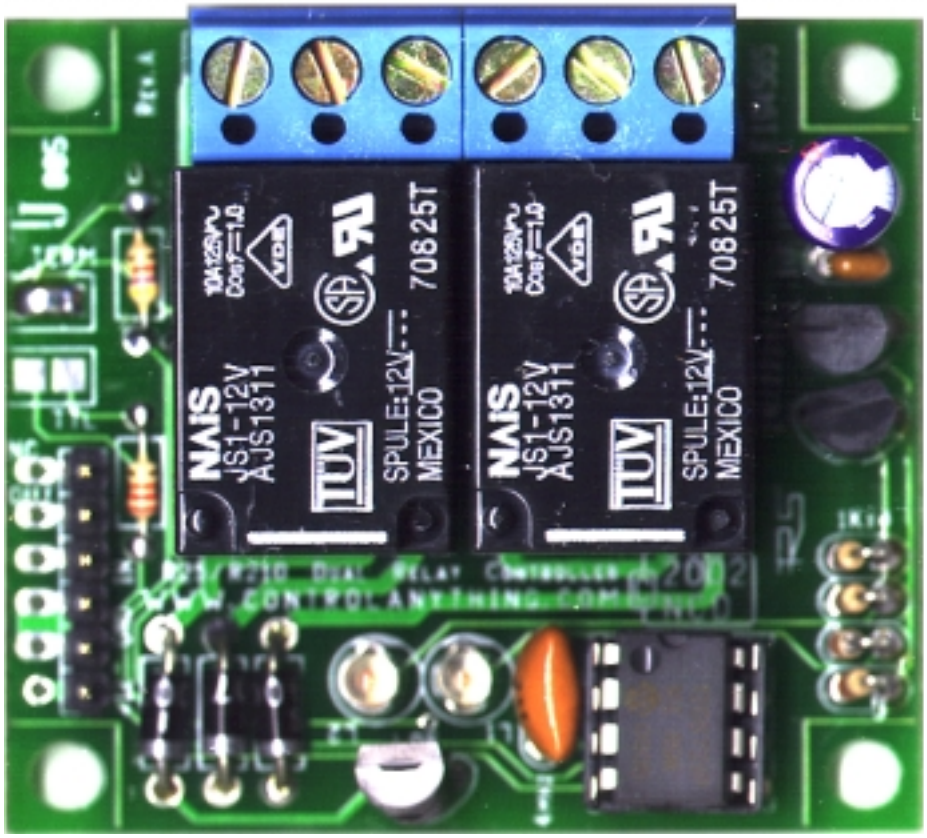
R25/R210
Dual Relay Controller
NCD RS-232 Networkable Dual Relay Controller



2” x 2” RS-232 Programmable Dual Relay Controller

Device Features

Control 256 Devices Simultaneously or Individually from a Single Serial Port
E3C Compliant Command Set with Programmable E3C Device Number
2 Relay Status LEDs
Dual Diode-Clamped Relay Driver Circuit
Small, Low-Profile Design
Reports Status of the Relays
Control Each Relay Individually or Both Relays at the Same Time
9600 Baud Operation
Programmable Power-Up Relay Activation Pattern
Query Input Selection
+12 VDC Operation
1-Way or 2-Way Communication
Just 2” x 2” Square x 1” High

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

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 ryan@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:
(417) 646-8302

Internet:
ryan@controlanything.com
www.controlanything.com
www.controleverything.com

Introduction to the R25/R210 Dual Relay Controllers

The R2 Series Dual Relay Controllers were designed to service small computer-controlled switching needs with maximum flexibility and expandability.

The R25 is equipped with 2 Aromat 5-Amp Relays capable of switching AC Loads up to 5 Amps at 240 VAC or 5 Amps at 24 VDC.

The R210 is equipped with 2 Omron 10-Amp Relays capable of switching AC Loads up to 10 Amps at 240 VAC or 10 Amps at 24 VDC.

Both types of relays are hermetically sealed to prevent moisture and dust contamination, extending the useful life of the relays.

The R2 Series supports RS-232 serial commands from any computer or microcontroller capable of sending ASCII character codes 0-255 at a 9600 baud (baud rate is fixed).

The command set of the R2 allows individual control of each relay or simultaneous control of both relays. The command set also supports 1-way or 2-way communication, allowing you to read the status of the relays, program automatic activation of one or both relays when power is first applied, and reading the power-up default status of the relays.

The R2 is fully E3C compliant, allowing up to 256 NCD devices to be shared on a single RS-232 serial port. Each device can be controlled individually or multiple devices can be controlled at once using the E3C command set. Extended E3C commands allow the user to program the device number (0-255) and read the device number from the R2 controller.

The R2 requires a regulated, computer grade +12 volt supply rated at 200 ma MAX. An optional QS12-F6 quick start kit is available for the R2, allowing quick power and data connections.

IMPORTANT POWER SUPPLY REQUIREMENTS

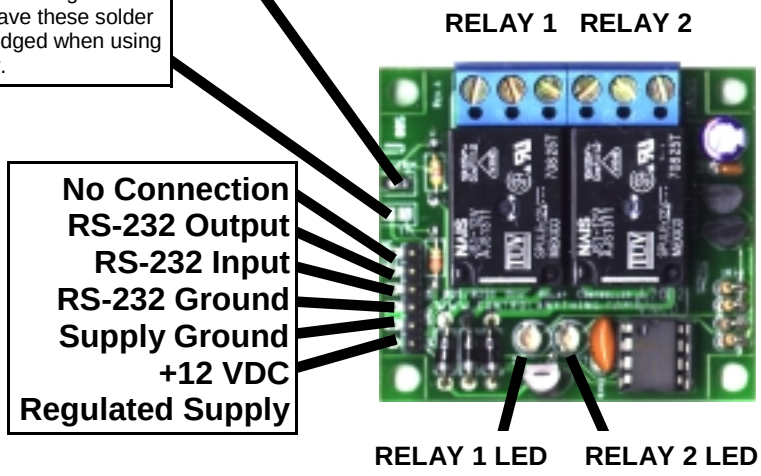
1) DO NOT USE A WALL WART TYPE UNREGULATED POWER SUPPLY.
2) USE ONLY A COMPUTER GRADE REGULATED SUPPLY RATED AT 12 VOLTS DC, 200ma OR GREATER.
3) USE A SUPPLY RATED FOR MORE AMPERAGE WHEN POWERING MULTIPLE BOARDS.
4) THIS DEVICE IS COMPATIBLE WITH AUTOMOTIVE ELECTRICAL SYSTEMS.
5) DC POWER SHOULD NEVER 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:

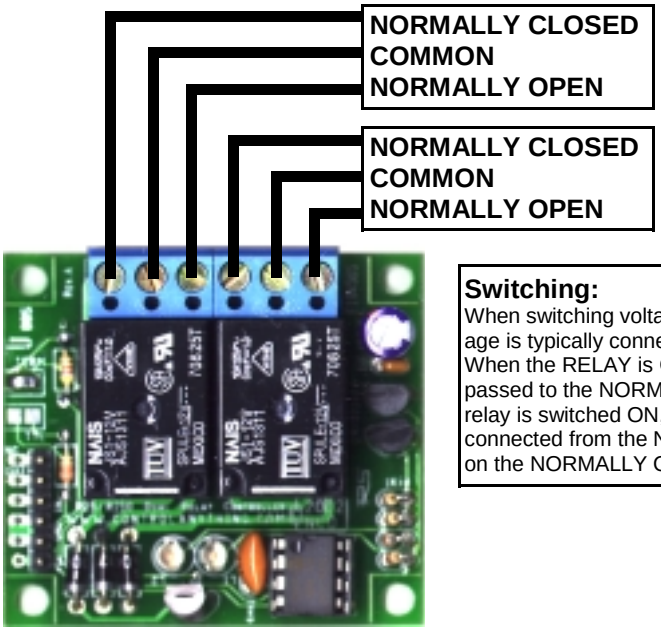
WE HIGHLY RECOMMEND THE USE OF OUR QS12-F6 QUICK-START KIT.

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.

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.



BAUD RATE IS FIXED AT 9600 BAUD.



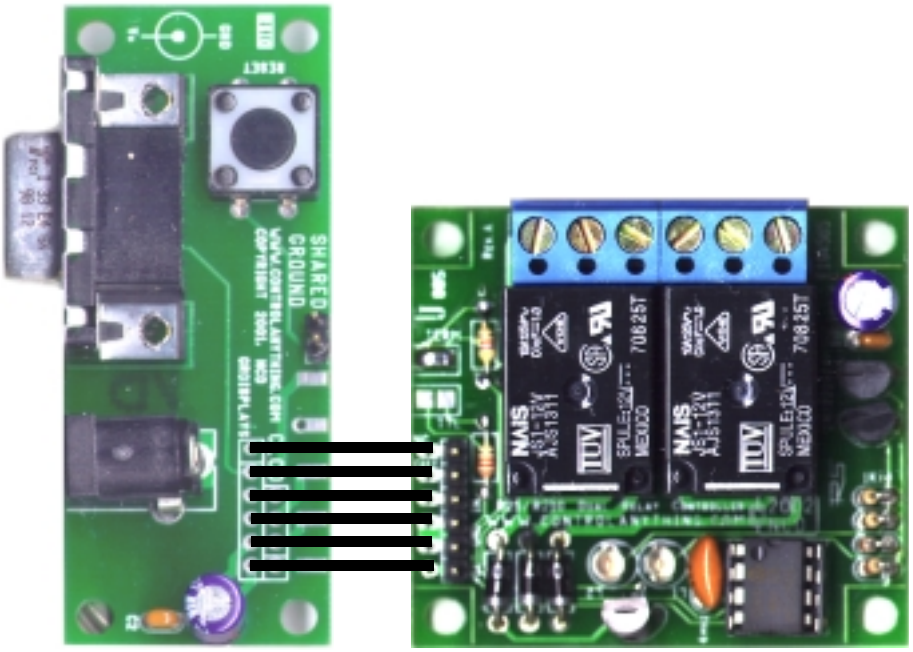
Switching:
When switching voltages using the R2, a user-supplied voltage is typically connected to the COMMON post of the relay. When the RELAY is OFF, the voltage on the COMMON is passed to the NORMALLY CLOSED position. When the relay is switched ON, the voltage on the COMMON is disconnected from the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

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 user 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 your 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.

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 into the IOServo 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 R2. This command requires a parameter (0-255), indicating the device number. DO NOT USE THIS COMMAND WHEN MORE THAN ONE R2 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 the R2 Dual Relay Controller

0 - Turn Off Relay 1

1 - Turn On Relay 1

2 - Turn Off Relay 2

3 - Turn On Relay 2

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

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

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.

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

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.

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

R2 Electrical Specifications

The R2 accepts RS-232 Level or TTL level data input at 9600 baud using 8 data bits, 1 stop bit, and no parity. The baud rate of the R2 cannot be changed. The R2 requires a +12 VDC Power Supply rated at 200 ma or more and is compatible with automotive electrical systems.

Absolute maximum input voltage is 14.5 VDC.
Absolute minimum input voltage is 10.0 VDC.

Switching Current R25: 5 Amps 240 VAC or 24 VDC.
Switching Current R210: 10 Amps 240 VAC or 24 VDC.
All Relays are Hermetically Sealed.

Our QS12-F6 quick-start kit meets all power and RS-232 communications requirements.

Sample Code: R2 Commands

```
Public Sub RelayOneOff()
    MSComm1.Output = Chr$(254)              'Enter Command Mode
    MSComm1.Output = Chr$(0)                'Turn Off Relay 1
End Sub

Public Sub RelayOneOn()
    MSComm1.Output = Chr$(254)              'Enter Command Mode
    MSComm1.Output = Chr$(1)                'Turn On Relay 1
End Sub

Public Sub RelayTwoOff()
    MSComm1.Output = Chr$(254)              'Enter Command Mode
    MSComm1.Output = Chr$(2)                'Turn Off Relay 2
End Sub

Public Sub RelayTwoOn()
    MSComm1.Output = Chr$(254)              'Enter Command Mode
    MSComm1.Output = Chr$(3)                'Turn On Relay 2
End Sub

Public Function RelayOneStatus()
    MSComm1.Output = Chr$(254)              'Enter Command Mode
    MSComm1.Output = Chr$(4)                'Ask for Status of Relay 1
    Do                                      'Wait for Device to Reply
        DoEvents                            'Allow Windows to MultiTask
    Until MSComm1.InBufferCount > 0         'If the Device Replies
    RelayOneStatus = Asc(MSComm1.Input)     'Get Relay 1 Status
End Sub

Public Function RelayTwoStatus()
    MSComm1.Output = Chr$(254)              'Enter Command Mode
    MSComm1.Output = Chr$(5)                'Ask for Status of Relay 1
    Do                                      'Wait for Device to Reply
        DoEvents                            'Allow Windows to MultiTask
    Until MSComm1.InBufferCount > 0         'If the Device Replies
    RelayTwoStatus = Asc(MSComm1.Input)     'Get Relay 2 Status
End Sub

Public Sub SetBothRelays(Status)
    MSComm1.Output = Chr$(254)              'Enter Command Mode
    MSComm1.Output = Chr$(6)                'Set Both Relays Command
    MSComm1.Output = Chr$(Status)           'Set Status of Both Relays
End Sub

Public Function GetBothRelayStatus()
    MSComm1.Output = Chr$(254)              'Enter Command Mode
    MSComm1.Output = Chr$(7)                'Ask for Status of Both Relays
    Do                                      'Wait for Device to Reply
        DoEvents                            'Allow Windows to MultiTask
    Until MSComm1.InBufferCount > 0         'If the Device Replies
    GetBothRelayStatus = Asc(MSComm1.Input) 'Get Status of Both Relays
End Sub

Public Sub StorePowerupDefault()
    MSComm1.Output = Chr$(254)              'Enter Command Mode
    MSComm1.Output = Chr$(8)                'Store Power-Up Default
End Sub

Public Function GetStartupDefault()
    MSComm1.Output = Chr$(254)              'Enter Command Mode
    MSComm1.Output = Chr$(9)                'Ask for Startup Relay Status
    Do                                      'Wait for Device to Reply
        DoEvents                            'Allow Windows to MultiTask
    Until MSComm1.InBufferCount > 0         'If the Device Replies
    GetStartupDefault = Asc(MSComm1.Input)  'Get Startup Default
End Sub
```