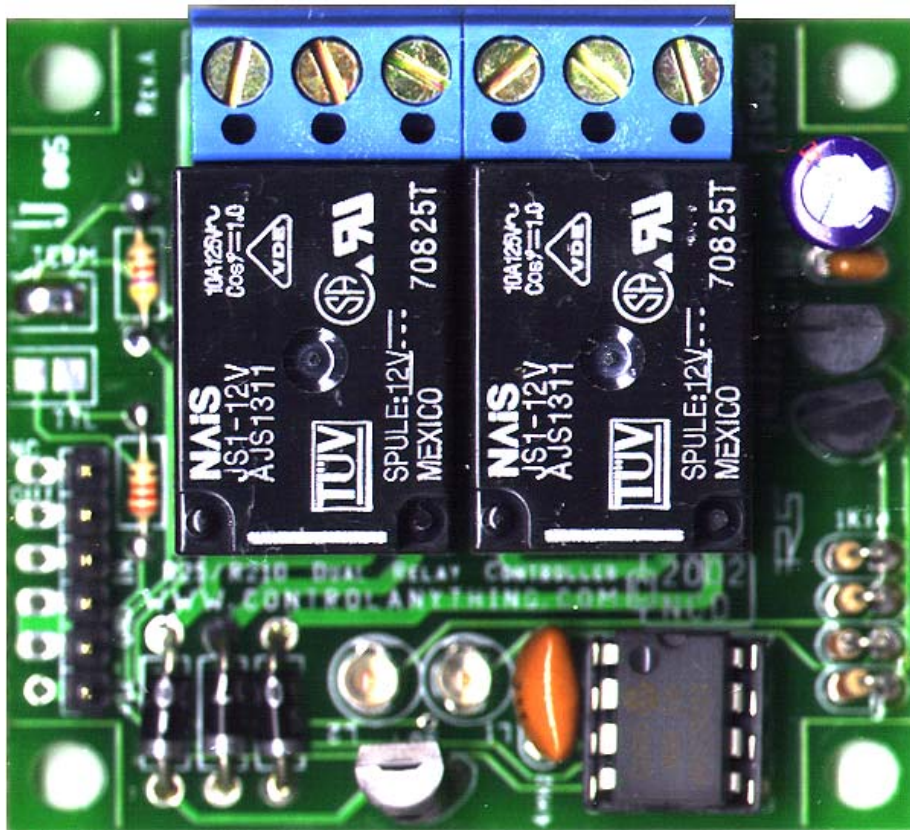


Wireless / RS-232 Single / Dual Relay Controllers

NCD RS-232 / ZigBee Networkable Relay Controllers



Programmable Single/Dual Relay Controller Devices

Device Features

- RS-232 Interface (9600 Baud)
- ZigBee Wireless Interface (115.2K Baud)
- Single / Dual Channel Configuration
- Available in SPST / SPDT / DPDT Relay Configurations
- LED Relay Status LEDs
- On-Board Terminal Blocks
- Easy Programming using Simple Commands
- Supports 2-Way Communications
- RS-232 Versions Support E3C Command Set for Controlling Multiple Devices
- ZigBee Versions Support Point to Multi-Point Networking

Warranty

NCD Warrants its products against defects in materials and workmanship for a period of 90 days. 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, using the same shipping method used to ship the product to NCD.

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

Copyright 2000-2008 by NCD. All rights reserved. Other brand and product names are trademarks of registered trademarks of their respective holders.

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 e-mailed to Ryan Sheldon at ryans@controlanything.com. Technical questions submitted via e-mail are answered up to 20 times daily. 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:

ryans@controlanything.com
www.controlanything.com
www.controleverything.com
www.iorelay.com

Introduction to the NCD Single/Dual Channel Relay Controllers

NCD Single and Dual Channel Relay Controllers were designed to service the needs of computer controlled switching applications using RS-232 direct wired communications or ZigBee Wireless Communications. Available in SPST, SPDT, and DPDT Relay Configurations, NCD offers the most versatile line of low-cost relay controllers in the world.

RS-232 Versions:

The RS-232 version requires no-more than 25ma to power the microprocessor. Each relay that is activated will impose no more than 80ma additional current draw. RS-232 versions can draw up to 185ma MAX when both relays are activated. Actual current may vary depending on model, but will not exceed these current requirements. Standard operating voltage is 12VDC. Voltage should not exceed automotive application voltages (13.8VDC). Low voltages (below 10VDC) may allow CPU to function, but the on-board relays may not activate.

RS-232 versions are compatible with IBM/PC systems with a compatible RS-232 Serial Port, USB to Serial Adapters, and Embedded microcontrollers with a TTL/CMOS communication lines. This device should not be used with Macintosh computers and should not be used with UART/USART lines of a microprocessor without inversion of output signals (use a MAX202 or equivalent RS-232 line driver or inverter circuit to make this device compatible with UART/USART data lines). RS-232 versions are available in a low cost direct wired configuration or with an integrated RS-232 port. The integrated RS-232 port version requires a serial extension cable, Male to Female, and a 12VDC power supply with a 2.1mm Center Positive Barrel Connector. Integrated LEDs will light when a relay is activated. The RS-232 series controllers should not be powered without a connection to a data source (relays may randomly activate).

RS-232 versions operate exclusively at 9600 Baud (8 Data bits, 1 Stop Bit, No Parity, No Flow Control), and include a E3C command set, which allows a single computer to talk to multiple relay controllers.

SignalSwitch Compatibility

All RS-232 Versions of this device are compatible with www.signalswitch.com, which allows you to control these devices from anywhere in the world using a free account with a user-customized interface. A dedicated computer is required to process commands from www.signalswitch.com. The computer is used to communicate data from www.signalswitch.com to the relay controller using a standard COM port. Listener software (provided by www.signalswitch.com) serves as the software link between the device and www.signalswitch.com. Listener software MUST BE CONFIGURED for 9600 Baud for proper operation.

ZigBee Versions:

The ZigBee version requires no-more than 250ma to power the microprocessor and wireless components. Each relay that is activated will impose no more than 80ma additional current draw. ZigBee versions can draw up to 410ma MAX when both relays are activated. Actual current may vary depending on model, but will not exceed these current requirements. Voltage should not exceed automotive application voltages (13.8VDC). Low voltages (below 10VDC) may allow CPU to function, but the on-board relays may not activate.

A 12VDC power supply with a 2.1mm Center Positive Barrel Connector is required to power this device. Integrated LEDs will light when a relay is activated.

ZigBee versions require a [ZigBee Modem](#). This wireless modem mounts as a COM port on your computer, making it act like a real serial port. This ensures easy software development. Communications to the ZigBee Modem are fixed at 115.2K Baud, 8 Data Bits, 1 Stop Bit, No Parity, and No Flow Control. ZigBee devices respond with ASCII Character Code 85 to acknowledge the receipt of all wireless commands. The E3C command set found in this manual applies to RS-232 devices ONLY. These commands have no function on ZigBee devices.

ZigBee Wireless Range

Depending on the version purchased, supported wireless communication range can be up to 300 feet (max 1mw versions) or up to 1 Mile (max 60mw versions). Actual range will depend on terrain, obstructions, electrical noise, and relative humidity.

ZigBee Networking

It is possible to communicate to multiple ZigBee relay controllers using the ZigBee communications protocol. The ZigBee protocol is extensive, and should be researched independently of this manual, as this manual only focuses on commands to talk to ZigBee relay devices. ZigBee networking is accomplished using standard ZigBee commands. The ZigBee command set is an international standard, and was not developed by National Control Devices, and will not be detailed in this product manual. [more info...](#)

ZigBee Mesh Networking

We anticipate supporting ZigBee Mesh Networking late 2008 or early 2009.

SignalSwitch Compatibility

All ZigBee Wireless Versions of this device are compatible with www.signalswitch.com, which allows you to control these devices from anywhere in the world using a free account with a user-customized interface. A dedicated computer is required to process commands from www.signalswitch.com. The computer is used to communicate data from www.signalswitch.com to the relay controller using a standard COM port. Listener software (provided by www.signalswitch.com) serves as the software link between the device and www.signalswitch.com. Listener software MUST BE CONFIGURED for 115.2K Baud for proper operation.

Coming in 2008:

NCD will offer a Ethernet to ZigBee Converter, as well as a 802.11b to ZigBee Converter. These devices will allow wireless control from www.signalswitch.com without a computer.

IMPORTANT POWER SUPPLY REQUIREMENTS

- 1) DO NOT USE A WALL WART TYPE UNREGULATED POWER SUPPLY.
- 2) USE ONLY A COMPUTER GRADE REGULATED SUPPLY RATED AT 12 VOLTS DC, 200ma OR GREATER.
- 3) USE A SUPPLY RATED FOR MORE AMPERAGE WHEN POWERING MULTIPLE BOARDS.
- 4) THIS DEVICE IS COMPATIBLE WITH AUTOMOTIVE ELECTRICAL SYSTEMS.
- 5) DC POWER SHOULD NEVER TRAVEL GREATER THAN 20 FEET. A SEPARATE POWER SUPPLY SHOULD BE USED FOR EACH CONTROLLER IF CONTROLLERS ARE NOT LOCATED WITHIN 20 FEET OF EACH OTHER.

NOTE:

WE HIGHLY RECOMMEND THE USE OF OUR QS12-F6 QUICK-START KIT.

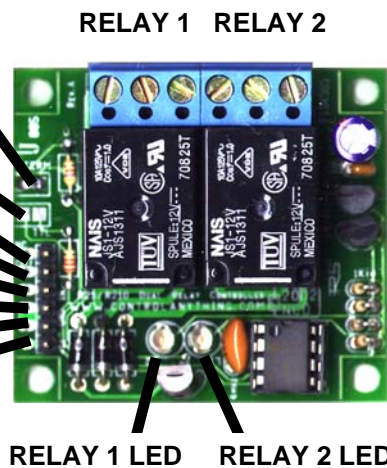
TTL Solder Pad:

When these solder pads are bridged, the R2 is configured to accept RS-232 data at TTL logic levels. These solder pads should be bridged when using a Basic Stamp. Leave these solder pads un-bridged when using a computer.

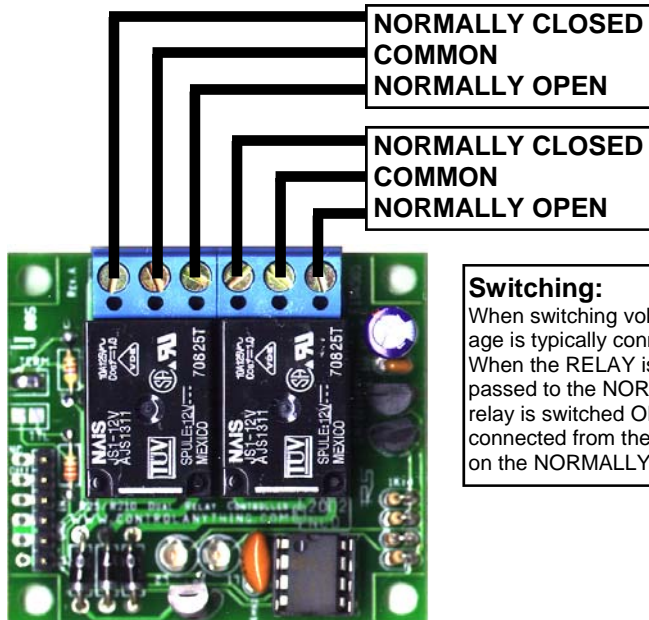
TERM Solder Pad:

This solder pad terminates the RS-232 Data Output. These Solder Pads **MUST** be Bridged with Solder when using a single R2 connected to your serial port. If using multiple R2 boards attached to the same serial port, bridge these pads with solder on the R2 that is wired closest to your computer. Leave these pads un-bridged on all other boards.

No Connection
RS-232 Output
RS-232 Input
RS-232 Ground
Supply Ground
+12 VDC
Regulated Supply



BAUD RATE IS FIXED AT 9600 BAUD.



Switching:

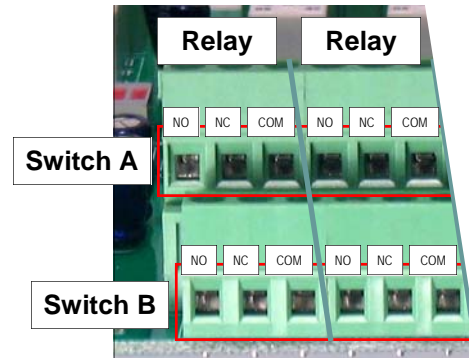
When switching voltages using the R2, a user-supplied voltage is typically connected to the COMMON post of the relay. When the RELAY is OFF, the voltage on the COMMON is passed to the NORMALLY CLOSED position. When the relay is switched ON, the voltage on the COMMON is disconnected from the NORMALLY CLOSED and reconnected on the NORMALLY OPEN terminal.

Connecting 1-Amp DPDT Relays (White Relays)

Connector Pinouts for 1-Amp DPDT Series Controllers

Any time you see one of our relay controllers with a green 12-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.

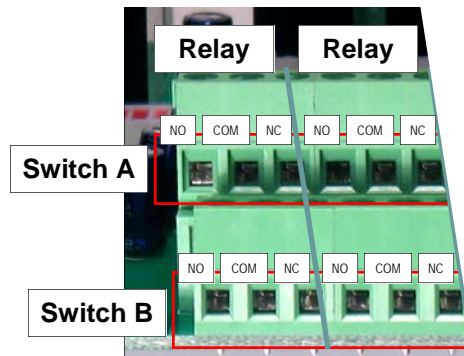
NC: Normally Close
This terminal is connected to COM when the relay is off.
This terminal has no connection with the relay is on.

Connecting 3-Amp/5-Amp DPDT Relays (Black Relays)

Connector Pinouts for 3-Amp and 5-Amp DPDT Series Controllers

Any time you see one of our relay controllers with a green 12-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.

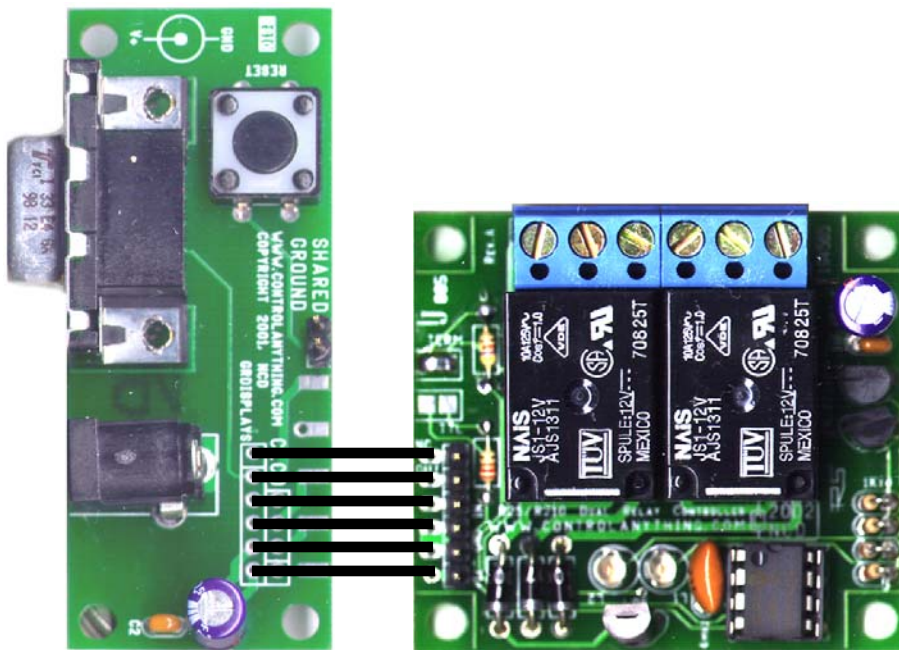
NC: Normally Close
This terminal is connected to COM when the relay is off.
This terminal has no connection with the relay is on.

IMPORTANT POWER SUPPLY REQUIREMENTS

- 1) DO NOT USE A WALL WART TYPE UNREGULATED POWER SUPPLY.
- 2) USE ONLY A COMPUTER GRADE REGULATED SWITCHER SUPPLY RATED AT 12 VOLTS DC, 200 ma OR GREATER.
- 3) USE A SUPPLY RATED FOR MORE AMPERAGE WHEN POWERING MULTIPLE BOARDS.
- 4) THIS DEVICE IS COMPATIBLE WITH AUTOMOTIVE ELECTRICAL SYSTEMS.
- 5) DC POWER SHOULD NEVER TRAVEL GREATER THAN 20 FEET. A SEPARATE POWER SUPPLY SHOULD BE USED FOR EACH CONTROLLER IF CONTROLLERS ARE NOT LOCATED WITHIN 20 FEET OF EACH OTHER.

NOTE:

WE HIGHLY RECOMMEND THE USE OF OUR QS12-F6 QUICK-START KIT SHOWN BELOW.



Above: R25 connected to the RSIO (included with the Quick Start 12 Kit (QS12-F6)).

Networking Multiple R2 Relay Controllers

The R2 supports our E3C command set, allowing you to control up to 256 NCD devices using a single RS-232 serial port. To control multiple R2 controllers from a single serial port, follow these simple steps:

PROGRAMMING:

- 1) Connect a single R2 to the serial port of your computer.
- 2) Program the R2 with a unique device number.
- 3) Label the R2 controller with its device number.
- 4) Repeat these steps until ALL R2 controllers are programmed with a unique device number and are clearly labeled.

WIRING:

Once all devices have been individually programmed with a unique device number, you are ready to connect multiple R2 controllers on a single serial port. Connect Power and Serial wires to EVERY R2 controller on the network. The quickest way to do this is to use our Quick Start 12 kit (QS12-F6) to provide serial and power connections to the first board. Next, additional flat-ribbon cable can be used to connect from the first R2 to the next R2 in the chain. Keep running flex cable in parallel with the first board until all R2 controllers are connected. Once connected, bridge the TERM solder pad with solder on the R2 board that is physically located closest to your computer. This will terminate RS-232 communications coming from the R2 back to your computer.

CONTROLLING:

Once all board are connected as described above and powered up, all R2 controllers will respond to your commands. So if you were to issue a command for switching, all R2 controllers will respond to you switch command. Simply use the E3C command set found on Page 7 to control which devices you would like to speak to using the E3C device number you have programmed into the controllers. The E3C command set supports commands for controlling all boards at once, each board individually (command 252, the most popular choice), or you can program some devices to listen and other devices to ignore your commands.

Sending Commands to the R2

The R2 is capable of receiving data via RS-232 serial communications and is compatible with just about any computer or microcontroller ever produced, including the Macintosh, Amiga, Basic Stamp, and of course, Windows & DOS based machines.

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 this device. Commands should be sent using ASCII character codes 0-255 rather than ASCII characters (A, B, C, 1, 2, 3, 4, etc.). See "ASCII Codes vs. Characters" on this page. Note that ASCII character codes are used for commanding the CLCD while actual ASCII characters are used to send text to the display.

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

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.

For your convenience, we have provided programming examples for Visual Basic 6. The provided examples should greatly speed development time. You may want to visit www.controlanything.com for the latest software and programming examples.

Programming examples for this device 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.

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 such as Notepad or WordPad.

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.

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.

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.

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. This device supports the full set of E3C commands.

Note: E3C Commands Do Not Work with ZigBee Wireless Devices. ZigBee Networking Uses ZigBee Commands.
[more info...](#)

How does E3C Work?

First of all, each device must be assigned a device number from 0 to 255. The device number may be programmed using command 255, described below, using our Visual Basic 6 example code, or using our Runtime .EXE programs.

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

247 Read E3C Device Number:

Reports back the pre-programmed E3C device number of the R2. This command will return a value of 0-255.

248 Enable All Devices:

Tells all devices to respond to your commands.

249 Disable All Devices:

Tells all devices to ignore your commands.

250 Enable a Selected Device:

Tells a specific device to listen to your commands.

251 Disable Selected Device:

Tells a specific device to ignore your commands.

252 Enable Selected Device Only:

Tells a specific device to listen to your commands, all other devices will ignore your commands.

253 Disable a Selected Device Only:

Tells a specific device to ignore your commands, all others will listen.

255 Store E3C Device Number:

Programs an E3C device number into the controller. This command requires a parameter (0-255), indicating the device number. DO NOT USE THIS COMMAND WHEN MORE THAN ONE DEVICE IS ATTACHED TO YOUR SERIAL PORT OR ALL DEVICES WILL BE PROGRAMMED WITH THE SAME DEVICE NUMBER.

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.

Sample Code: The E3C Command Set

```
Public Sub GetE3CDeviceNumber()  
    'Enable All E3C Devices  
    MSComm1.Output = Chr$(254)      'Enter Command Mode  
    MSComm1.Output = Chr$(247)      'Read Device Number Command  
    Do                               'Wait for Device to Reply  
        DoEvents                    'Allow Windows to MultiTask  
    Until MSComm1.InBufferCount > 0 'If the Device Replies  
    GetDeviceNumber = Asc(MSComm1.Input) 'Get Device Number from Buffer  
End Sub  
  
Public Sub EnableAllDevices()  
    'Enable All E3C Devices  
    MSComm1.Output = Chr$(254)      'Enter Command Mode  
    MSComm1.Output = Chr$(248)      'E3C Enable All Device Command  
End Sub  
  
Public Sub DisableAllDevices()  
    'Disable All E3C Devices  
    MSComm1.Output = Chr$(254)      'Enter Command Mode  
    MSComm1.Output = Chr$(249)      'E3C Disable All Device Command  
End Sub  
  
Public Sub EnableSpecificDevice(Device)  
    '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  
End Sub  
  
Public Sub DisableSpecificDevice(Device)  
    '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  
End Sub  
  
Public Sub DisableAllDevicesExcept(Device)  
    '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  
End Sub  
  
Public Sub EnableAllDevicesExcept(Device)  
    '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  
End Sub  
  
Public Sub StoreDeviceNumber(Device)  
    'Store an E3C Device Number into the Controller  
    MSComm1.Output = Chr$(254)      'Enter Command Mode  
    MSComm1.Output = Chr$(255)      'E3C Store Device Number Command  
    MSComm1.Output = Chr$(Device)  'Device Number that will be Stored  
End Sub
```


Using RS-232 Single/Dual Relay Controllers

NOTICE: Send All Commands to RS-232 Single/Dual Relay Controllers at 9600 Baud, 8 Data Bits, 1 Stop Bit, No Parity, No Flow Control.

254, 0 - Turn Off Relay 1

254, 1 - Turn On Relay 1

254, 2 - Turn Off Relay 2

254, 3 - Turn On Relay 2

254, 4 - Get Relay 1 Status

This Command reports back 0 or 1 indicating the status of Relay 1. 0 = Off, 1 = On.

254, 5 - Get Relay 2 Status

This Command reports back 0 or 1 indicating the status of Relay 2. 0 = Off, 1 = On.

254, 6 - Set Status of Both Relays

This command sets the status of both relays at one time. This command requires a parameter of 0-3:

Parameter 0: Turn Off Both Relays

Parameter 1: Turn On Relay 1, Turn Off Relay 2

Parameter 2: Turn On Relay 2, Turn Off Relay 1

Parameter 3: Turn On Both Relays.

254, 7 - Get Status of Both Relays

This command reports the on/off status of both relays. This command will send a byte from 0 to 3 back to the user indicating the status of both relays:

Return Byte 0: Both Relays are Off

Return Byte 1: Relay 1 is On, Relay 2 is Off

Return Byte 2: Relay 2 is On, Relay 1 is Off

Return Byte 3: Both Relays are On

254, 8 - Store Relay Status as Power-Up Default

This command will store the current on/off state of the relays in non-volatile EEPROM. The next time power is applied, the relays will automatically return to the store on/off state.

254, 9 - Retrieve Startup Default State of the Relays

This command retrieves the stored power-up default state of the relays. This command will send a byte from 0-3 indicating the on/off state of the relays when power is first applied to the board.

Return Byte 0: Both Relays are Off on Power-up

Return Byte 1: Relay 1 is On, Relay 2 is Off on Power-up

Return Byte 2: Relay 2 is On, Relay 1 is Off on Power-up

Return Byte 3: Both Relays are On when Powered-up

Sample Code: Relay Command Set

```
Public Sub RelayOneOff()  
    MSCOMM1.Output = Chr$(254)  
    MSCOMM1.Output = Chr$(0)  
End Sub  
Public Sub RelayOneOn()  
    MSCOMM1.Output = Chr$(254)  
    MSCOMM1.Output = Chr$(1)  
End Sub  
Public Sub RelayTwoOff()  
    MSCOMM1.Output = Chr$(254)  
    MSCOMM1.Output = Chr$(2)  
End Sub  
Public Sub RelayTwoOn()  
    MSCOMM1.Output = Chr$(254)  
    MSCOMM1.Output = Chr$(3)  
End Sub  
Public Function RelayOneStatus()  
    MSCOMM1.Output = Chr$(254)  
    MSCOMM1.Output = Chr$(4)  
    Do  
        DoEvents  
        Until MSCOMM1.InBufferCount > 0  
    RelayOneStatus = Asc(MSCOMM1.Input)  
End Sub  
Public Function RelayTwoStatus()  
    MSCOMM1.Output = Chr$(254)  
    MSCOMM1.Output = Chr$(5)  
    Do  
        DoEvents  
        Until MSCOMM1.InBufferCount > 0  
    RelayTwoStatus = Asc(MSCOMM1.Input)  
End Sub  
Public Sub SetBothRelays(Status)  
    MSCOMM1.Output = Chr$(254)  
    MSCOMM1.Output = Chr$(6)  
    MSCOMM1.Output = Chr$(Status)  
End Sub  
Public Function GetBothRelayStatus()  
    MSCOMM1.Output = Chr$(254)  
    MSCOMM1.Output = Chr$(7)  
    Do  
        DoEvents  
        Until MSCOMM1.InBufferCount > 0  
    GetBothRelayStatus = Asc(MSCOMM1.Input)  
End Sub  
Public Sub StorePowerupDefault()  
    MSCOMM1.Output = Chr$(254)  
    MSCOMM1.Output = Chr$(8)  
End Sub  
Public Function GetStartupDefault()  
    MSCOMM1.Output = Chr$(254)  
    MSCOMM1.Output = Chr$(9)  
    Do  
        DoEvents  
        Until MSCOMM1.InBufferCount > 0  
    GetStartupDefault = Asc(MSCOMM1.Input)  
End Sub  
'Enter Command Mode  
'Turn Off Relay 1  
'Enter Command Mode  
'Turn On Relay 1  
'Enter Command Mode  
'Turn Off Relay 2  
'Enter Command Mode  
'Turn On Relay 2  
'Enter Command Mode  
'Ask for Status of Relay 1  
'Wait for Device to Reply  
'Allow Windows to MultiTask  
'If the Device Replies  
'Get Relay 1 Status  
'Enter Command Mode  
'Ask for Status of Relay 1  
'Wait for Device to Reply  
'Allow Windows to MultiTask  
'If the Device Replies  
'Get Relay 2 Status  
'Enter Command Mode  
'Set Both Relays Command  
'Set Status of Both Relays  
Enter Command Mode  
'Ask for Status of Both Relays  
'Wait for Device to Reply  
'Allow Windows to MultiTask  
'If the Device Replies  
'Get Status of Both Relays  
'Enter Command Mode  
'Store Power-Up Default  
'Enter Command Mode  
'Ask for Startup Relay Status  
'Wait for Device to Reply  
'Allow Windows to MultiTask  
'If the Device Replies  
'Get Startup Default
```

Using ZigBee Single/Dual Relay Controllers

NOTICE: Commands Sent to ZigBee Wireless Devices are Communicated to the Remote ZigBee Device using a ZigBee Modem. The ZigBee Modem Mounts as a COM Port on your Computer. Send All Commands to the ZigBee Modem at 115.2K Baud, 8 Data Bits, 1 Stop Bit, No Parity, No Flow Control.

254, 0 - Turn Off Relay 1

This Command will return ASCII Character Code 85 to Confirm the Receipt of this Command.

254, 1 - Turn On Relay 1

This Command will return ASCII Character Code 85 to Confirm the Receipt of this Command.

254, 2 - Turn Off Relay 2

This Command will return ASCII Character Code 85 to Confirm the Receipt of this Command.

254, 3 - Turn On Relay 2

This Command will return ASCII Character Code 85 to Confirm the Receipt of this Command.

254, 4 - Get Relay 1 Status

This Command reports back 0 or 1 indicating the status of Relay 1. 0 = Off, 1 = On.

254, 5 - Get Relay 2 Status

This Command reports back 0 or 1 indicating the status of Relay 2. 0 = Off, 1 = On.

254, 6 - Set Status of Both Relays

This command sets the status of both relays at one time. This command requires a parameter of 0-3:

Parameter 0: Turn Off Both Relays

Parameter 1: Turn On Relay 1, Turn Off Relay 2

Parameter 2: Turn On Relay 2, Turn Off Relay 1

Parameter 3: Turn On Both Relays.

This Command will return ASCII Character Code 85 to Confirm the Receipt of this Command.

254, 7 - Get Status of Both Relays

This command reports the on/off status of both relays. This command will send a byte from 0 to 3 back to the user indicating the status of both relays:

Return Byte 0: Both Relays are Off

Return Byte 1: Relay 1 is On, Relay 2 is Off

Return Byte 2: Relay 2 is On, Relay 1 is Off

Return Byte 3: Both Relays are On

254, 8 - Store Relay Status as Power-Up Default

This command will store the current on/off state of the relays in non-volatile EEPROM. The next time power is applied, the relays will automatically return to the store on/off state.

This Command will return ASCII Character Code 85 to Confirm the Receipt of this Command.

254, 9 - Retrieve Startup Default State of the Relays

This command retrieves the stored power-up default state of the relays. This command will send a byte from 0-3 indicating the on/off state of the relays when power is first applied to the board.

Return Byte 0: Both Relays are Off on Power-up

Return Byte 1: Relay 1 is On, Relay 2 is Off on Power-up

Return Byte 2: Relay 2 is On, Relay 1 is Off on Power-up

Return Byte 3: Both Relays are On when Powered-up

Sample Code: Relay Command Set

```
Public Sub RelayOneOff()  
    MSCOMM1.Output = Chr$(254)  
    MSCOMM1.Output = Chr$(0)  
    GetData  
End Sub  
'Enter Command Mode  
'Turn Off Relay 1  
'Wait for Device to Reply  
  
Public Sub RelayOneOn()  
    MSCOMM1.Output = Chr$(254)  
    MSCOMM1.Output = Chr$(1)  
    GetData  
End Sub  
'Enter Command Mode  
'Turn On Relay 1  
'Wait for Device to Reply  
  
Public Sub RelayTwoOff()  
    MSCOMM1.Output = Chr$(254)  
    MSCOMM1.Output = Chr$(2)  
    GetData  
End Sub  
'Enter Command Mode  
'Turn Off Relay 2  
'Wait for Device to Reply  
  
Public Sub RelayTwoOn()  
    MSCOMM1.Output = Chr$(254)  
    MSCOMM1.Output = Chr$(3)  
    GetData  
End Sub  
'Enter Command Mode  
'Turn On Relay 2  
'Wait for Device to Reply  
  
Public Function RelayOneStatus()  
    MSCOMM1.Output = Chr$(254)  
    MSCOMM1.Output = Chr$(4)  
    Do  
        DoEvents  
        Until MSCOMM1.InBufferCount > 0  
        RelayOneStatus = Asc(MSCOMM1.Input)  
    End Sub  
'Enter Command Mode  
'Ask for Status of Relay 1  
'Wait for Device to Reply  
'Allow Windows to Multitask  
'If the Device Replies  
'Get Relay 1 Status  
  
Public Function RelayTwoStatus()  
    MSCOMM1.Output = Chr$(254)  
    MSCOMM1.Output = Chr$(5)  
    Do  
        DoEvents  
        Until MSCOMM1.InBufferCount > 0  
        RelayTwoStatus = Asc(MSCOMM1.Input)  
    End Sub  
'Enter Command Mode  
'Ask for Status of Relay 1  
'Wait for Device to Reply  
'Allow Windows to Multitask  
'If the Device Replies  
'Get Relay 2 Status  
  
Public Sub SetBothRelays(Status)  
    MSCOMM1.Output = Chr$(254)  
    MSCOMM1.Output = Chr$(6)  
    MSCOMM1.Output = Chr$(Status)  
    GetData  
End Sub  
'Enter Command Mode  
'Set Both Relays Command  
'Set Status of Both Relays  
'Wait for Device to Reply  
  
Public Function GetBothRelayStatus()  
    MSCOMM1.Output = Chr$(254)  
    MSCOMM1.Output = Chr$(7)  
    Do  
        DoEvents  
        Until MSCOMM1.InBufferCount > 0  
        GetBothRelayStatus = Asc(MSCOMM1.Input)  
    End Sub  
'Enter Command Mode  
'Ask for Status of Both Relays  
'Wait for Device to Reply  
'Allow Windows to Multitask  
'If the Device Replies  
'Get Status of Both Relays  
  
Public Sub StorePowerupDefault()  
    MSCOMM1.Output = Chr$(254)  
    MSCOMM1.Output = Chr$(8)  
    GetData  
End Sub  
'Enter Command Mode  
'Store Power-Up Default  
'Wait for Device to Reply  
  
Public Function GetStartupDefault()  
    MSCOMM1.Output = Chr$(254)  
    MSCOMM1.Output = Chr$(9)  
    Do  
        DoEvents  
        Until MSCOMM1.InBufferCount > 0  
        GetStartupDefault = Asc(MSCOMM1.Input)  
    End Sub  
'Enter Command Mode  
'Ask for Startup Relay Status  
'Wait for Device to Reply  
'Allow Windows to Multitask  
'If the Device Replies  
'Get Startup Default  
  
Public Function GetData()  
    Do  
        DoEvents  
        Until MSCOMM1.InBufferCount > 0  
        GetData = Asc(MSCOMM1.Input)  
        Debug.Print GetData  
    End Sub  
'Wait for Device to Reply  
'Allow Windows to Multitask  
'If the Device Replies  
'Get Startup Default  
'Display Returned Data
```