AirRelay / AirSwitch

Wireless Long Range Computer Controlled Relay Switching

5-Year Repair or Replace Warranty!!!

AirSwitch:

AirSwitch Software allows you to control the AirRelay Line of relay controllers via a simple graphical user interface. AirSwitch includes an installer and source code, adding universal wireless control of our wireless line of relay controllers. AirSwitch can be used to program remote devices with a unique device number, test device number settings, test various device functions, and can even help extend the communication range of the AirControl transmitter.

Wireless 1-Way Computer Controlled Relay Switching
High Speed Communication with up to 72-Bit Noise Rejection
Remotely Control an Unlimited Number of NCD Wireless Devices
Supports 2400 to 38.4K Baud Operation using the AirControl
The AirControl Communicates to the AirRelay at 2.5 to 10KBPS Data Rates
Ready and Transmit LEDs Indicate RF Data Transmission on the AirControl
Receive and Ready Status LEDs Indicate AirRelay Device Operation
XR Expansion Ports allow you to Add Banks of Relays as your Needs Grow

AirSwitch Software Requires AirControl RF Command Transmitter
AirRelay Controllers Require Any Compatible RF Command Transmitter
The AirMonitor can be used to Monitor Communications
Mechanical Drawings are ONLY Available on our Web Site

AirRelay:

Our powerful new AirRelay series relay controllers offer high-speed wireless switching with up to 72-bit noise rejection. The AirRelay lineup offers high reliability wireless computer controlled switching across distances up to 3,000 feet. AirRelay controllers can switch everything from communication signals to power signals.

XR Ports:

New for most relay controllers released in 2005 and later will be the inclusion of our XR series relay expansion ports. XR Port allow you to add banks of relays to your main relay controller.

The AirSwitch software and AirRelay controllers required the AirControl universal RF command transmitter or other NCD devices capable of transmitting AirRelay control commands.

FAST FACTS

PLEASE READ THIS SECTION IF YOU READ NOTHING ELSE!

- The Mechanical Drawing can be found on the Product Description Page for each device on our web site at www.controlanything.com.
- A Terminal Program such as HyperTerminal is NOT SUITABLE for use with this device.
- Only Two Things are Required for Wireless Computer Control: AirControl Transmitter and an AirRelay Controller.
- We HIGHLY RECOMMEND you Download and Install our FREE AirSwitch Software, used to Quickly Test the Communications to a Wireless AirRelay Controller.
- Your Computer Connects to the AirControl RF Packet Encoder/Transmitter.
- The AirControl Transmits Wireless Commands to NCD Wireless LR Series Devices via 1-Way Communications.
- The AirSwitch Program Communicates to the AirControl RF Packet Encoder/Transmitter.
- The AirSwitch Program lets you Program Remote Devices with Various Parameters, Including Device Keys, Which Allow Computer Control of Multiple Devices.
- The AirSwitch Program is Very Important for Testing. AirSwitch allows you to Familiarize yourself with Computer Controlled Wireless Switching Commands.
- The AirSwitch Program also includes an Interface for Controlling a large number of AirControl Parameters used to Increase Communication Range or Speed.
- The AirSwitch Program Should ALWAYS be Used during for Troubleshooting.
- You can control how many times Data packets are sent to your remote relay controller by sending simple commands to the AirControl.
- Increasing the number of data packets sent greatly increases communication range.
- Decreasing the number of data packets sent greatly increases speed of communications.
- The AirControl will send your computer an ASCII Character Code 85 when the AirControl has finished Transmitting your data packets.
- Your software should never attempt to transmit new data packets until the previous data packets have completed or your new data will be lost.
- Regardless of how many times you send data packets, NCD Wireless Devices only respond to one data packet, all other duplicate packets will be rejected.

5-Year Repair or Replace Warranty

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, 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

Wireless AirRelay Controller Models

This Manual Covers the Following Long Range 1-Way Wireless Relay Controllers:

- **R85W1LR** Wireless 8-Relay 5-Amp SPDT 10KBPS Long Range 1-Way Relay Controller
- **R810W1LR** Wireless 8-Relay 10-Amp SPDT 10KBPS Long Range 1-Way Relay Controller
- **R45W1LR*** Wireless 4-Relay 5-Amp SPDT 10KBPS Long Range 1-Way Relay Controller
- **R410W1LR*** Wireless 4-Relay 10-Amp SPDT 10KBPS Long Range 1-Way Relay Controller
- **R81DPDTW1LR** Wireless 8-Relay 1-Amp DPDT 10KBPS Long Range 1-Way Relay Controller
- **R83DPDTW1LR** Wireless 8-Relay 3-Amp DPDT 10KBPS Long Range 1-Way Relay Controller
- **R85DPDTW1LR** Wireless 8-Relay 5-Amp DPDT 10KBPS Long Range 1-Way Relay Controller
- **R41DPDTW1LR*** Wireless 4-Relay 1-Amp DPDT 10KBPS Long Range 1-Way Relay Controller
- **R43DPDTW1LR*** Wireless 4-Relay 3-Amp DPDT 10KBPS Long Range 1-Way Relay Controller
- **R45DPDTW1LR*** Wireless 4-Relay 5-Amp DPDT 10KBPS Long Range 1-Way Relay Controller
- **R820W1LR** Wireless 8-Relay 20-Amp SPDT 10KBPS Long Range 1-Way Relay Controller
- **R830W1LR** Wireless 8-Relay 30-Amp SPST 10KBPS Long Range 1-Way Relay Controller
- **R420W1LR** Wireless 4-Relay 20-Amp SPDT 10KBPS Long Range 1-Way Relay Controller
- **R430W1LR** Wireless 4-Relay 30-Amp SPST 10KBPS Long Range 1-Way Relay Controller
- **R420HPW1LR** Wireless 4-Relay 20-Amp High Power Low Cost SPDT 10KBPS Long Range 1-Way Relay Controller
- **R820HPW1LR** Wireless 8-Relay 20-Amp High Power Low Cost SPDT 10KBPS Long Range 1-Way Relay Controller
- **R1620HPW1LR** Wireless 16-Relay 20-Amp High Power Low Cost SPDT 10KBPS Long Range 1-Way Relay Controller
- **R165W1LR** Wireless 16-Channel 5-Amp SPDT 10KBPS Long Range 1-Way Relay Controller
- **R1610W1LR** Wireless 16-Channel 10-Amp SPDT 10KBPS Long Range 1-Way Relay Controller

* These models do NOT have an XR Port and Cannot be Used with our XR Line of Expansion Relay Controllers.

Compatible XR Expansion Relay Controllers:

- **XR165** Expansion Relay Controller with 16 5-Amp SPDT Relays
- **XR1610** Expansion Relay Controller with 16 10-Amp SPDT Relays
- **XR1620HP** Low Cost Expansion Relay Controller with 16 20-Amp High Power Relays
- Any Future NCD Device that begins with the part number XR.

Compatible Wireless Accessories:

- **AC418LR** AirControl 418MHz Long Range Command Transmitter for LR Series Controllers
- **AM418LR** AirMonitor 418MHz Long Range Wireless Traffic Monitor

Recommended Test Software:

- **AirSwitch** available for free download from our web site at www.controlanything.com

MECHANICAL DRAWINGS:

MECHANICAL DRAWINGS CAN BE FOUND ON OUR WEB SITE ON THE PRODUCT DESCRIPTION PAGE FOR EACH OF THE PRODUCTS LISTED ABOVE. ALL MECHANICAL DRAWINGS ARE IN HIGH RESOLUTION .JPG IMAGE FORMAT, AND WILL NOT FIT WITHIN THIS MANUAL.

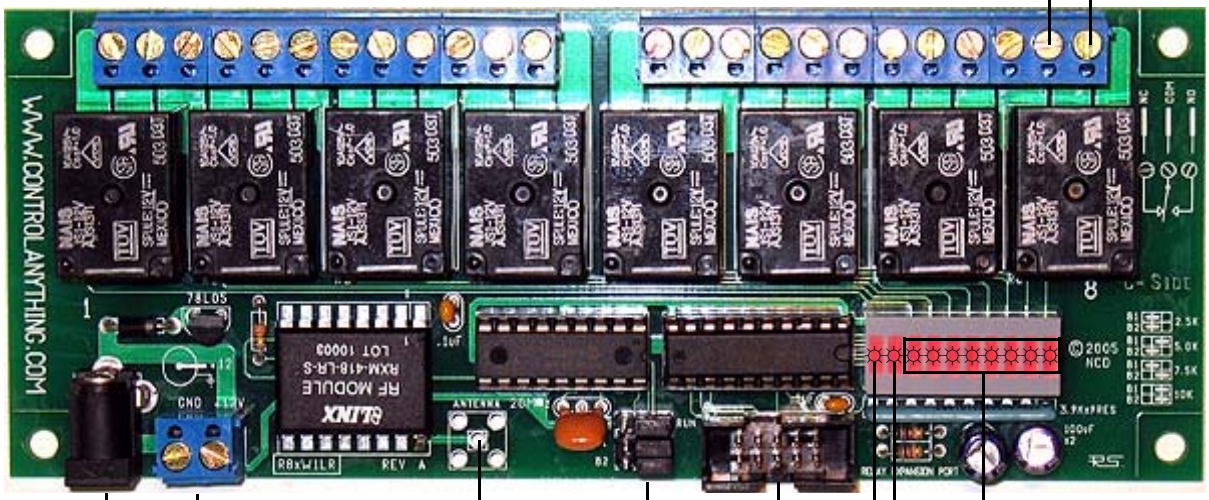
SPDT Relays have Three Connections:

Normally Open, Normally Closed, and Common
The Common is the Part of the Relay that Moves.
When a Relay is Off, Common is Connected to Normally Closed
When a Relay is On, Common is Connected to Normally Open
Most Switching Application Make Use of the Common and Normally Open Connections. Apply power to the Common. When the relay is turned On, Power Will Be Routed out the Normally Open Connection.

In the example below, a power supply is connected to a device such as a motor or a light. The other power supply line is routed into the relay controller. When the relay is activated, a connection is made between the power supply and your device, allowing for wireless computer controlled switching.

Power Supply Connection Example

Switched Output to a Motor or Light or Other Device



Direct Wire Power Connector Ground (left) +12VDC (right)

2.1MM Barrel Connector, +12VDC Female, Center Positive

Antenna Connector Pad Shown with No Antenna Attached. This Hole Pattern Fits a 90° RPSMA Connector for use ONLY with 418 MHz Antennas. Optionally, a Wire Antenna May be Soldered Into the Center Hole.

Eight Status LEDs Display the On/Off Status of Each Relay

Device Enabled and Ready LED Stays Lit when Ready to Receive Data

Data Receive/Decoded LED Flashes when Valid Data is Decoded

If your controller is equipped with an XR relay expansion port, you can add additional banks of relays to the controller by simply plugging in a relay bank into the XR port. The current firmware supports 16 banks of 8, so you can add an additional 15 banks of 8 relays to this controller, allowing you to control a total of 128 different relays.

Program/Run Jumper

When Set to Program Mode (jumpered to the left), Device Keys and Other Parameters may be stored into the controller. This device will not process any relay control functions while in program mode. Program mode is used strictly for configuring the controller from a remote location. When communicating with multiple devices, we do NOT recommend setting more than one controller into Program Mode at a time. Any live devices that are in run mode will ignore any commands used to configure a remote device. This allows you to easily add/modify devices to a wireless network without disturbing your existing installations. After configuration changes have been made, set the jumper to Run mode (jumpered to the right) and power cycle the controller. Remember, jumper changes have no effect until the controller has been power cycled.

	2.5KBPS	5KBPS	7.5KBPS	10KBPS
B1 Jumper	Left	Right	Left	Right
B2 Jumper	Left	Left	Right	Right

The Bit Rate Jumpers B1 and B2 are used to set the communication speed between the AirControl Transmitter and AirRelay Receiving Device. Lower Bit Rates can increase communication distance while higher Bit Rates significantly improve communication speed. The default setting is 10KBPS, with both B1 and B2 jumpers set to the right.

Mechanical Relays will make an audible clicking noise when switched. This is normal operation. Relays will stay ON (Latched) until a command is received to turn off the relay. Relay status will be lost if power is lost, but it is possible to program a default power-up relay status.

The memory built into this controller is write protected while in RUN mode. It is NOT possible to make significant configuration changes while in RUN mode.

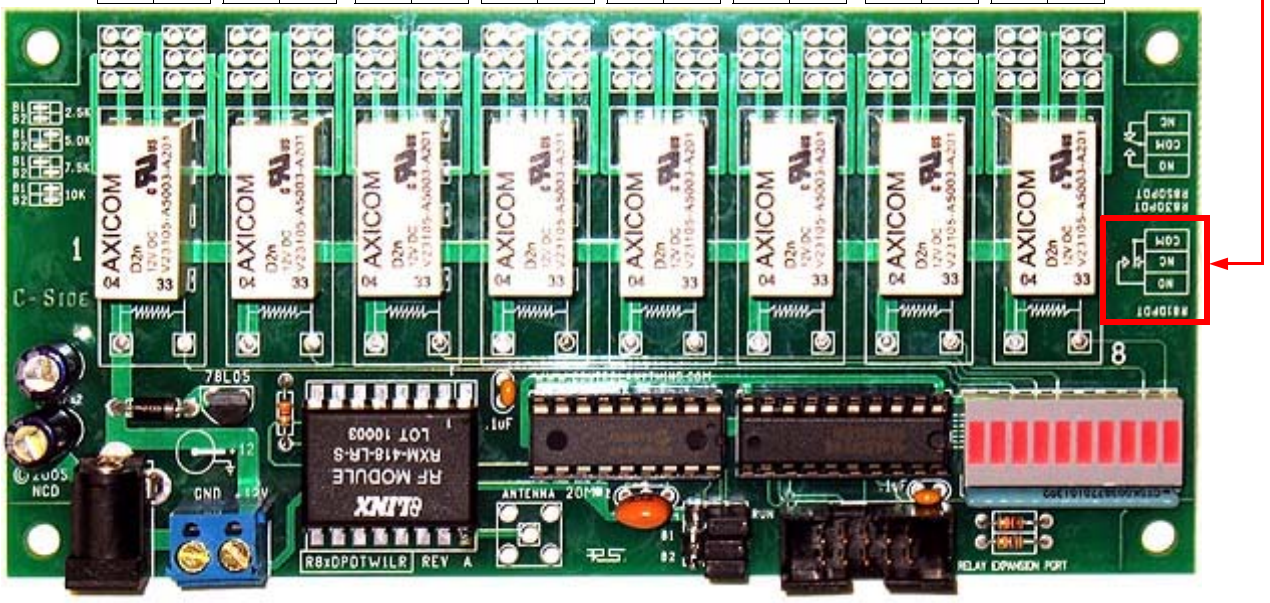
Jumper Settings are Read ONLY when Power is First applied to the board. Jumper Changes Will Not Take Effect Until the Controller has Been Power Cycled.

Remember, A Relay is Just a Switch. It does not deliver any form of output voltage other than what you put into it.

1ADPDT Relay Connections:

This Note Applies to the R41DPDT, R81DPDT, R41DPDTW1LR and R81DPDTW1LR.
Note that we have labeled some of our newer board revisions with the diagram highlighted in red. For your convenience, there are two connection points for each relay contact.

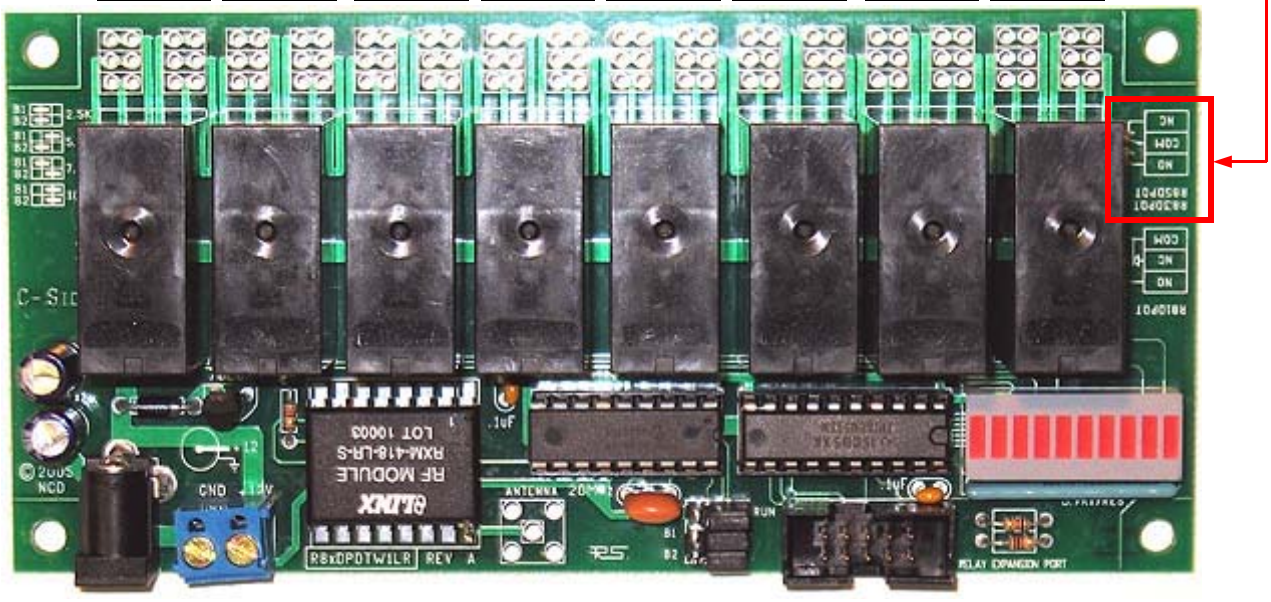
Relay 1		Relay 2		Relay 3		Relay 4		Relay 5		Relay 6		Relay 7		Relay 8	
COM	COM	COM	COM	COM	COM	COM	COM	COM	COM	COM	COM	COM	COM	COM	COM
NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC
NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO



3DPDT/5DPDT Relay Connections:

This Note Applies to the R43DPDT, R45DPDT, R83DPDT, R85DPDT, R43DPDTW1LR, R45DPDTW1LR, R83DPDTW1LR, and R85DPDTW1LR.
Note that we have labeled some of our newer board revisions with the diagram highlighted in red. For your convenience, there are two connection points for each relay contact.

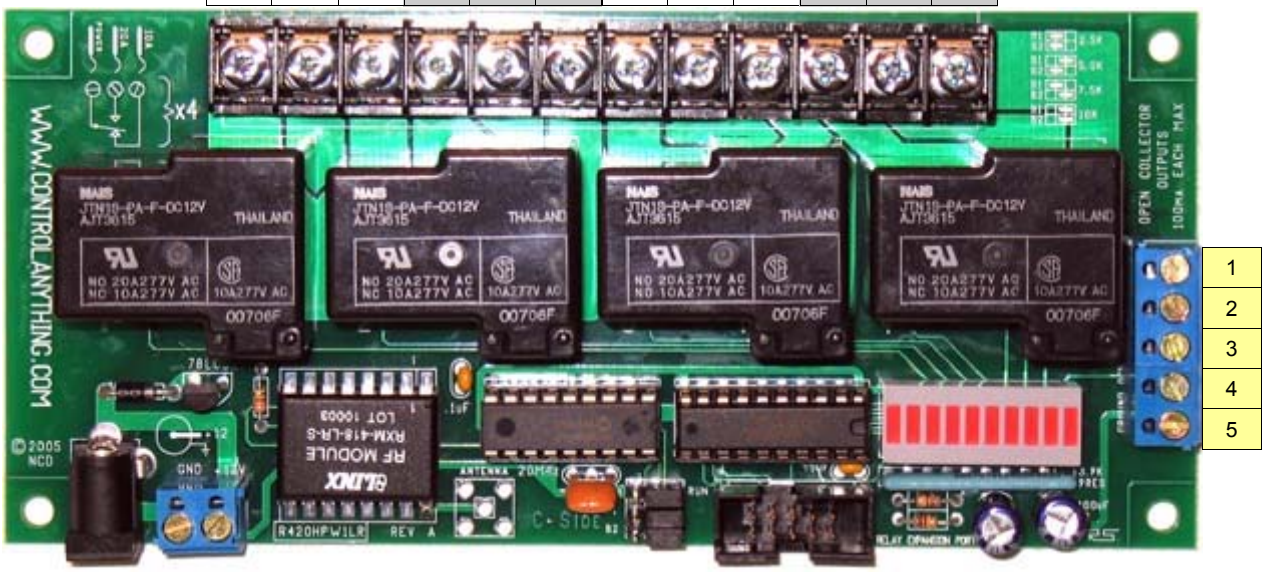
Relay 1		Relay 2		Relay 3		Relay 4		Relay 5		Relay 6		Relay 7		Relay 8	
NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC
COM	COM	COM	COM	COM	COM	COM	COM	COM	COM	COM	COM	COM	COM	COM	COM
NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO



R420HPW1LR, R820HPW1LR:

Special Notes for SPDT Relays on the HP Series Controllers:
Do Not Exceed 20A at 250VAC/24VDC Between COM and NC
Do Not Exceed 10A 250VAC/24VDC Between COM and NO

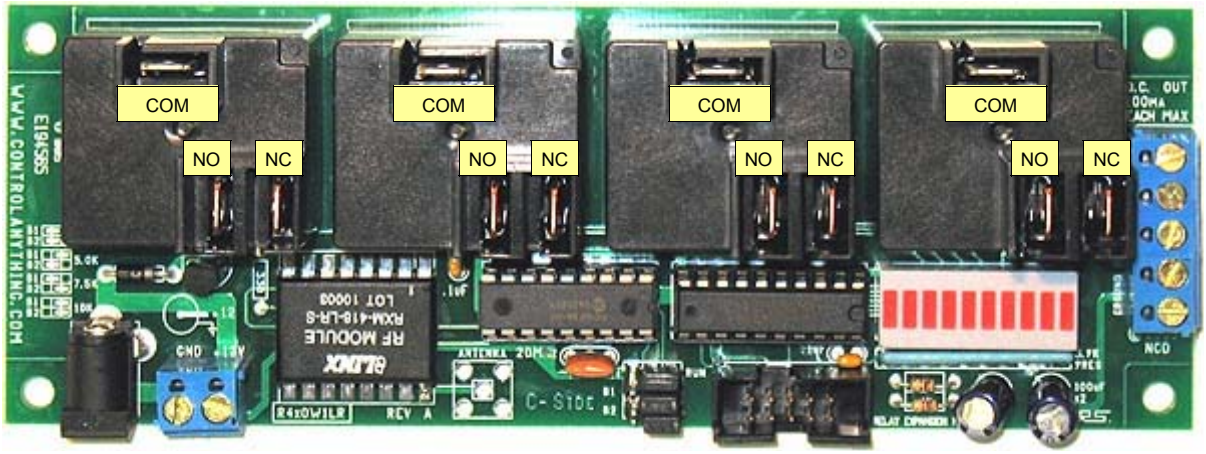
Relay 1			Relay 2			Relay 3			Relay 4		
COM	NO	NC	COM	NO	NC	COM	NO	NC	COM	NO	NC



R420HPW1LR, R420W1LR, R430W1LR ONLY:

These relay controllers can drive four external relays using the connections shown above. Apply a DC voltage to one side of the coil on the external relay. The relay controller will connect your external relay coil to ground. Be sure to share the ground of your external DC supply with the Shared Ground.

1	External Relay #8	Connect to an External Relay Coil on the Negative Side, Apply Voltage to the External Relay Coil on the Positive Side.
2	External Relay #7	Connect to an External Relay Coil on the Negative Side, Apply Voltage to the External Relay Coil on the Positive Side.
3	External Relay #6	Connect to an External Relay Coil on the Negative Side, Apply Voltage to the External Relay Coil on the Positive Side.
4	External Relay #5	Connect to an External Relay Coil on the Negative Side, Apply Voltage to the External Relay Coil on the Positive Side.
5	Shared Ground Connection	Locate the ground wire of your external power supply used to power the coils of the external relays. Connect this ground wire to the shared ground connection. The relay controller will activate the external relay by connecting it to ground.



AirRelay Modes of Operation

Configuration Mode	Labeled "Program" on the board, Configuration mode is used to: • Program Device Keys • Program "Requires Keys" Mode • Clear "Requires Keys" Mode	1. Power Down the Relay Controller 2. Set the Program/Run jumper to "Program" (printed on the circuit board) 3. Power Up the Relay Controller 4. Use the AirSwitch Software Interface Buttons Shown Below to Program Different Modes of Operation. When Programming Device Keys, use the Dark Gray Sliders to Set the Key Values you would like to program. *These Commands Only Work when the Remote Device is in "PROGRAM" Mode* Program the Device Number into the Controller (Set the Device Key Above, Then Press this Button to Store the Device Number) Remote Device will NOT Require Keys to Process Each Command / Remote Device will Require Keys to Process Each Command
Run Mode	Run Mode is the normal mode of operation for the AirRelay series controllers. In Run Mode, only relay control and networking commands will functions. The command buttons shown above right will not function, preventing accidental programming of device keys.	1. Power Down the Relay Controller 2. Set the Program/Run jumper to the "Run" position (printed on the circuit board) 3. Power Up the Relay Controller

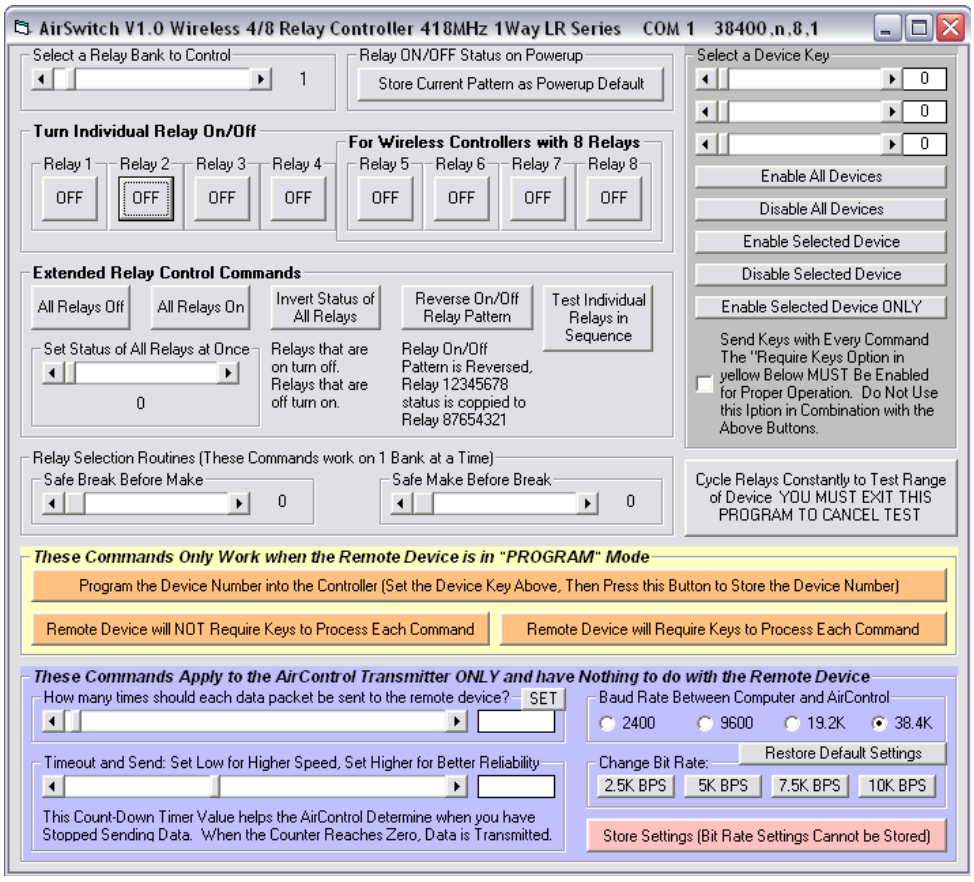
AirSwitch Software

Our AirSwitch software was written using Visual Basic 6. For your convenience, we have provided the Source Code so you can see how various functions operate. The AirSwitch software was written using XP Professional as the primary operating system. Our software releases require very minimal system resources, and are universally compatible with Windows 95, 98, ME, XP Home, and XP Pro.

We have not tested our software using Windows 2000 and we have reason to believe that some Windows 2000 systems will be unable to properly communicate via the serial port.

Download the AirSwitch software from our web site. After installation, choose the Start Menu, Programs, AirSwitch, then choose an AirSwitch program that matches the communication settings on your computer. For instance, Choose "AirSwitch COM1" if you are using the AirControl connected to COM 1 to communicate to the AirRelay line of controllers. Make sure the AirControl is Set for 38.4K Baud.

YOU WILL RECEIVE AN ERROR IF THE AIRCONTROL IS NOT FOUND ON THE SERIAL PORT.



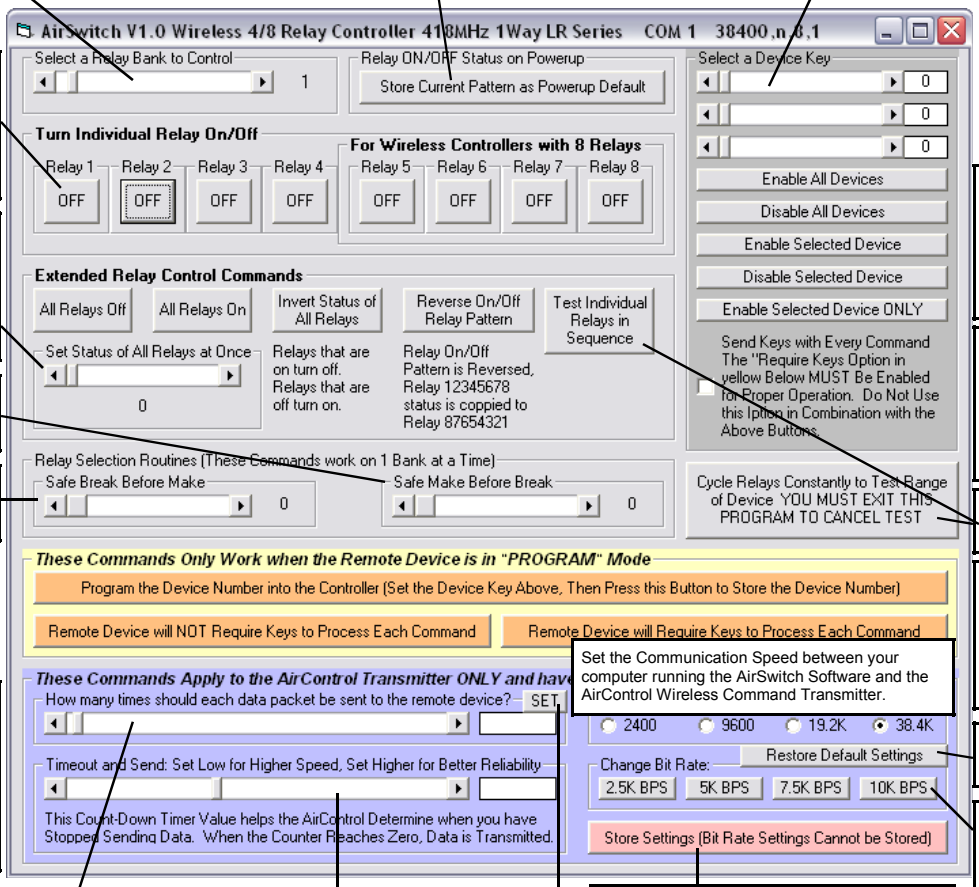
AirSwitch Interface

We strongly encourage our customers to carefully experiment with ALL functions in the AirSwitch interface. Experimentation will greatly increase your understanding of how the AirControl and the AirRelay controllers work together to process your commands. This interface has been written to carefully demonstrate the most powerful functions our AirRelay controllers have to offer. We highly suggest careful reading of all captions below. This is an excellent opportunity to quickly learn how functions are processed, and will be a very valuable tool when you begin programming your own control software. Some of the most basic functions have some hidden capabilities, that are dependant on other functions and interface settings. Take careful note of the Bank Control slider and the Device Key slider. This program can effectively demonstrate control of an unlimited number of wireless devices.

This Slider is Used to Select a Relay Bank to Control. Most relay controllers only have 1 bank of 4 or 8 relays. Most of our AirRelay controllers have an XR expansion port, allowing you to add Relay Banks to your AirRelay Controller. Set this Slider to 0 to control all banks at one time. This slider does not perform any functions by itself, but this slider is referenced by nearly all commands on this interface.

This Button is used to set the Power Up Default Status of Relays. The Relay Bank Slider (left) performs an Important Roll in the operation of this command. Setting the slider to 0 stores the power up default status of all relay banks. Setting the slider to another value allows you to store a default relay pattern for a specific relay bank, leaving all other banks unchanged.

The Key Sliders are used to select a device number for the AirSwitch software. The selected device number set by these sliders are referenced by many other commands in the AirSwitch interface. These sliders are also used to define the remote device number, provided the remote device is in "Program" mode.



These buttons allow you to individually control the on/off status of each relay. Setting the "Select Relay Bank to Control" slider to 0 will cause these buttons to function on all banks of relays.

This slider allows you to write a byte of data directly to one or all relay banks depending on the bank control slider. This slider is most useful for setting the status of 8 relays at a time.

This slider turns on all relays, pauses, and turns off a selected relay.

This slider turns off all relays, pauses, and turns on a selected relay.

The five buttons shown at left demonstrate the use of EWC commands. These commands do not function if the remote device is programmed to "Require Keys".

The Check Box at Left is used to determine if Keys are Transmitted along with Each Command. "Require Keys" MUST be enabled on the remote device.

These buttons are most useful for range testing.

The Yellow buttons shown at left are used to define settings in a remote device. The buttons do not work if the Remote Device is in RUN mode.

Set the Communication Speed between your computer running the AirSwitch Software and the AirControl Wireless Command Transmitter.

Just in case things stop working properly.

Allows you to change Bit Rates at Any Time. These Buttons over-ride the Dip Switch settings on the AirControl. The Dip Switch Settings define the Power Up Default Values ONLY. These buttons are useful for communicating to devices using multiple networks. Slower speeds can increase communication range in some installations.

The settings in Blue allow you to change important communications settings in the AirControl Wireless RF Command Transmitter. Some Settings may be updated in real time for your testing, or may be stored as Default Settings.

The Slider Values at Left will be Stored into the AirControl as the Powerup Default Settings.

The SET Button is used to change how many times each command is transmitted. For important commands, you may want to transmit them 255 times, whereas less important commands may only be sent a few times. The SET button references the slider below the button.

This is probably one of the most useful sliders the AirSwitch has to offer. This slider allows you to send data packets multiple times to ensure reliable reception by the remote device. Sending multiple packets is a function provided by the AirControl. All you have to do is tell the AirControl how many times each packet should be sent. The default value can be programmed by the red button in the lower right corner of the interface. You can over-ride this setting using the SET button, allowing you to transmit important commands for a longer duration. Regardless of how many times packets are sent, all NCD wireless remote devices will only process one valid data packet. Redundant data packets will be ignored. A new data packet is established when a new serial command is sent to the AirControl.

Sending multiple data packets can GREATLY increase communication range, but at the cost of speed. Only you can determine a good value for this application. By default, a modest value of 5 is selected, providing excellent speed and very good communication range.

TimeOut and Send (TOS) can be a very important slider, allowing you to finely tune the performance of your system with to the AirControl. This slider helps the AirControl determine when your computer has stopped sending a command. Lower values can increase performance, but can also cause communication errors. If you find any unreliable operation, you should increase this slider. Setting to a value of 100 is usually excessive, even on slow computers. A value of 255 will greatly slow communication speed, any will offer no better reliability than a setting of around 100. We have seen this slider work properly with a fast computer as low as 1. But for most applications, you will find a setting of 20-50 to offer excellent performance and reliability. Experimentation is recommended when adjusting this slider. If you have a fast computer and you are compiling your software, you may see incredible reliability and speed at a very low setting.

Understanding Keys

Any wireless NCD device that is capable of receiving wireless remote commands has the ability to use “keys”. A key is best described as a device number. A complete device number, or key, is made up of three numbers. Each number has a value of 0 to 255.

Some devices, such as the AirMonitor, are capable of connecting to the serial port, allowing you to program the keys into the controller via the serial port. Other devices, such as our AirRelay line of relay controllers, require that you place the remote device in Configuration Mode (Also known as PROGRAM or CONFIG mode) before keys are wirelessly programmed into the remote device. Wireless devices are typically not useable while in configuration mode. Wireless devices must be in run mode to operate correctly. Some devices use a program/run jumper while other devices use DIP switches to select modes of operation.

This lockout system is used to prevent accidental programming of Keys into wireless devices that should not be programmed. All NCD wireless devices have some hardware method of locking out a device from accidental programming. Some devices are programmed wirelessly, while other devices may require a direct connection to the serial port. You will appreciate our lockout strategy more and more as your wireless network grows to include more devices. You will be able to safely program the device number keys into new controllers without ever being concerned with accidental reprogramming of existing devices. Nonetheless, we STONGLY encourage you to label each device with its Keys. Some devices will not be able to tell you what it’s keys are. So once they are programmed, you must keep track of them, otherwise you may not be able to control the wireless device. New keys can be programmed at any time, so your device is never “lost”.

Keys Required Networking Method

When a remote device has the “Keys Required” or “Require Keys” option enabled, keys must be transmitted, along with each command to unlock the command.

Here is an example of how you would turn on a relay using three different devices that have the “Require Keys” option enabled.

The First device is programmed with KEYS 1,1,1
The Second device is programmed with KEYS 2,2,2
The Third device is programmed with KEYS 3,3,3

(Keep in mind, you can use any values for keys, each number must be a value from 0 to 255).

The following code turns on relay 1 on KEY 1,1,1

```
MSComm1.Output = Chr$(254)    'Enter Command Mode
MSComm1.Output = Chr$(1)      'Device Number (Key 1) to Enable
MSComm1.Output = Chr$(1)      'Device Number (Key 2) to Enable
MSComm1.Output = Chr$(1)      'Device Number (Key 3) to Enable
MSComm1.Output = Chr$(1)      'Control Relays on Bank 1
MSComm1.Output = Chr$(8)      'Turn on the First Relay.
```

The following code turns on relay 1 on KEY 2,2,2

```
MSComm1.Output = Chr$(254)    'Enter Command Mode
MSComm1.Output = Chr$(2)      'Device Number (Key 1) to Enable
MSComm1.Output = Chr$(2)      'Device Number (Key 2) to Enable
MSComm1.Output = Chr$(2)      'Device Number (Key 3) to Enable
MSComm1.Output = Chr$(1)      'Control Relays on Bank 1
MSComm1.Output = Chr$(8)      'Turn on the First Relay.
```

The following code turns on relay 1 on KEY 3,3,3

```
MSComm1.Output = Chr$(254)    'Enter Command Mode
MSComm1.Output = Chr$(3)      'Device Number (Key 1) to Enable
MSComm1.Output = Chr$(3)      'Device Number (Key 2) to Enable
MSComm1.Output = Chr$(3)      'Device Number (Key 3) to Enable
MSComm1.Output = Chr$(1)      'Control Relays on Bank 1
MSComm1.Output = Chr$(8)      'Turn on the First Relay.
```

This method is very easy to implement and offers EXCELLENT noise immunity. The drawbacks being that you can only speak to one device at a time. Since keys are sent along with every command, execution time will be slightly slower.

EWC Networking Method

The EWC Networking Command set is explained in much greater detail on the next page. The following example shows how you would turn on a relay using three different devices using the EWC networking method.

The First device is programmed with KEYS 1,1,1
The Second device is programmed with KEYS 2,2,2
The Third device is programmed with KEYS 3,3,3
The “Require Keys” option is OFF on ALL remote devices.

Enable Device with KEY 1,1,1

```
MSComm1.Output = Chr$(254)    'Enter Command Mode
MSComm1.Output = Chr$(252)    'Enable a Remote Device with These Keys:
MSComm1.Output = Chr$(1)      'Device Number (Key 1) to Enable
MSComm1.Output = Chr$(1)      'Device Number (Key 2) to Enable
MSComm1.Output = Chr$(1)      'Device Number (Key 3) to Enable
```

All subsequent commands will be directed to Key 1,1,1.

Turn On Relay 1

```
MSComm1.Output = Chr$(254)    'Enter Command Mode
MSComm1.Output = Chr$(1)      'Control Relays on Bank 1
MSComm1.Output = Chr$(8)      'Turn on the First Relay.
```

Enable Device with KEY 2,2,2

```
MSComm1.Output = Chr$(254)    'Enter Command Mode
MSComm1.Output = Chr$(252)    'Enable a Remote Device with These Keys:
MSComm1.Output = Chr$(2)      'Device Number (Key 1) to Enable
MSComm1.Output = Chr$(2)      'Device Number (Key 2) to Enable
MSComm1.Output = Chr$(2)      'Device Number (Key 3) to Enable
```

All subsequent commands will be directed to Key 2,2,2.

Turn On Relay 1

```
MSComm1.Output = Chr$(254)    'Enter Command Mode
MSComm1.Output = Chr$(1)      'Control Relays on Bank 1
MSComm1.Output = Chr$(8)      'Turn on the First Relay.
```

Enable Device with KEY 3,3,3

```
MSComm1.Output = Chr$(254)    'Enter Command Mode
MSComm1.Output = Chr$(252)    'Enable a Remote Device with These Keys:
MSComm1.Output = Chr$(3)      'Device Number (Key 1) to Enable
MSComm1.Output = Chr$(3)      'Device Number (Key 2) to Enable
MSComm1.Output = Chr$(3)      'Device Number (Key 3) to Enable
```

All subsequent commands will be directed to Key 3,3,3.

Turn On Relay 1

```
MSComm1.Output = Chr$(254)    'Enter Command Mode
MSComm1.Output = Chr$(1)      'Control Relays on Bank 1
MSComm1.Output = Chr$(8)      'Turn on the First Relay.
```

YOU SHOULD ALWAYS CHOOSE A WIRELESS NETWORKING METHOD, NEVER MIX METHODS WITHIN RADIO RANGE OF EACH OTHER. SINCE KEYS CAN BE ANY VALUE, KEYS CAN TRIGGER RELAYS IF NETWORKING METHODS ARE MIXED.
IF THE AIRMONITOR IS USED TO ADD WIRELESS NETWORKING TO EXISTING NCD DEVICES, IT WILL NOT BE POSSIBLE TO ISSUE E3C COMMANDS TO THE REMOTE DEVICE IF THE EWC NETWORKING METHOD IS USED.

The EWC Command Set: Software Control of Multiple Wireless NCD Devices

Using the EWC Command Set, it is possible to remotely control any number of remote devices individually or simultaneously. EWC commands are used to "Enable" and "Disable" remote devices. Each remote device is identified by a unique Key, which is a set of three numbers that you store into the remote device. Once these three keys have been stored, you can use EWC commands to enable and disable the remote device. Using this simple protocol, it is very easy to control which devices are responding to your commands and which devices are ignoring your commands. Keep in mind, EWC commands are the only set of commands that are NEVER ignored. NOTE: THE EWC COMMAND SET DOES NOT WORK ON REMOTE DEVICES THAT HAVE THE "KEYS REQUIRED" OPTION ENABLED. IN GENERAL, THE "KEYS REQUIRED" OPTION IS A LITTLE EASIER TO USE, BUT COMMUNICATIONS IS SLOWER, AND IS LIMITED TO CONTROLLING ONLY ONE DEVICE AT A TIME.

The EWC command set is virtually identical to the E3C command set found on our directly wired RS-232 devices. There are only a few minor differences. Three keys are used to replace a single device number. The AirControl does not really know anything about EWC commands, it just transmits the command codes to all wireless devices within range. The remote device then determines if it should be enabled or disabled. Since the AirControl is used for communication of EWC commands, the AirControl will report back an 85 after processing, which is another notable difference from standard E3C commands. Since our wireless LR Series controllers are only one way, it is not possible to retrieve the keys or other settings from the remote device. So take note of your keys (device number) settings stored within your remote devices. Lastly, E3C command 253 was eliminated from the EWC command set due to its lack of use.

Networking Comparison

As you probably already know, it is possible to control an unlimited number of devices remotely. There are two ways of doing this, depending on your preference. Devices are identified by a device number, which consists of three number, each set to a value between 0 and 255. You will need to program the remote device with a device number, usually done when the remote device is in "Program" mode. When the remote device is in "Run" mode, it is ready to respond to your commands.

Require Keys Method
This is a very convenient protocol, very easy to implement that allows you to control one remote device at a time. Communications is slower, but this method offer superior 72-bit noise rejection. When the "Require Keys" options is enabled on a remote device, keys must be immediately transmitted after the 254 command mode header. Only the device with correct key will respond. This is one our favorite methods of remote control because of it's ease of implementation….just add the three key numbers after the 254 and before the actual command code. It really doesn't get any easier than to use the Required Keys Method.

EWC Networking
EWC cuts down on the number of data bytes sent out the serial port, which greatly increases communication speed. Noise rejection is limited to 42 bits using this method. However, it is possible to control multiple devices at one time by enabling and disabling remote devices prior to sending commands. Only the enabled devices will accept and process your commands. The apropriate command and three keys can be transmitted to control which devices are enabled (listening to your control commands) and which devices are disabled (ignoring your control commands).

The EWC Command Set

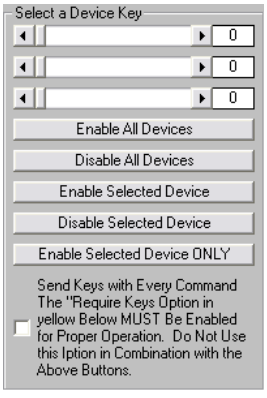
248 Enable All Devices:
Tells all devices to respond to your commands.

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.
You will need to send 3 keys, identifying the device you would like to enable.

251 Disable Selected Device:
Tells a specific device to ignore your commands.
You will need to send 3 keys, identifying the device you would like to disable.

252 Enable Selected Device Only:
Tells a specific device to listen to your commands, all other devices will ignore your commands. You will need to send 3 keys, identifying the device you would like to enable.



The AirSwitch Software uses the five buttons shown in the photo to demonstrate the use of EWC commands.

When the check box is enabled (left), the AirSwitch Software patches in a set of keys that are transmitted with each command using the "Keys Required" networking method.

Sample Code: The EWC Command Set

```
Public Sub EnableAllDevices()
    'Enable All EWC Devices
    MSComm1.Output = Chr$(254)    'Enter Command Mode
    MSComm1.Output = Chr$(248)    'EWC Enable All Device Command
    GetData
End Sub

Public Sub DisableAllDevices()
    'Disable All EWC Devices
    MSComm1.Output = Chr$(254)    'Enter Command Mode
    MSComm1.Output = Chr$(249)    'EWC Disable All Device Command
    GetData
End Sub

Public Sub EnableSpecificDevice(Device)
    'Enable A Specific EWC Devices, Other Devices will be unchanged
    MSComm1.Output = Chr$(254)    'Enter Command Mode
    MSComm1.Output = Chr$(250)    'EWC Disable Specific Device Command
    MSComm1.Output = Chr$(Key1)   'Device Number (Key 1) to Enable
    MSComm1.Output = Chr$(Key2)   'Device Number (Key 2) to Enable
    MSComm1.Output = Chr$(Key3)   'Device Number (Key 3) to Enable
    GetData
End Sub

Public Sub DisableSpecificDevice(Device)
    'Disable A Specific EWC Devices, Other Devices will be unchanged
    MSComm1.Output = Chr$(254)    'Enter Command Mode
    MSComm1.Output = Chr$(251)    'EWC Disable Specific Device Command
    MSComm1.Output = Chr$(Key1)   'Device Number (Key 1) to Disable
    MSComm1.Output = Chr$(Key2)   'Device Number (Key 2) to Disable
    MSComm1.Output = Chr$(Key3)   'Device Number (Key 3) to Disable
    GetData
End Sub

Public Sub DisableAllDevicesExcept(Device)
    'Disable All EWC Devices Except (Device)
    MSComm1.Output = Chr$(254)    'Enter Command Mode
    MSComm1.Output = Chr$(252)    'EWC Disable All Device Except Command
    MSComm1.Output = Chr$(Key1)   'Device Number (Key 1) to Enable
    MSComm1.Output = Chr$(Key2)   'Device Number (Key 2) to Enable
    MSComm1.Output = Chr$(Key3)   'Device Number (Key 3) to Enable
    GetData
End Sub
```

Transmitting Wireless Commands to the AirRelay via the AirControl

The AirControl is capable of sending and receiving data via RS-232 serial communications. The AirControl 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 sending data to the AirControl. 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.

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.

In your program, you will need to make provisions for reading data from the serial port. Your programming language will support commands for reading data from the AirControl.

For your convenience, we have provided several programming examples in Visual Basic 6 for communicating with the AirControl and the AirRelay. These examples should greatly speed development time. You may want to visit **www.controleverything.com** for the latest software and programming examples.

Programming examples for the AirRelay and the AirControl 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.

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 where you want to speak to lots of outputs (which are numbered), inputs (which are also numbered), or control specific devices using their device number (from 0 to 255).

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, they 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.

Introduction to AirControl and AirRelay Communications

The AirControl is a universal command encoder/transmitter for use with 418 MHz wireless 1-way devices, such as the AirRelay line of relay controllers. The AirControl connects to the serial port of your computer and waits for a command from the AirSwitch software or your own computer program. All communication is done with the AirControl. The AirControl handles all communications with the wireless device. Speaking to the AirControl is just as easy as speaking with any other NCD device. There are only a few very subtle differences: 1) the AirControl responds to every command by send you back an ASCII character code 85, signifying the successful completion of your command. 2) There are two separate command sets for you to work with. There is a command set built into the remote device, which always begins with ASCII character code 254. There is also a separate command set that allows you to speak to the AirControl hardware at any time. AirControl commands always begin with ASCII character code 253. The AirControl command set can help you control the speed and distance of communication. Please see the AirControl manual for more information on the use of AirControl commands. This manual will focus on the AirRelay line of long-range wireless one-way relay controllers.

Depending on the settings you have chosen for the AirControl, data transmission can be very fast or very slow. ASCII character code 85 is used to signal your computer when transmission of your command has completed. Use the Visual Basic routine below to wait for the ASCII character code 85 to be returned.

```
Public Function GetData()
    Do                                      'Wait for Device to Reply
      DoEvents                              'Allow Windows to MultiTask
    Until MSComm1.InBufferCount > 0         'If the Device Replies
    GetData = Asc(MSComm1.Input)            'Return Response Code
    Debug.Print GetData
End Sub
```

Here is another example of the GetData function. This routine includes a timeout, which keeps the program from ever locking up should the AirControl take to long to respond. The 25,000 value below may need to be increased or decreased depending on AirControl settings. Experimentation with the usage of this routine is advised...but keep in mind you should NEVER try to send a new command while the AirControl is busy transmitting your previous command. Doing so will cause your new command to be ignored. Use this routine with caution, we have never seen the AirControl lock up, but it can take several seconds to complete a transmission is some AirControl settings are set too high.

```
Public Function GetData()
    Timeout = 0
    Do                                      'Wait for Device to Reply
      Timeout = Timeout + 1
      If Timeout > 25000 then Exit Function
      DoEvents                              'Allow Windows to MultiTask
    Until MSComm1.InBufferCount > 0         'If the Device Replies
    GetData = Asc(MSComm1.Input)            'Return Response Code
    Debug.Print GetData
End Sub
```

Important Note:

To save space in the manual, we have written all commands as if the "Require Keys" function is off on the AirRelay controllers. THIS IS NOT NECESSARILY OUR RECOMMENDATION, WE ARE ONLY TRYING TO SIMPLIFY THE MANUAL. The syntax of each command will depend on the "Require Keys" setting in the remote device.

If "Keys Required" is OFF, use the Following Command Structure:

```
MSComm1.Output = Chr$(254)     'Enter Command Mode
MSComm1.Output = Chr$(1)       'Control Relays on Bank 1
MSComm1.Output = Chr$(8)       'Turn on the First Relay.
```

If "Keys Required" is ON, use the Following Command Structure:

```
MSComm1.Output = Chr$(254)     'Enter Command Mode
MSComm1.Output = Chr$(Key1)    'Device Number (Key 1) to Enable
MSComm1.Output = Chr$(Key2)    'Device Number (Key 2) to Enable
MSComm1.Output = Chr$(Key3)    'Device Number (Key 3) to Enable
MSComm1.Output = Chr$(1)       'Control Relays on Bank 1
MSComm1.Output = Chr$(8)       'Turn on the First Relay.
```

NEVER MIX EWC COMMANDS WITH "REQUIRE KEY" COMMANDS ON DEVICES THAT ARE WITHIN COMMUNICATION RANGE. THE EWC DEVICES CAN RESPOND TO KEYS AS IF THEY WERE COMMANDS, CAUSING UNRELIABLE OPERATION.

Reading LED and Sending Commands

This section is useful for understanding the communication process. This section will help you determine where to look if you have any communication problems.

The AirControl has no knowledge of any particular command set except its own. You will know you have successfully communicated with the AirControl when you receive ASCII character code 85 back from the AirControl transmitter. The AirControl will only respond to commands that begin with 254 (for controlling remote devices) or commands that begin with 253 (for changing AirControl settings).

You will know that the AirControl is transmitting your commands when you see the Transmit LED light up on the AirControl transmitter. You will also know the remote relay controller is decoding your commands when you see one of the LEDs flash during data transmission. Depending on the AirRelay controller, at least one LED will flash when a command is received. If the LED does not flash, then the AirRelay is either not within radio range or the Bit Rate settings are incorrect. By default, AirRelay controllers are set to 10KBPS. By default, the AirControl will communicate to your PC at 38.4K baud, and transmit at 10KBPS when all 4 dip switches are in the ON position.

You will know there is a problem communicating to the AirControl if you never get a response.

You will know when a command has been sent by watching the Transmit LED on the AirControl.

You will know there is a reception/decoding problem on the AirRelay by watching the Receive (Decoding LED), which only lights when a valid packet header is detected and only turns off when it has completed the processing of your command.

If you experience any unreliable or no operation (but all LEDs are flashing), make sure the TOS value is set to a reasonable value (100 is almost always safe). See the AirControl manual for details on changing the TOS value or use our AirSwitch software to set the TOS value.

If all LEDs are flashing and there is still no communication, make sure you know the status of the "REQUIRE KEYS" option. This can prevent communication if this function is enabled. Make sure you always know the remote device key. Before contacting us for technical assistance, please use the AirSwitch software to attempt to establish communication with the remote device. The AirSwitch software is a very powerful troubleshooting and demonstration tool that should always be used if you ever have any communications problems.

AirRelay Command Set

The Following commands are built into the AirRelay line of wireless 1-way relay controllers. It is important to realize the AirControl will forward your commands to the AirRelay controllers. The AirControl has no knowledge of this command set. It will forward any command that begins with a 254. The AirControl will always send data back to the computer when transmission has completed. Please make sure you have read the previous page ***ENTIRELY*** before sending these commands. The GetData function, used in these examples, is fully explained on the previous page, as is the "Require Keys" option, which is very important to understanding command syntax.

The commands shown in this manual were written for Visual Basic 6 Professional. Other versions of Visual Basic may be used. Make sure your version supports the MSComm Control by right clicking the vb toolbox (with all the small icons), and select "Components". Make sure you have a check in the box that says "Microsoft Comm Control". You can also use the NCD ActiveX to send these commands from any programming language that supports ActiveX controls.

XR Ports, Relay Banks, and the Bank Parameter

Most NCD relay controllers manufactured in 2005 and later have an XR relay expansion port. The XR port allows you to add banks of relays to your existing controller. A relay bank consists of 8 relays. The number of possible relay banks is determined by the firmware used. At the time of writing, the AirSwitch software and AirRelay firmware were written to allow you to individually control 16 banks of relays. This allows individual control of up to 128 different relays from a single AirRelay controller. This number is subject to change as we continue to evolve our product line.

Most commands have a "Bank" or "Relay Bank" parameter. In these cases, a value of 0-16 should be used to specify which relay bank to control. A value of 0 universally applies a particular command to all relay banks while a value of 1-16 directs a particular command to a specific relay bank. Please review the AirSwitch interface diagram found earlier in this manual and experiment with the AirSwitch software if you have any difficulty understanding relay banks. It is very important that you have a complete understanding of how the Bank parameter affects each command before continuing.

Controlling Individual Relays

The AirRelay allows you to set the status of individual relays. The following routine requires a Relay and a Bank parameter. A relay parameter value of 0 to 7 turns off relays 1 to 8. A relay parameter of 8 to 15 turns on individual relays 1 to 8. The bank parameter is described above.

Sample Code: Controlling Individual Relays

```
Public Sub SetRelayState(Bank,Relay)
    MSComm1.Output = Chr$(254)    'Enter Command Mode
    MSComm1.Output = Chr$(Bank)   'Specify a Relay Bank to Control
    MSComm1.Output = Chr$(Relay)  'Relay = 0-7 OFF, Relay = 8-15 ON.
    GetData                       'Wait for Command to Finish
End Sub
```

Controlling Multiple Relays

It is possible to set the status of all relays within a bank or all relay banks to the same value using the command shown at right. This command requires two parameters, a Value parameter sets all relays within a bank to the specified value. A value of 0 turns off all relays, a value of 255 turns all relays on. Any number in between appears as a binary pattern on the relay bank. For example, a value of 85 activates every other relay because the binary value of 85 is 01010101. The Bank parameter is described above.

Sample Code: Controlling Multiple Relays

```
Public Sub SetAllRelayState(Bank,Value)
    MSComm1.Output = Chr$(254)    'Enter Command Mode
    MSComm1.Output = Chr$(Bank)   'Specify a Relay Bank to Control
    MSComm1.Output = Chr$(40)     'Set the Status of All Relays
    MSComm1.Output = Chr$(Value)  'Byte to be Copied to Relay Banks
    GetData                       'Wait for Command to Finish
End Sub
```

Turn Off All Relays

The following command turns off all relays. This command requires only the Bank parameter, described above.

Sample Code: Turn Off All Relays

```
Public Sub TurnOffRelays(Bank)
    MSComm1.Output = Chr$(254)    'Enter Command Mode
    MSComm1.Output = Chr$(Bank)   'Specify a Relay Bank to Control
    MSComm1.Output = Chr$(29)     'Turn Off All Relays Command
    GetData                       'Wait for Command to Finish
End Sub
```

Turn On All Relays

The following command turns on all relays. This command requires only the Bank parameter, described above.

Sample Code: Turn On All Relays

```
Public Sub TurnOnRelays(Bank)
    MSComm1.Output = Chr$(254)    'Enter Command Mode
    MSComm1.Output = Chr$(Bank)   'Specify a Relay Bank to Control
    MSComm1.Output = Chr$(30)     'Turn On All Relays Command
    GetData                       'Wait for Command to Finish
End Sub
```

Invert Relays

The following command inverts the status of the relays in a given relay bank. Relays that are on turn off, relays that are off turn on. This command requires only the Bank parameter, described above.

Sample Code: Invert Relays

```
Public Sub InvertRelays(Bank)
    MSComm1.Output = Chr$(254)    'Enter Command Mode
    MSComm1.Output = Chr$(Bank)   'Specify a Relay Bank to Control
    MSComm1.Output = Chr$(31)     'Invert Relays Command
    GetData                       'Wait for Command to Finish
End Sub
```

Reverse Relays

The Reverse Relays command reverses the order of the relays on a specified bank. Relay 1 results in holding the status of relay 8. Relay 8 results in holding the status of relay 1. All relay statuses are reversed ordered. This command requires only the Bank parameter, described above.

Sample Code: Reverse Relays

```
Public Sub InvertRelays(Bank)
    MSComm1.Output = Chr$(254)    'Enter Command Mode
    MSComm1.Output = Chr$(Bank)   'Specify a Relay Bank to Control
    MSComm1.Output = Chr$(32)     'Reverse Relays Command
    GetData                       'Wait for Command to Finish
End Sub
```

AirRelay Command Set

The Following commands are built into the AirRelay line of wireless 1-way relay controllers. It is important to realize the AirControl will forward your commands to the AirRelay controllers. The AirControl has no knowledge of this command set. It will forward any command that begins with a 254. The AirControl will always send data back to the computer when transmission has completed. Please make sure you have read the previous page ***ENTIRELY*** before sending these commands. The GetData function, used in these examples, is fully explained on the previous page, as is the "Require Keys" option, which is very important to understanding command syntax.

The commands shown in this manual were written for Visual Basic 6 Professional. Other versions of Visual Basic may be used. Make sure your version supports the MSComm Control by right clicking the vb toolbox (with all the small icons), and select "Components". Make sure you have a check in the box that says "Microsoft Comm Control". You can also use the NCD ActiveX to send these commands from any programming language that supports ActiveX controls.

XR Ports, Relay Banks, and the Bank Parameter

Most NCD relay controllers manufactured in 2005 and later have an XR relay expansion port. The XR port allows you to add banks of relays to your existing controller. A relay bank consists of 8 relays. The number of possible relay banks is determined by the firmware used. At the time of writing, the AirSwitch software and AirRelay firmware were written to allow you to individually control 16 banks of relays. This allows individual control of up to 128 different relays from a single AirRelay controller. This number is subject to change as we continue to evolve our product line.

Most commands have a "Bank" or "Relay Bank" parameter. In these cases, a value of 0-16 should be used to specify which relay bank to control. A value of 0 universally applies a particular command to all relay banks while a value of 1-16 directs a particular command to a specific relay bank. Please review the AirSwitch interface diagram found earlier in this manual and experiment with the AirSwitch software if you have any difficulty understanding relay banks. It is very important that you have a complete understanding of how the Bank parameter affects each command before continuing.

Controlling Individual Relays

The AirRelay allows you to set the status of individual relays. The following routine requires a Relay and a Bank parameter. A relay parameter value of 0 to 7 turns off relays 1 to 8. A relay parameter of 8 to 15 turns on individual relays 1 to 8. The bank parameter is described above.

Sample Code: Controlling Individual Relays

```
Public Sub SetRelayState(Bank,Relay)
    MSComm1.Output = Chr$(254)    'Enter Command Mode
    MSComm1.Output = Chr$(Bank)   'Specify a Relay Bank to Control
    MSComm1.Output = Chr$(Relay)  'Relay = 0-7 OFF, Relay = 8-15 ON.
    GetData                       'Wait for Command to Finish
End Sub
```

Controlling Multiple Relays

It is possible to set the status of all relays within a bank or all relay banks to the same value using the command shown at right. This command requires two parameters, a Value parameter sets all relays within a bank to the specified value. A value of 0 turns off all relays, a value of 255 turns all relays on. Any number in between appears as a binary pattern on the relay bank. For example, a value of 85 activates every other relay because the binary value of 85 is 01010101. The Bank parameter is described above.

Sample Code: Controlling Multiple Relays

```
Public Sub SetAllRelayState(Bank,Value)
    MSComm1.Output = Chr$(254)    'Enter Command Mode
    MSComm1.Output = Chr$(Bank)   'Specify a Relay Bank to Control
    MSComm1.Output = Chr$(40)     'Set the Status of All Relays
    MSComm1.Output = Chr$(Value)  'Byte to be Copied to Relay Banks
    GetData                       'Wait for Command to Finish
End Sub
```

Turn Off All Relays

The following command turns off all relays. This command requires only the Bank parameter, described above.

Sample Code: Turn Off All Relays

```
Public Sub TurnOffRelays(Bank)
    MSComm1.Output = Chr$(254)    'Enter Command Mode
    MSComm1.Output = Chr$(Bank)   'Specify a Relay Bank to Control
    MSComm1.Output = Chr$(29)     'Turn Off All Relays Command
    GetData                       'Wait for Command to Finish
End Sub
```

Turn On All Relays

The following command turns on all relays. This command requires only the Bank parameter, described above.

Sample Code: Turn On All Relays

```
Public Sub TurnOnRelays(Bank)
    MSComm1.Output = Chr$(254)    'Enter Command Mode
    MSComm1.Output = Chr$(Bank)   'Specify a Relay Bank to Control
    MSComm1.Output = Chr$(30)     'Turn On All Relays Command
    GetData                       'Wait for Command to Finish
End Sub
```

Invert Relays

The following command inverts the status of the relays in a given relay bank. Relays that are on turn off, relays that are off turn on. This command requires only the Bank parameter, described above.

Sample Code: Invert Relays

```
Public Sub InvertRelays(Bank)
    MSComm1.Output = Chr$(254)    'Enter Command Mode
    MSComm1.Output = Chr$(Bank)   'Specify a Relay Bank to Control
    MSComm1.Output = Chr$(31)     'Invert Relays Command
    GetData                       'Wait for Command to Finish
End Sub
```

Reverse Relays

The Reverse Relays command reverses the order of the relays on a specified bank. Relay 1 results in holding the status of relay 8. Relay 8 results in holding the status of relay 1. All relay statuses are reversed ordered. This command requires only the Bank parameter, described above.

Sample Code: Reverse Relays

```
Public Sub InvertRelays(Bank)
    MSComm1.Output = Chr$(254)    'Enter Command Mode
    MSComm1.Output = Chr$(Bank)   'Specify a Relay Bank to Control
    MSComm1.Output = Chr$(32)     'Reverse Relays Command
    GetData                       'Wait for Command to Finish
End Sub
```