Integrator's Guide to Serial Protocols

Table of Contents

This document will describe the serial protocols for all of the NeoPro and Neothings products.

The first part of this guide describes the serial commands and responses which are common to both the USB and the RS-232 ports. The matrix under control does not treat either port different. The second part describes installation of the USB COM port drivers for a MS Windows based control system. In the case of an RS-232 controlled system, it is not necessary to install or use the USB port drivers, so these steps should be skipped.

In both cases the baud rate and settings for all communications is ***9600 baud, 8 data, no parity, 1 stop, no flow control***

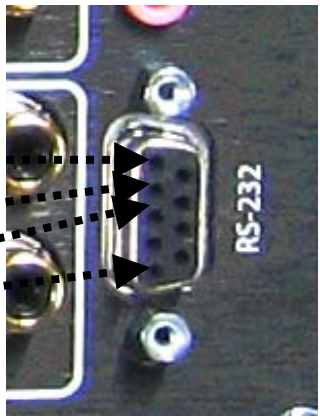
RS232 pinouts

Pin 1 (for reference only)

Pin 2 Transmit (Data out of unit)

Pin 3 Receive (Data in to unit)

Pin 5 Ground



The commands are structured so that a control program can communicate with the matrix in a structured and reliable two-way method. The control system can always understand the state of the matrix switch. These commands are also human readable ascii text, which will help in troubleshooting and testing.

Some system designers may choose to control the matrix with one way serial communications, which is also a perfectly acceptable method for control.

Some matrix switch models have the ability to switch audio and video separately. For these switches, the main concept to understand is that to the control system, the matrix looks like several independent switches. For convenience we have included ‘macro’ switch functions for controlling all switching levels in parallel.

Common Structures

A command is always wrapped in square braces.

```
[command]
```

It is not necessary to follow the command by any carriage returns or other special characters. The closing square brace will trigger the switch to process the command.

Responses from the switch are in a similar pattern. In many cases there are multiple response to a command, so a group of responses is wrapped in another set of square braces.

```
[[response1][response2][response3]]
```

Within a command or response, there will be one or more fields, separated by commas.

```
[B1,C8,01,02]
```

One Way Serial Hints

Any of the examples shown in this guide can be sent to the matrix at anytime without concern as to the state of the matrix. If you need to send a string of commands, wait about 150ms in between each command.

Errors

The switch will only attempt to process a command between matching [and] braces, so garbage before and after the braces is thrown away. However the characters between the braces will be processed, and if invalid, an error signal is sent.

Any command syntax error will result in a response of

```
[E]
```

Maximum Response Time

Maximum time to process any command and complete the serial response, is 150mS.

Avalon Protocol

Command structure prototype:

```
[Bx,Xx,X,Y]
```

Bx = board number

Xx = board type

X= the input number to route from

Y= the output number to route to

Bx Board number

B0 = Virtual board for all matrices

B1 = Component video matrix

B2 = Digital audio matrix

Xx Board Type

00 = Virtual board 0 type[1]

C2 = 8x2 component video matrix

C4 = 8x4 component video matrix

C8 = 8x8 component video matrix

B2 = 8x2 digital audio matrix

B4 = 8x4 digital audio matrix

B8 = 8x8 digital audio matrix

---

[1] Since board 0 will control multiple types of boards at the same time, the type field has no real meaning. Fill this field with 00 to make it a valid command.

X Input

0 = The ‘mute’ or disable selection

1-8 = Source inputs 1 through 8

Y Output

1-8 = Outputs 1 through 8, but only up to the max number of outputs on the model in use.

Examples

```
[B0,00,2,4]
```

Routes input 2 to output 4 on all matrices

```
[B2,B2,6,1]
```

Routes input 6 to output 1 on the audio 8x2 matrix

```
[B2,B8,0,1]
```

Mutes the audio on output 1 of the 8x8 audio matrix

The serial port will not echo characters back to the sender. Instead once the command is completed, the response will be sent back as shown in the Query section below.

Setup and Misc commands

The following commands are stored in EEPROM memory, so they will still be set after a power outage.

```
[S,x,y]
```

S = Setup function
x=function letter, y=argument

L=LED lights on the front panel
0=blackout

1=normal state (default)

R=Front IR receiver
0=disabled

1=enabled (default)

B=Front panel buttons
0=disabled

1=enabled (default)

V=Verbosity
0=responds only when sent a command

1=any change in state from front, IR, or other ports are sent out like queries (default)

Setup Example

```
[S,R,0]
```

will disable the front IR receiver

Power Control

```
[P,n]
```

P=power control

0=power off

1=power on

The response to these commands is the same as the command itself i.e. sending

```
[P,1]
```

will get a response of

```
[P,1]
```

Query commands

```
[?B0]
```

Putting a “?” inside the bracket, followed by Bx will return the full structure of the matrix being queried.

Since there are up to eight outputs on each board, up to eight structures will be returned per board.

Querying a single board will invoke a single board structure response, but querying a macro such as B0 will invoke all boards to respond.

Example response to [?B0] on an 8x4 Avalon matrix will look like:

```
[[B1,C4,3,1][B1,C4,2,2][B1,C4,6,3][B1,C4,2,4
][B2,B4,3,1][B2,B4,2,2][B2,B4,3,3][B2,B4,2,4
]]
```

Note that there is another level of braces that wraps the entire query response.

Sending the below query on a on an 8x2 matrix will look like:

```
[?B1]
```

Response:

```
[[B1,C2,3,1][B1,C2,2,2]]
```

Send:

```
[?V]
```

Returns product ID and version

```
[V,A12]
```

A= Avalon

12= major and minor version, or v1.2[2]

Send:

```
[?S]
```

Returns all setup parameters. i.e.:

```
[[S,L,1][S,R,1][S,B,1][S,V,1]]
```

Send:

```
[?P]
```

Returns power state

```
[P,1]
```

[2] The major and minor version is more referring to the serial protocol version. If there are any changes in the protocol, the at least the minor version will increment. However there may be minor-minor product firmware upgrades that occur that do not effect the serial protocol, and will not be reflected in theis version string.

Borrego Protocol

Structure prototype:

`[Bx,Xx,X,Y]`

Bx = board number

Xx = board type

X= the input number to route from

Y= the output number to route to

Bx Board number

B0 = Virtual board for all matrices

B1 = Component video matrix

B2=Analog audio matrix

B3=Composite video (or SPDIF) matrix

Xx Board Type

00 = Virtual board 0 type[3]

C2 = 8x2 component video matrix

C4 = 8x4 component video matrix

C8 = 8x8 component video matrix

B2 = 8x2 analog audio matrix

B4 = 8x4 analog audio matrix

---

[3] Since board 0 will control multiple types of boards at the same time, the type field has no real meaning. Fill this field with 00 to make it a valid command.

B8 = 8x8 analog audio matrix

A2 = 8x2 composite video (or SPDIF) matrix

A4 = 8x4 composite video (or SPDIF) matrix

A8 = 8x8 composite video (or SPDIF) matrix

X Input

0 = The ‘mute’ or disable selection

1-8 = Source inputs 1 through 8

Y Output

1-8 = Outputs 1 through 8, but only up to the max number of outputs on the model in use.

Examples

```
[B0,00,2,4]
```

Routes input 2 to output 4 on all matrices

```
[B2,B2,6,1]
```

Routes input 6 to output 1 on the audio 8x2 matrix

```
[B2,B8,0,1]
```

Mutes the audio on output 1 of the 8x8 audio matrix

The serial port will not echo characters back to the sender. Instead once the command is completed, the response will be sent back as shown in the Query section below.

Setup and Misc commands

The following commands are stored in EEPROM memory, so they will still be set after a power outage.

```
[S,x,y]
```

S = Setup function
x=function letter, y=argument

L=LED lights on the front panel
0=blackout

1=normal state (default)

R=Front IR receiver
0=disabled

1=enabled (default)

B=Front panel buttons
0=disabled

1=enabled (default)

V=Verbosity
0=responds only when sent a command

1=any change in state from front, IR, or other ports are sent out like queries (default)

Setup Example

```
[S,R,0]
```

will disable the front IR receiver

Power Control

```
[P,n]
```

P=power control

0=power off

1=power on

The response to these commands is the same as the command itself i.e. sending

```
[P,1]
```

will get a response of

```
[P,1]
```

Query commands

```
[?B0]
```

Putting a “?” inside the bracket, followed by Bx will return the full structure of the matrix being queried.

Since there are up to eight outputs on each board, up to eight structures will be returned per board.

Querying a single board will invoke a single board structure response, but querying a macro such as B0 will invoke all boards to respond.

Example response to [?B0] on an 8x4 Borrego matrix will look like:

```
[[B1,C4,3,1][B1,C4,2,2][B1,C4,6,3][B1,C4,2,4
][B2,B4,3,1][B2,B4,2,2][B2,B4,3,3][B2,B4,2,4
][B3,B4,3,1][B3,B4,2,2][B3,B4,3,3][B3,B4,2,4
]]
```

Note that there is another level of braces that wraps the entire query response.

Sending the below query on a on an 8x2 matrix will look like:

```
[?B1]
```

Response:

```
[[B1,C2,3,1][B1,C2,2,2]]
```

Send:

```
[?V]
```

Returns product ID and version

```
[V,B12]
```

B= Borrego

12= major and minor version, or v1.2[4]

Send:

[?S]

Returns all setup parameters. i.e.:

[[S,L,1][S,R,1][S,B,1][S,V,1]]

Send:

[?P]

Returns power state

[P,1]

[4] The major and minor version is more referring to the serial protocol version. If there are any changes in the protocol, the at least the minor version will increment.

Concord Protocol

Structure prototype:

`[Bx,X,Y]`

Bx = board number

X= the input number to route from

Y= the output number to route to

Bx Board number

B0 = Virtual board for all matrices

B1 = Component video matrix

B2=Analog audio matrix

B3=Digital audio/Composite video matrix

X Input

0 = The ‘mute’ or disable selection

1-8 = Source inputs 1 through 8

Y Output

1-8 = Outputs 1 through 8, but only up to the max number of outputs on the model in use.

Examples

```
[B0,2,4]
```

Routes input 2 to output 4 on all matrices

```
[B2,6,1]
```

Routes input 6 to output 1 on the analog audio matrix

```
[B2,0,1]
```

Mutes the audio on output 1 on the analog audio matrix

The serial port will not echo characters back to the sender. Instead once the command is completed, the response will be sent back as shown in the Query section below.

Setup and Misc commands

The following commands are stored in EEPROM memory, so they will still be set after a power outage.

```
[S,x,y]
```

S = Setup function
x=function letter, y=argument

L=LED lights on the front panel
0=blackout

1=normal state (default)

R=Front IR receiver
0=disabled

1=enabled (default)

B=Front panel buttons
0=disabled

1=enabled (default)

V=Verbosity
0=responds only when sent a command

1=any change in state from front, IR, or other ports are sent out like queries (default)

Setup Example

```
[S,R,0]
```

will disable the front IR receiver

Power Control

```
[P,n]
```

P=power control

0=power off

1=power on

The response to these commands is the same as the command itself i.e. sending

```
[P,1]
```

will get a response of

```
[P,1]
```

Query commands

```
[?B0]
```

Putting a “?” inside the bracket, followed by Bx will return the full structure of the matrix being queried.

Since there are up to eight outputs on each board, up to eight structures will be returned per board.

Querying a single board will invoke a single board structure response, but querying a macro such as B0 will invoke all boards to respond.

Example response to [?B0] will look like:

```
[[B1,3,1][B1,2,2][B1,6,3][B1,2,4][B1,3,5][B1
,2,6][B1,6,7][B1,2,8][B2,3,1][B2,2,2][B2,6,3
][B2,2,4][B2,3,5][B2,2,6][B2,6,7][B2,2,8][B3
,3,1][B3,2,2][B3,6,3][B3,2,4][B3,3,5][B3,2,6
][B3,6,7][B3,2,8]]
```

Note that there is another level of braces that wraps the entire query response.

Sending the below query on a on an 8x2 matrix will look like:

```
[?B1]
```

Response:

```
[[B1,3,1][B1,2,2][B1,6,3][B1,2,4][B1,3,5][B1
,2,6][B1,6,7][B1,2,8]]
```

Send:

```
[?V]
```

Returns product ID and version

```
[V,C12]
```

C= Concord

12= major and minor version, or v1.2[5]

Send:

```
[?S]
```

Returns all setup parameters. i.e.:

```
[[S,L,1][S,R,1][S,B,1][S,V,1]]
```

Send:

---

[5] The major and minor version is more referring to the serial protocol version. If there are any changes in the protocol, the at least the minor version will increment.

```
[?P]
```

Returns power state

```
[P,1]
```

Delano Protocol

Structure prototype:

[DV,XX,YY]

DV = Delano Video

XX= the input number to route from

YY= the two digit output number to route to

XX Input

00 = The ‘mute’ or disable selection

01-08 = Source inputs 1 through 8, this field shall always use two characters.

YY Output

01-16 = Outputs 1 through 16, this field shall always use two characters.

Examples

```
[DV,03,01]
```

Routes input 3 to output 1

```
[DV,00,16]
```

Routes input 00 (mute) to output 16

The serial port will not echo characters back to the sender. Instead once the command is completed, the response will be sent back as shown in the Query section below.

Setup and Misc commands

The following commands are stored in EEPROM memory, so they will still be set after a power outage.

```
[S,x,y]
```

S = Setup function
x=function letter, y=argument

L=LED lights on the front panel
0=blackout

1=normal state (default)

R=Front IR receiver
0=disabled

1=enabled (default)

B=Front panel buttons
0=disabled

1=enabled (default)

V=Verbosity
0=responds only when sent a command

1=any change in state from front, IR, or other ports are sent out like queries (default)

Setup Example

```
[S,R,0]
```

will disable the front IR receiver

Power Control

```
[P,n]
```

P=power control

0=power off

1=power on

The response to these commands is the same as the command itself i.e. sending

```
[P,1]
```

will get a response of

```
[P,1]
```

Query commands

```
[?D]
```

Putting a “?” inside the bracket, followed by D will return the full state of the matrix.

Example response to [?D] on an Delano matrix will look like:

```
[[DV,01,01][DV,01,02][DV,01,03][DV,04,04][DV,01,05][DV,00,06][DV,07,07][DV,02,08][DV,01,09][DV,03,10][DV,00,11][DV,01,12][DV,04,13][DV,08,14][DV,00,15][DV,00,16]]
```

Note that there is another level of braces that wraps the entire query response.

Send:

```
[?V]
```

Returns product ID and version

```
[V,D10]
```

D= Delano

10= major and minor version, or v1.0[6]

Send:

[?S]

Returns all setup parameters. i.e.:

[[S,L,1][S,R,1][S,B,1][S,V,1]]

Send:

[?P]

Returns power state

[P,1]

[6] The major and minor version is more referring to the serial protocol version. If there are any changes in the protocol, the at least the minor version will increment.

Eureka Protocol

Structure prototype:

`[EX,XX,YY]`

ED = Eureka digital audio matrix

EA = Eureka analog audio matrix

E0 = Both digital and analog audio matrices

XX= the input number to route from

YY= the two digit output number to route to

XX Input

00 = The ‘mute’ or disable selection

01-08 = Source inputs 1 through 8, this field shall always use two characters.

YY Output

01-16 = Outputs 1 through 16, this field shall always use two characters.

Examples

`[E0,03,01]`

Routes input 3 to output 1 for both digital and analog audio

`[ED,00,16]`

Routes input 00 (mute) to digital audio output 16

`[EA,08,01]`

Routes analog audio input 08 to analog audio output 01

The serial port will not echo characters back to the sender. Instead once the command is completed, the response will be sent back as shown in the Query section below.

Setup and Misc commands

The following commands are stored in EEPROM memory, so they will still be set after a power outage.

```
[S,x,y]
```

S = Setup function
x=function letter, y=argument

L=LED lights on the front panel
0=blackout

1=normal state (default)

R=Front IR receiver
0=disabled

1=enabled (default)

B=Front panel buttons
0=disabled

1=enabled (default)

V=Verbosity
0=responds only when sent a command

1=any change in state from front, IR, or other ports are sent out like queries (default)

Setup Example

```
[S,R,0]
```

will disable the front IR receiver

Power Control

```
[P,n]
```

P=power control

0=power off

1=power on

The response to these commands is the same as the command itself i.e. sending

```
[P,1]
```

will get a response of

```
[P,1]
```

Query commands

```
[?E]
```

Putting a “?” inside the bracket, followed by E will return the full state of the matrix.

Example response to [?E] on an Eureka matrix will look like:

```
[[ED,01,01][ED,01,02][ED,01,03][ED,01,04][ED
,01,05][ED,01,06][ED,01,07][ED,01,08][ED,01,
09][ED,04,10][ED,01,11][ED,04,12][ED,01,13][
ED,01,14][ED,01,15][ED,03,16][EA,01,01][EA,0
1,02][EA,01,03][EA,01,04][EA,01,05][EA,01,06
][EA,01,07][EA,01,08][EA,01,09][EA,04,10][EA
,01,11][EA,04,12][EA,01,13][EA,01,14][EA,01,
15][EA,03,16]]
```

Note that there is another level of braces that wraps the entire query response.

Send:

```
[?V]
```

Returns product ID and version

```
[V,E10]
```

E= Eureka

10= major and minor version, or v1.0[7]

Send:

```
[?S]
```

Returns all setup parameters. i.e.:

```
[[S,L,1][S,R,1][S,B,1][S,V,1]]
```

Send:

```
[?P]
```

Returns power state

```
[P,1]
```

[7] The major and minor version is more referring to the serial protocol version. If there are any changes in the protocol, the at least the minor version will increment.

Fallbrook Protocol

Structure prototype:

[FV,XX,YY]

FV = Fallbrook Video

XX= the input number to route from

YY= the two digit output number to route to

XX Input

00 = The ‘mute’ or disable selection

01-08 = Source inputs 1 through 8, this field shall always use two characters.

YY Output

01-16 = Outputs 1 through 16, this field shall always use two characters.

Examples

```
[FV,03,01]
```

Routes input 3 to output 1

```
[FV,00,16]
```

Routes input 00 (mute) to output 16

The serial port will not echo characters back to the sender. Instead once the command is completed, the response will be sent back as shown in the Query section below.

Setup and Misc commands

The following commands are stored in EEPROM memory, so they will still be set after a power outage.

```
[S,x,y]
```

S = Setup function
x=function letter, y=argument

L=LED lights on the front panel
0=blackout

1=normal state (default)

R=Front IR receiver
0=disabled

1=enabled (default)

B=Front panel buttons
0=disabled

1=enabled (default)

V=Verbosity
0=responds only when sent a command

1=any change in state from front, IR, or other ports are sent out like queries (default)

Setup Example

```
[S,R,0]
```

will disable the front IR receiver

Power Control

```
[P,n]
```

P=power control

0=power off

1=power on

The response to these commands is the same as the command itself i.e. sending

```
[P,1]
```

will get a response of

```
[P,1]
```

Query commands

```
[?F]
```

Putting a “?” inside the bracket, followed by F will return the full state of the matrix.

Example response to [?F] on an Fallbrook matrix will look like:

```
[[FV,01,01][FV,01,02][FV,01,03][FV,04,04][FV,01,05][FV,00,06][FV,07,07][FV,02,08][FV,01,09][FV,03,10][FV,00,11][FV,01,12][FV,04,13][FV,08,14][FV,00,15][FV,00,16]]
```

Note that there is another level of braces that wraps the entire query response.

Send:

```
[?V]
```

Returns product ID and version

```
[V,F10]
```

F= Fallbrook

10= major and minor version, or v1.0[8]

[8] The major and minor version is more referring to the serial protocol version. If there are any changes in the protocol, the at least the minor version will increment.

Send:

```
[?S]
```

Returns all setup parameters. i.e.:

```
[[S,L,1][S,R,1][S,B,1][S,V,1]]
```

Send:

```
[?P]
```

Returns power state

```
[P,1]
```

Gillespie Protocol

Structure prototype:

`[GX,XX,YY]`

GD = Gillespie digital audio matrix

GA = Gillespie analog audio matrix

G0 = Both digital and analog audio matrices

XX= the input number to route from

YY= the two digit output number to route to

XX Input

00 = The ‘mute’ or disable selection

01-08 = Source inputs 1 through 8, this field shall always use two characters.

YY Output

01-16 = Outputs 1 through 16, this field shall always use two characters.

Examples

`[G0,03,01]`

Routes input 3 to output 1 for both digital and analog audio

`[GD,00,16]`

Routes input 00 (mute) to digital audio output 16

`[GA,08,01]`

Routes analog audio input 08 to analog audio output 01

The serial port will not echo characters back to the sender. Instead once the command is completed, the response will be sent back as shown in the Query section below.

Setup and Misc commands

The following commands are stored in EEPROM memory, so they will still be set after a power outage.

```
[S,x,y]
```

S = Setup function
x=function letter, y=argument

L=LED lights on the front panel
0=blackout

1=normal state (default)

R=Front IR receiver
0=disabled

1=enabled (default)

B=Front panel buttons
0=disabled

1=enabled (default)

V=Verbosity
0=responds only when sent a command

1=any change in state from front, IR, or other ports are sent out like queries (default)

Setup Example

```
[S,R,0]
```

will disable the front IR receiver

Power Control

```
[P,n]
```

P=power control

0=power off

1=power on

The response to these commands is the same as the command itself i.e. sending

```
[P,1]
```

will get a response of

```
[P,1]
```

Query commands

```
[?G]
```

Putting a “?” inside the bracket, followed by G will return the full state of the matrix.

Example response to [?G] on an Gillespie matrix will look like:

```
[[GD,01,01][GD,01,02][GD,01,03][GD,01,04][GD
,01,05][GD,01,06][GD,01,07][GD,01,08][GD,01,
09][GD,04,10][GD,01,11][GD,04,12][GD,01,13][
GD,01,14][GD,01,15][GD,03,16][GA,01,01][GA,0
1,02][GA,01,03][GA,01,04][GA,01,05][GA,01,06
][GA,01,07][GA,01,08][GA,01,09][GA,04,10][GA
,01,11][GA,04,12][GA,01,13][GA,01,14][GA,01,
15][GA,03,16]]
```

Note that there is another level of braces that wraps the entire query response.

Send:

```
[?V]
```

Returns product ID and version

```
[V,G10]
```

G= Gillespie

10= major and minor version, or v1.0[9]

Send:

```
[?S]
```

Returns all setup parameters. i.e.:

```
[[S,L,1][S,R,1][S,B,1][S,V,1]]
```

Send:

```
[?P]
```

Returns power state

```
[P,1]
```

[9] The major and minor version is more referring to the serial protocol version. If there are any changes in the protocol, the at least the minor version will increment.

Hawthorne Protocol

Structure prototype:

[HV,XX,YY]

HV = Hawthorne Video

XX= the input number to route from

YY= the two digit output number to route to

XX Input

00 = The ‘mute’ or disable selection

01-08 = Source inputs 1 through 8, this field shall always use two characters.

YY Output

01-08 = Outputs 1 through 16, this field shall always use two characters.

Examples

```
[HV,03,01]
```

Routes input 3 to output 1

```
[HV,00,08]
```

Routes input 00 (mute) to output 8

The serial port will not echo characters back to the sender. Instead once the command is completed, the response will be sent back as shown in the Query section below.

Setup and Misc commands

The following commands are stored in EEPROM memory, so they will still be set after a power outage.

```
[S,x,y]
```

S = Setup function
x=function letter, y=argument

L=LED lights on the front panel
0=blackout

1=normal state (default)

R=Front IR receiver
0=disabled

1=enabled (default)

B=Front panel buttons
0=disabled

1=enabled (default)

V=Verbosity
0=responds only when sent a command

1=any change in state from front, IR, or other ports are sent out like queries (default)

Setup Example

```
[S,R,0]
```

will disable the front IR receiver

Power Control

```
[P,n]
```

P=power control

0=power off

1=power on

The response to these commands is the same as the command itself i.e. sending

```
[P,1]
```

will get a response of

```
[P,1]
```

Query commands

```
[?H]
```

Putting a “?” inside the bracket, followed by H will return the full state of the matrix.

Example response to [?H] on an Hawthorne matrix will look like:

```
[[HV,01,01][HV,01,02][ HV,01,03][ HV,04,04][
HV,01,05][ HV,00,06][ HV,07,07][ HV,02,08]]
```

Note that there is another level of braces that wraps the entire query response.

Send:

```
[?V]
```

Returns product ID and version

```
[V,H10]
```

H= Hawthorne

10= major and minor version, or v1.0[10]

[10] The major and minor version is more referring to the serial protocol version. If there are any changes in the protocol, the at least the minor version will increment.

Send:

```
[?S]
```

Returns all setup parameters. i.e.:

```
[[S,L,1][S,R,1][S,B,1][S,V,1]]
```

Send:

```
[?P]
```

Returns power state

```
[P,1]
```

Imperial Protocol

Structure prototype:

`[IV,XX,YY]`

IV = Imperial Video

XX= the input number to route from

YY= the two digit output number to route to

XX Input

00 = The ‘mute’ or disable selection

01-08 = Source inputs 1 through 8, this field shall always use two characters.

YY Output

01-16 = Outputs 1 through 16, this field shall always use two characters.

Examples

```
[IV,03,01]
```

Routes input 3 to output 1

```
[IV,00,08]
```

Routes input 00 (mute) to output 8

The serial port will not echo characters back to the sender. Instead once the command is completed, the response will be sent back as shown in the Query section below.

Setup and Misc commands

The following commands are stored in EEPROM memory, so they will still be set after a power outage.

```
[S,x,y]
```

S = Setup function
x=function letter, y=argument

L=LED lights on the front panel
0=blackout

1=normal state (default)

R=Front IR receiver
0=disabled

1=enabled (default)

B=Front panel buttons
0=disabled

1=enabled (default)

V=Verbosity
0=responds only when sent a command

1=any change in state from front, IR, or other ports are sent out like queries (default)

Setup Example

```
[S,R,0]
```

will disable the front IR receiver

Power Control

```
[P,n]
```

P=power control

0=power off

1=power on

The response to these commands is the same as the command itself i.e. sending

```
[P,1]
```

will get a response of

```
[P,1]
```

Query commands

```
[?I]
```

Putting a “?” inside the bracket, followed by I will return the full state of the matrix.

Example response to [?I] on an Imperial matrix will look like:

```
[[IV,01,01][IV,01,02][ IV,01,03][ IV,04,04][
IV,01,05][ IV,00,06][ IV,07,07][ IV,02,08]]
```

Note that there is another level of braces that wraps the entire query response.

Send:

```
[?V]
```

Returns product ID and version

```
[V,I10]
```

`I= Imperial`

10= major and minor version, or v1.0[11]

[11] The major and minor version is more referring to the serial protocol version. If there are any changes in the protocol, the at least the minor version will increment.

Send:

```
[?S]
```

Returns all setup parameters. i.e.:

```
[[S,L,1][S,R,1][S,B,1][S,V,1]]
```

Send:

```
[?P]
```

Returns power state

```
[P,1]
```

USB Driver Installation

When using the USB port, the matrix switch will be installed as a virtual COM port. This means that any control program capable of controlling a device through a normal serial port should be able to control the matrix through a USB port.

This driver set is for all versions of MS Windows. Linux and Mac drivers can be made available upon request.

The following process is for Windows XP, but other versions of Windows will be very similar.

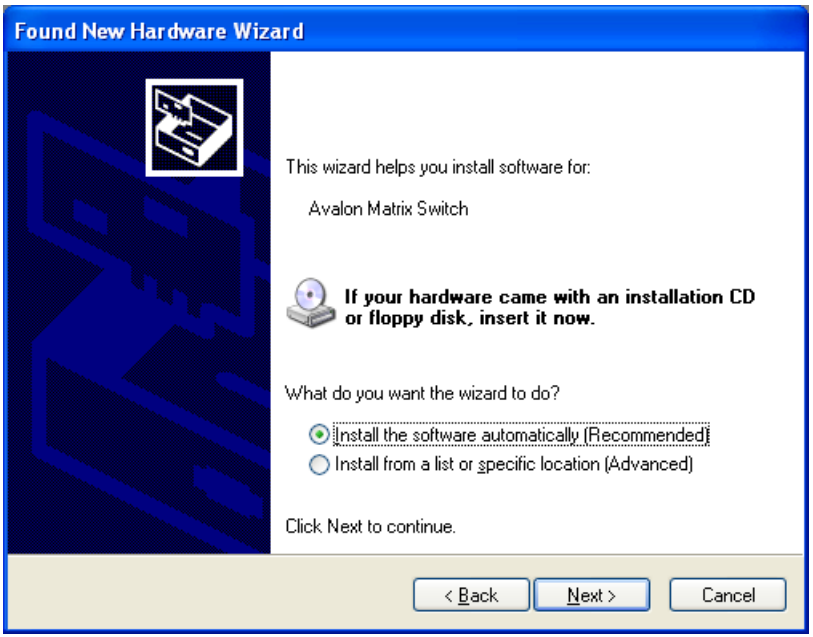
Step 1 – Connect the cable

Windows will detect the new hardware, and launch the plug and play wizard.

Step 2 – Found New Hardware Wizard

The first window will attempt to use the internet to find the driver, Select "**No, not at this time**", and click **Next**.

Step 3 – Driver location



The next window attempts to find the driver disc. Insert the driver CD disc in the your CD-ROM drive if you haven't already.

Leave the button labeled "**Install the software automatically (recommended)**" selected, and click **Next**.

Step 4 – Continue Anyway

Windows will prompt on logo testing. Click "**Continue Anyway**"

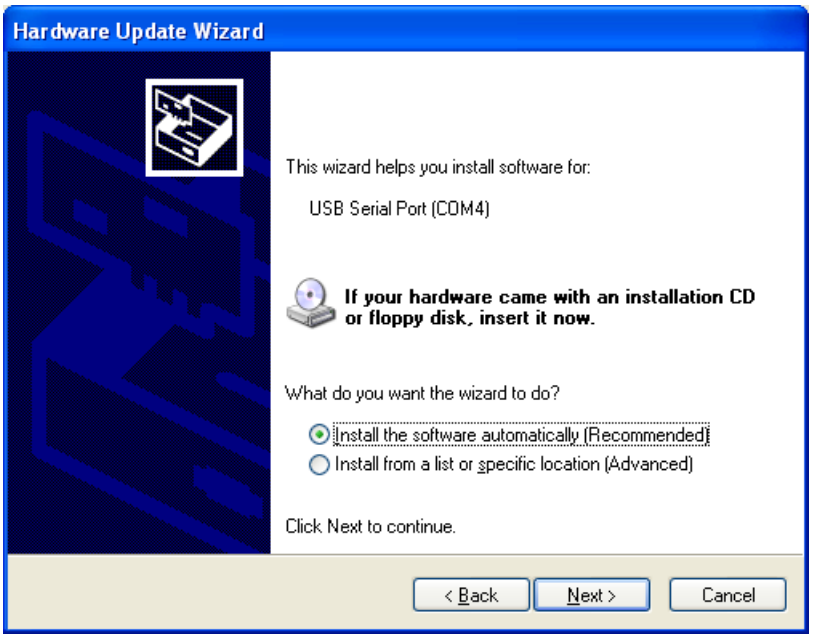
Step 5 – Completion of first half

Click "**Finish**".

Step 6 – Installing virtual COM port driver

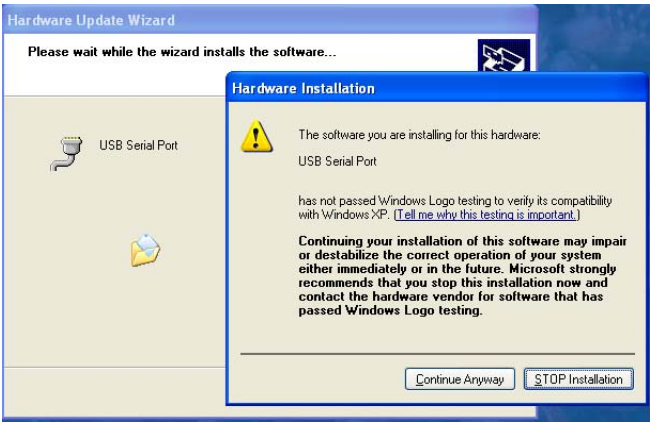
Windows will again start the new hardware wizard to install the virtual COM port driver. Click "**No, not this time**", then click **Next**.

Step 7 – Finding the driver



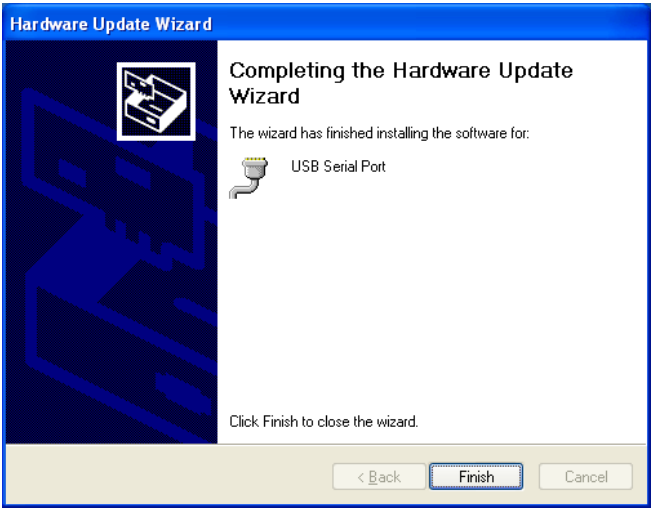
Leave the button labeled "**Install the software automatically (recommended)**" selected, and click **Next**.

Step 8 – Continue Anyway



Windows will prompt on logo testing. Click "**Continue Anyway**"

Step 9 – Completing the Hardware Update Wizard

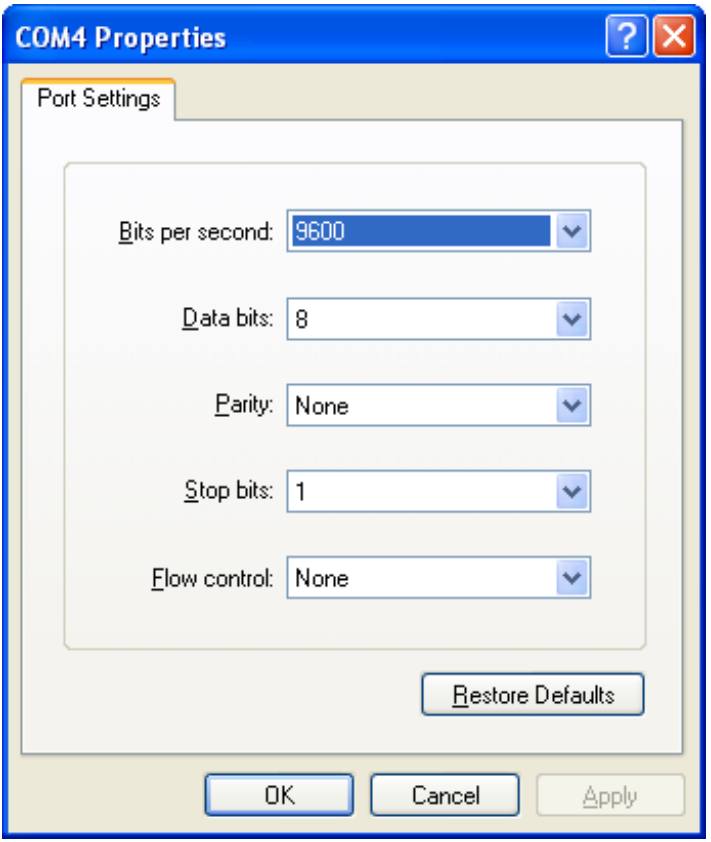


This is the final step, click **Finish**

At this point, the drivers are installed properly.

USB COM Port Settings

Whether using a Windows terminal program such as Hyperterminal, a control application, or a dedicated control systems, the baud rate settings are the same: 9600 baud, 8 data, no parity, 1 stop, no flow control. The COM port shown in the following example may change depending on your system.



Uninstalling the USB drivers

There is typically no harm in leaving the drivers installed in Windows. It is usually desired to keep them installed so that when the device is plugged back in, it will be recognized automatically and is assigned the same COM port number.

However, if you need to uninstall the drivers for any reason, use the Windows Control panel to do so.

To Uninstall:

Click the **Start** Menu

Select **Settings**, then **Control Panel**

Click **Add or Remove Programs**

Find **Matrix Switch USB Drivers**

Click **Change /Remove**

Follow the on screen instructions.

Notes

Notes

www.neoprointegrator.com