter Includes 6 Conductor Flex Cable

The QS5-F6 and QS12-F6 quick start kits are most commonly used on NCD devices where space is limited, but 2-way communications is supported by the connected device. This wiring standard is the same as the QS5/QS12, but also includes a RESET line that can be used to reboot the CPU on some devices that support this feature. Look for the 6 position header positioned near the outside of the NCD device that supports this wiring standard.



The QS5–F6 and QS12-F6 Versions of the Quick Start Kit include a 6-Conductor Flex Cable that plugs directly into the NCD device. The pin-out for this ribbon cable is identical to the quick start kit shown at left. The 6th conductor (slightly discolored from the other conductors of the ribbon cable shown in the photo above) is a RESET line used by some older NCD devices. The F6 version shown above includes a RESET button. Today's NCD devices do not use the RESET button, therefore the F6 series quick start kits do NOT include a reset button. At some point, future devices may need to take advantage of this feature.

QS5-F4/QS12-F4 Quick Start Kit 15.C

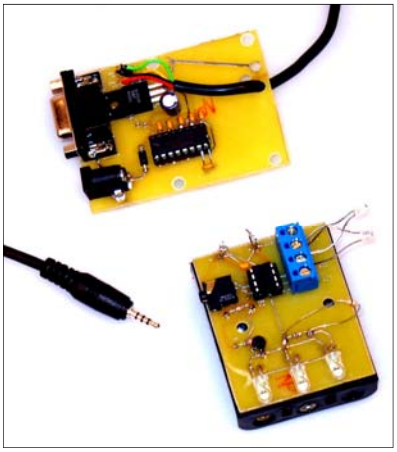


NOTE: RSIO Serial Adapter Includes 4 Conductor Flex Cable

QS5-F4 and QS12-F4 are identical to the QS5-F6 and QS12-F6 with the exception that only 4 conductors are used and one way communication (to the device) is supported.

PJ4 Quick Start Kit 15.D

The Phone Jack 4-Conductor (PJ4) standard was introduced in 2007 to interface small devices that have temporary connection to a PC or require communication to a PC with minimal space and cost. When a PJ4 plug is connected to an NCD device, the device automatically switches to configuration mode for configuring the device. A PJ4 adapter is REQUIRED by some devices. A PJ4 adapter is used by our devices that have a heavy emphasis on consumer applications. A PJ4 adapter includes a power supply, serial cable, and an enclosed converter that adapts a serial DB9 connector into the PJ4 standard.

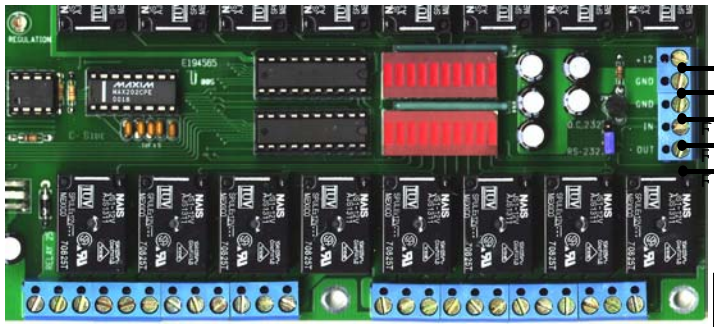


An early prototype of a PJ4 Quick Start Kit is shown at left.

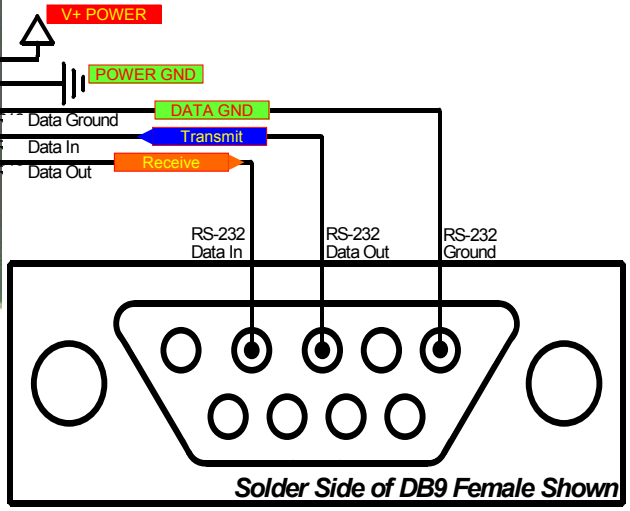
Hardwired RS-232 Connection to NCD Devices

NCD Devices can be wired to your computer without the use of a Quick Start kit. Follow the wiring diagrams shown below for details. Devices that require a F4 or F6 flex cable (as described on the previous page) can be wired in the same way, simply follow the color codes on this page with the color codes of the connectors shown on the previous page. The order of connection matches the position of the photos shown on the previous page.

Different NCD devices have different power requirements. Most devices require +12 Volts DC, but some devices require +5 volts DC. Never exceed the voltage rating of the device, doing so will damage the controller.



Note the connector (right) is a DB9 Female. The diagram illustrates connection to the solder side of the connector. This connector is plugged into the computer or microcontroller.



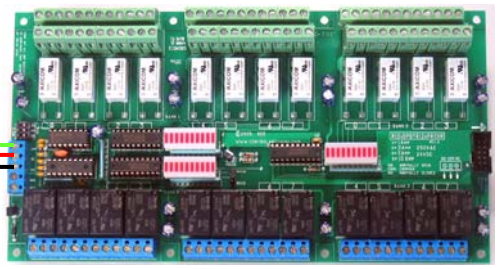
E3C Wiring Standards for Networking NCD Devices on a Single Serial Port

NCD Devices that support E3C networking via the RS-232 port should be configured with a E3C device number BEFORE networking. The E3C command set allows you to speak to one or more devices connected to a single serial port using the E3C device number. The software component of E3C is described in detail on page 13 of this manual.

Different NCD devices may have different ways of configuring a E3C device number. Later sections of this manual will describe how to program a E3C device number into a given device.

Once the E3C device number has been programmed, you may begin networking NCD devices on a single serial port. As a general rule, the RSB serial booster will be required for networking multiple devices together. The RSB serial booster facilitates 2-way communication with multiple devices. The RSB is not required for 1-way communications with multiple devices.

E3C networking is compatible only with NCD devices with a blue RS-232/power connector unless otherwise noted.



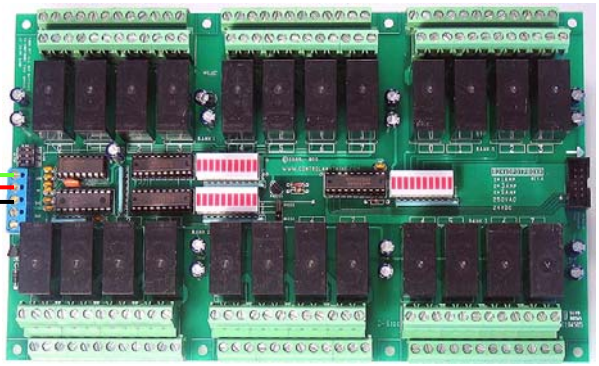
The Output of the RSB serial booster is connected to the Input on EACH device.

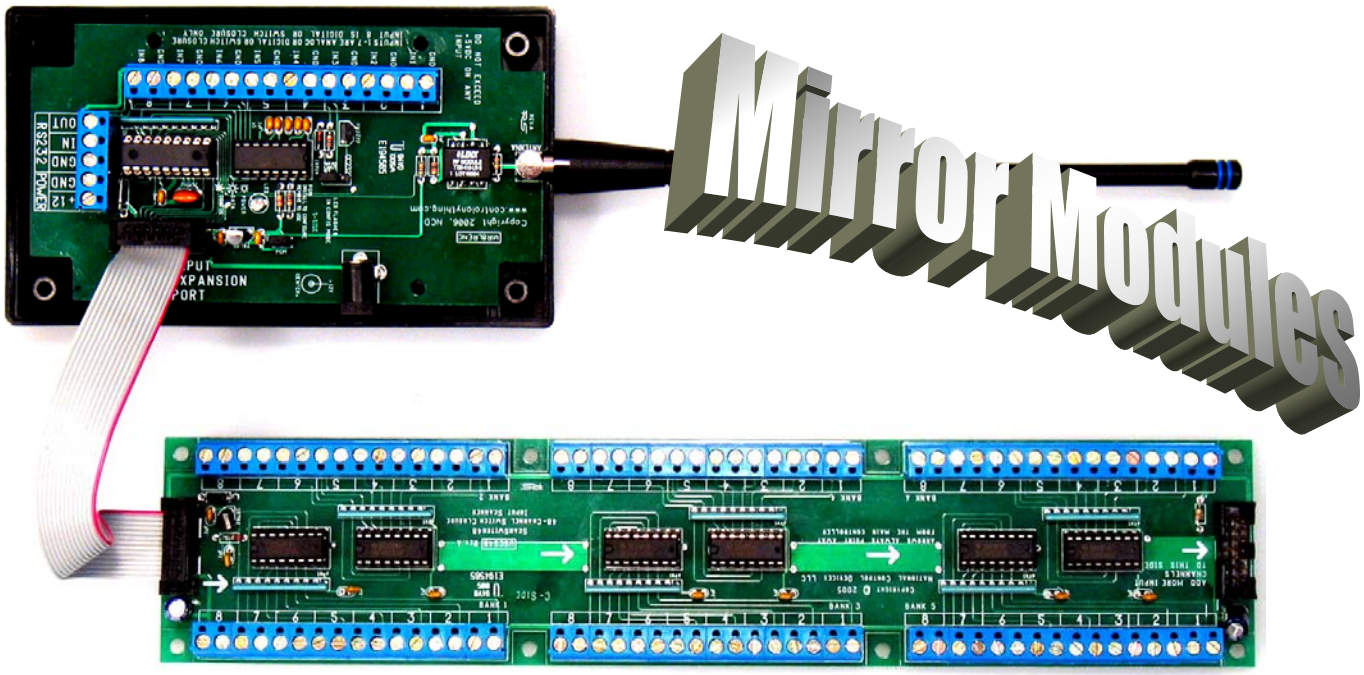
The Input on the RSB serial booster is connected to the output on EACH device.

The Ground on the RSB serial booster is shared with the Ground on every device on the network.

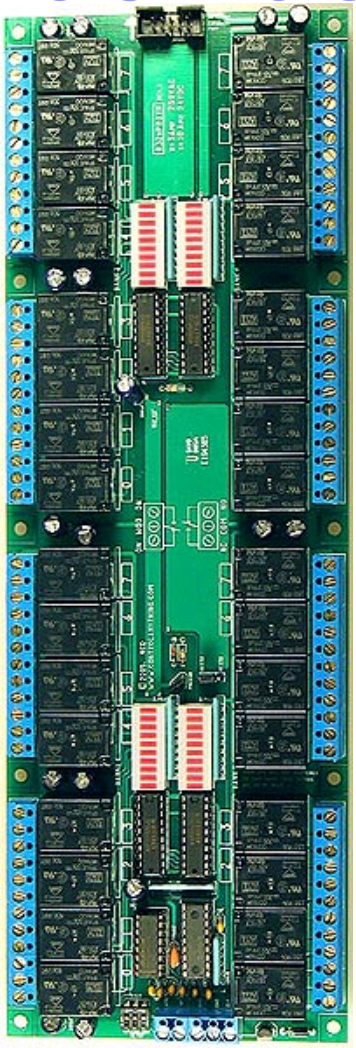
Each device should have its own power supply, including the RSB serial booster.

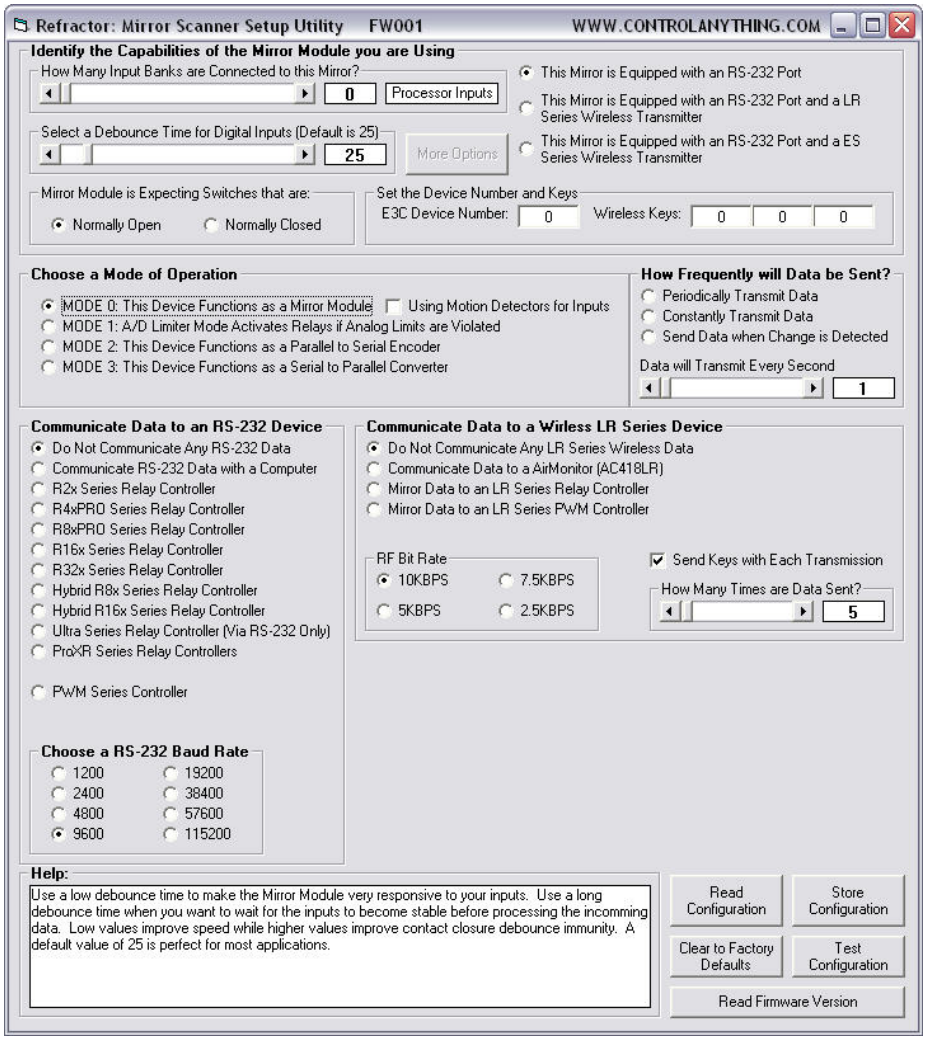
IMPORTANT: YOU MUST SET EACH INDIVIDUAL DEVICE TO OC232 WHEN NETWORKING MULTIPLE CONTROLLERS TO A SINGLE RSB SERIAL BOOSTER.





The Universal Remote for NCD Devices!





Mirror Modules: Introduction

Mirror Modules are best described as being the exact opposite of a relay controller. They take a switch closure input and generate data that can be interfaced to a NCD device or to a computer, either by wireless RF or hardwired RS-232 communications.

Mirror Modules effectively take the computer out of our line of computer controlled relay boards, allowing remote operation of our extensive line of relay controllers. When used in conjunction with our relay controllers, Mirror Modules effectively give you contact closure inputs on one side, contact closure outputs on the other side, and only 2 wires of communication in between the inputs and outputs. Wireless Mirror Modules act as a remote control for our wireless line of relay controllers, offering highly reliable contact closure replication up to 3,000 feet away.

Mirror Modules arose from customers needs to gather data and replicate it in a remote location with few or no wires. Mirror Modules have literally thousands of possible configurations. It is up to our customer to configure the Mirror Module for their specific needs. Mirror Modules MUST be connected to a computer to configure various user parameters. Once configured, the Mirror Module will ALWAYS process data according to your settings. Configuration settings are stored in non-volatile memory. A Mirror Module will not function without configuration. By default, the Mirror Module has no function, and MUST be connected to a PC for configuration prior to use.

Refractor: The Mirror Module Setup Utility

We have developed a program that MUST be used to configure the Mirror Module. This program, titled "Refractor" is a highly interactive user interface that contains built in instructions on how to use every possible function. For this reason, the Refractor software will remain largely undocumented in a printed manual. Refractor does NOT allow the user to make any errors in settings. As various options are chosen, certain elements of the user interface will appear and disappear to prevent misconfiguration.

Refractor Hardware Requirements

Refractor expects to communicate to a Mirror Module interfaced to your computer via serial communications. A Quick Start kit as shown on page 15.A is required for proper communication between the computer and the Mirror Module. In addition, the PGM jumper should be installed and the RS-232/OC-232 jumper should be set to the RS-232 position. After these jumpers have been set, connect the power adapter to the Quick Start kit. You should see the LED on the Mirror Module flashing. You are now ready to run the Refractor Software.

Refractor Software Requirements

Refractor MUST be installed on a Windows XP Home or XP Professional system. We do not have a release for any other operating system, including older operating systems. In addition, Refractor REQUIRES Service Pack 2, and will not run without it. For best results, a fully updated computer, including .NET framework, should be installed on your computer when running any of our software.

Quick Start Guide

Step 1: Download and Install "Refractor" from our web site. This program can be found on the Product Description Pages for ALL Mirror Modules.
Step 2: Install Program/Run Jumper on the Mirror Module.
Step 3: Connect the QS12 Quick Start Kit to the Mirror Module as Shown on Page 15.A.
Step 4: Apply Power to the Mirror Module. The LED Should be Flashing, indicating the Mirror Module is Ready for Configuration.
Step 5: Run "Refractor", choosing the correct COM port for your system.
Step 6: Use the Guide Box at the Bottom of the Refractor Software to guide you through device configuration options and settings.
Step 7: Store your configuration into the Mirror Module.
Step 8: Remove the Program/Run Jumper and Disconnect the Quick Start Kit.
Step 9: Connect the Mirror Module to your Remote Device if you have Chosen RS-232 Communications.
Step 10: Power the Remote Device and the Mirror Module.
Step 11: Touch the Contact Closure Inputs Together on the Mirror Module. The LED Should Flash Brightly when Connected Together.
Step 12: Verify the Remote Device is Responding to your configuration.

Troubleshooting:

If the remote device is not responding, then likely the configuration is not correct for the device you are attempting to control. If you experience problems with an RS-232 device, make sure the correct RS-232 device is chosen during configuration.

Hint for Wireless Users:

If you see an LED flash on a remote device, but the device does not respond, then everything is working, but a configuration setting must be changed. Most likely, the Mirror Module is sending keys, or is not sending keys. If this setting is in conflict with your remote device, the LED will flash without a proper response. Make sure both the Mirror Module and the remote device are set to Require Keys (and to Send Keys), and that the keys for both devices are the same. Otherwise, make sure the Mirror Module and the Remote Device are both set to NOT send or require keys.

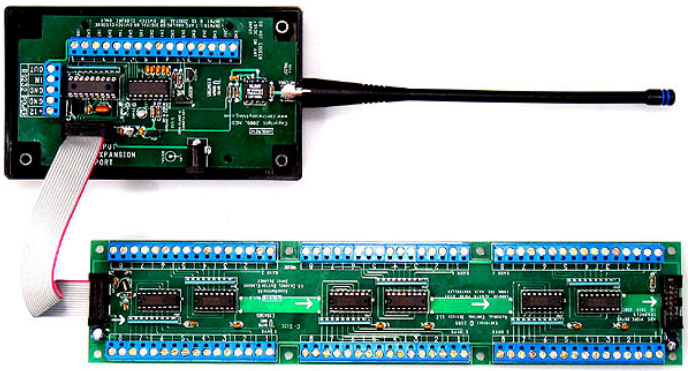
Hint for All Users:

The Guide Box at the Bottom of the Refractor Software provides detailed instructions and help for all settings. Some settings are not available for some versions of the Mirror Module. Refractor will read the version connected and hide or display options accordingly. Some interface options become available only when used in combination with other options. The interface was designed to prevent the user from making a configuration error. All rules have been carefully programmed and tested.

Mirror Modules

Mirror Modules are COMPLETELY reliant on our “Refractor” software, used to configure the Mirror Module. The capabilities of the Mirror Module are dependant on which version of the Mirror Module you are using. The screen shot shown (right) applies to the MIR8LRENC. MIR = Mirror Module, 8 = 8 Contact Closure Input Channels Built into the Mirror Module, LR = LR Series Transmitter is built it (allowing you to communicate to LR series relay controllers), and ENC = ENCLOSURE, meaning the format of the PCB will fit an enclosure. Other versions of the Mirror Module have 32 built in contact closure inputs, and is designated with a 32 in the part number. Also available is an ES option for communicating to ES series controllers.

Mirror Modules are also expandable, allowing you to add contact closure inputs to the controller. The photo below shows the MIR8LRENC connected to a USCS32 32-Channel Contact Closure Expansion Module.



What a Mirror Module does is completely up to you as a user within the constraints of the basic idea that a Mirror Module will generate data based on the contact closure inputs. The data that is generated by the Mirror Module can be delivered directly to an NCD device, such as a relay controller or PWM series controller. These data can also be delivered to a computer. A Mirror Module with an RS-232 interface has only one means of communication: RS-232. So once the device is configured, it can only communicate RS-232 data to a computer or to a NCD device. A Mirror Module with a Transmitter built in has two methods of communication: RS-232 and Wireless RF data transmission. These Mirror Modules can speak to a wireless NCD device (or a computer with an RF receiver), they can also speak to a NCD relay controller or a PC via RS-232 communications. The point being, you will use our Refractor software to configure how data is communicated using each output method. Here are a FEW examples:

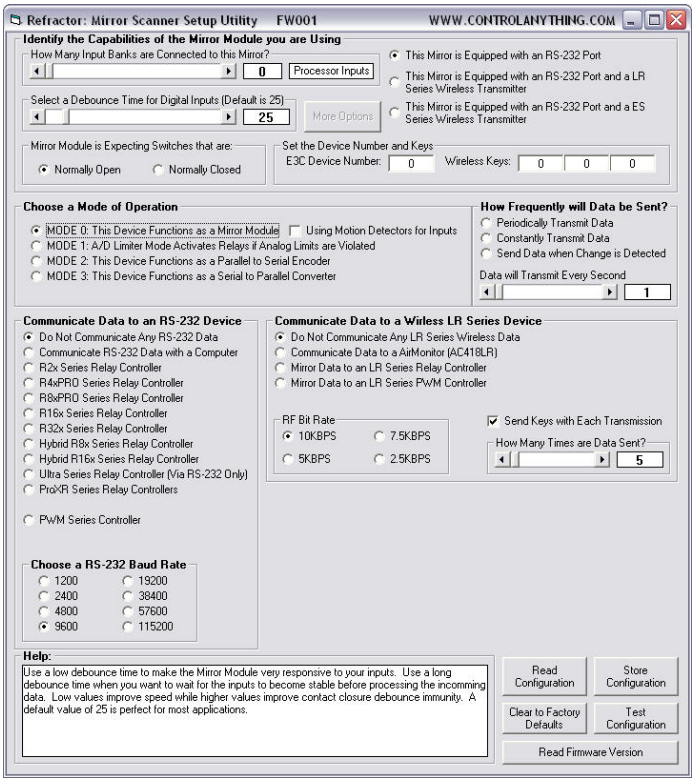
MM Speaks to NCD Device via RS-232 Communications
MM Speaks to Computer via RS-232 Communications
MM Speaks to NCD Device via Wireless Communications
MM Speaks to Computer via Wireless Communications

All Mirror Modules are capable of speaking to a device via RS-232 communications. This output method supports the following configurations:

MM Speaks to NCD Device via RS-232 Communications

-or-

MM Speaks to Computer via RS-232 Communications



Mirror Modules with two output methods (RS-232 and wireless) can speak to two devices at the same time. For example, a single Mirror Module can do the following at the same time:

MM Speaks to NCD Device via RS-232 Communications
MM Speaks to NCD Device via Wireless Communications

-or-

MM Speaks to NCD Device via RS-232 Communications
MM Speaks to Computer via Wireless Communications

-or-

MM Speaks to Computer via RS-232 Communications
MM Speaks to Computer via Wireless Communications

-or-

MM Speaks to Computer via RS-232 Communications
MM Speaks to NCD Device via Wireless Communications

IMPORTANT:
Before you can run Refractor, the Quick Start Kit MUST be connected as shown on Page 15.A. You MUST install the Program/Run jumper and THEN power-up the Mirror Module. The LED on the Mirror Module should be flashing. Once you have completed your configuration, the Program/Run jumper should be removed and the Mirror Module should be power cycled. The LED on the Mirror Module will ONLY flash when data is transmitted. During regular standby (no data communication), the LED will be dim.

Mirror Modules

Mirror Modules are capable of doing so much more and there are many options to discuss. These options will help you better understand the capabilities of a Mirror Module. But here are some example capabilities (which will be discussed in the coming pages):

MM can Mirror Contact Closure Inputs on a Remote Relay Controller via wireless and/or RS-232 communications.

MM can be wired directly to Motion Detectors and activate lights for a given period of time if motion is detected (you specify the time).

MM can be wired to sensors. Connecting a MM to a CDS photocell, you can activate a light when it gets dark outside. Or, you can activate a heater if it gets too cold (when wired to a thermistor). You can set the thresholds for activating remote relays.

MM can also mirror data to a PWM series controller. In this mode, the first 7 inputs are converted to 0-5VDC analog inputs. These data are forwarded to the PWM controller and a proportional output is replicated. Input 8 can only be used as a digital input, and simply activates/deactivates output 8 on the PWM controller.

MM can be configured as a Parallel to Serial Encoder, generating simply bytes of data based on the inputs.

MM can be configured as a Serial to Parallel Converter, using the inputs marked on the board as data outputs based on the incoming serial signal.

MM is 100% compatible with ALL NCD relay controllers, including the ProXR and Ultra Series, Wireless and RS-232. Every controller we manufacture is supported by the Mirror Module.

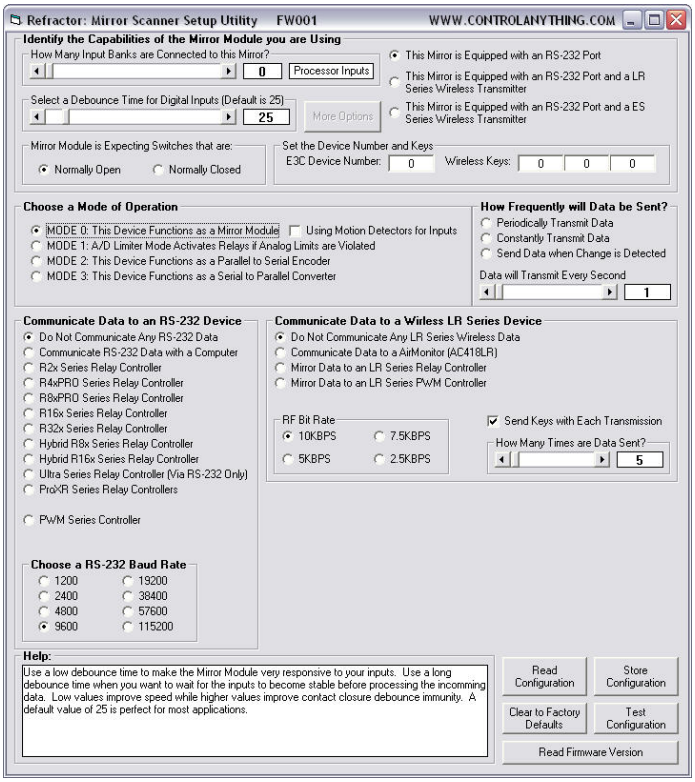
You can define WHEN data is transmitted: Periodic mode generates data every few seconds. Or, you can set data to constantly transmit from the Mirror Module. The default setting is to generate data when a change is detected. This is the most useful mode for most users because it keeps RF communications to a minimum, and only generates data when it is relevant.

Adding External Inputs to Mirror Modules with 8 Inputs:

The slider in the upper left corner of the Refractor software controls how many groups of inputs are connected to the Mirror Module. A value of 0 means 8 inputs built into the Mirror Module will be used to generate data. Increasing this slider COMPLETELY disables the on-board inputs. Instead, data will be read ONLY from the inputs connected to the Input Expansion port. If external inputs are used, Analog input options cannot be used.

Adding External Inputs to Mirror Modules with 32 Inputs:

Mirror Modules with 32 inputs cannot be configured to read data from analog inputs. These Mirror Modules cannot be used with the PWM series controllers, and they do not support Mode 1 A/D Limiter Mode. These Mirror Modules can be expanded to use more inputs if needed.



Mirror Module Debounce Time

The second slider on the Mirror Module interface sets the debounce time of the inputs. The debounce time is a timer built into the Mirror Module and it directly controls the sensitivity and reaction time of the inputs. Short values make the Mirror Module very sensitive to inputs, which can sometimes be undesirable for some applications. But if you want a faster reaction time, you can reduce the default value of 25, but at the cost of noise immunity on the inputs. Increasing this slider causes the Mirror Module to react slower to changes on the inputs. This is very useful in applications where there is a lot of noise or the potential for quick pulses. Increasing this slider is like telling the Mirror Module to make absolutely certain of the input BEFORE reacting.

Communicating to Wireless Devices

Wireless Mirror Module communications to remote NCD devices is very simple, but there is one setting that is very important. If it is set incorrectly, it will not work. You need to pay close attention to the Wireless Keys. More specifically, you need to know if the remote relay controller is set to “Require Keys” mode. If it is, you also need to know the 3 keys that will be used to communicate to the device. These settings MUST be properly programmed into the Mirror Module for successful communications. If the remote device is set to “Require Keys”, then you MUST check the “Send Keys with Each Transmission” . You must also enter the 3 keys numbers of the device you are communicating to in the three “Wireless Keys” data entry boxes at the top of the screen. If these settings are wrong, the remote device WILL NOT RESPOND and you may find yourself very frustrated with its lack of “proper” operation.

Mirror Modules

Targeting Wireless Devices
Mirror Modules broadcast wireless data to all NCD devices within radio range. If a Mirror Module is programmed to send keys with each data transmission, then any remote device with matching keys will respond to the Mirror Module. In other words, a single Mirror Module can speak to several wireless relay controllers at one time.

It is possible to have several Mirror Modules and remote relay controllers all in the same location. Each Mirror Module can be programmed to target a SPECIFIC controller using keys. Simple program each Mirror Module with a specific and matching key as a remote relay controller.

Wireless Repetitions
The final option to consider is RF repetitions. The Refractor software has a slider titled “How Many Times are Data Sent?”. This slider controls how many times data is sent from the Mirror Module to a remote device. Increasing this slider can significantly improve radio range. Decreasing this slider will decrease range, but increase overall speed of the Mirror Module. If you find the Mirror Module is not always communicating to a wireless device, then consider increasing this slider to increase the chances of reliability.

Connecting Mirror Modules to a Computer
To configure a Mirror Module or to use it with a computer, you will need a QS12 Quick Start Kit. The QS12 should be connected to the Mirror Module as shown on page 15.A.

Connecting Mirror Modules to NCD Devices
If you are using a Mirror Module in conjunction with an NCD device, you will only need to connect two wires. RS-232 Data and RS-232 Ground. The RS-232 Ground of the Mirror Module connects to the RS-232 Ground of the remote device. The RS-232 Data Output of the Mirror Module is connected to the RS-232 Data Input of the RS-232 device.

IMPORTANT:
IF YOU ARE USING THE MIRROR MODULE IN CONJUNCTION WITH A R4xPRO or R8xPRO SERIES CONTROLLERS: or any relay controller that is optoisolated, you MUST set the PC/MAC Jumper on these devices to the MAC position for compatibility with the Mirror Module.

INPUTS:
The inputs on the Mirror Module are designed to work with contact closures such as switches and buttons. Inputs are pulled high (when left open) and are connected to ground to “activate” the input. You can control whether the Mirror Module is expecting Normally Closed or Normally Open inputs using the Refractor software. By default, inputs are configured as Normally Open. When used in Analog mode, the Mirror Module will expect an input voltage from 0-5VDC. You should NEVER exceed this input voltage under any circumstances or damage will result.

In serial to parallel converter mode, the Mirror Module will use the inputs as outputs, delivering 0/5V logic that is TTL/CMOS compatible. All inputs are diode clamped as a feature of the CPU.

For additional information on the settings in the Refractor software, please move the mouse button over the different interface options. The Help box at the bottom of the Refractor software will change as you move the pointer, helping you better understand the options and features of the Mirror Module.

Should you have any questions, please feel free to contact as. Please reference the contact page on our web site at www.iorelay.com for latest contact information.