This document provides additional assistance with wiring your Extron IP Link Pro Control Processor to your device. Different components may require a different wiring scheme than those listed below.

For complete operating instructions, refer to the user's manual for the specific IP Link Pro Control Processor or the documentation supplied by the manufacturer of the controlled device.

For more information on using Global Scripter Modules, refer to the "Guide to Using Scripter Modules" document.

Device Specifications

Device Type:	Audio Processor
Manufacturer:	QSC
Firmware Version:	7.2.1
Model(s):	Q-Sys Core 250i, Q-Sys Core 500i, Q-Sys Core 1000, Q-Sys Core 1100, Q-Sys Core 3000, Q-Sys Core 4000, Q-Sys Core 3100, Q-Sys Core 110f, Q-Sys Core 510i, Q-Sys Cinema Core 110c

Tested on the Following Software and Firmware Versions

IP Link Pro Control Processor Firmware	Global Scripter Version
3.09.0000-b009	2.4.2

Version History

Module Version	Date	Notes
1_12_4_0	7/10/2020	Changed range of Control Set Value to 0-100.
1_12_1_0	10/22/2019	Updated Gain Value Range. Fixed Design Name status. Added Call History Result Set. Added the following Commands: - PhonebookControl - PhonebookListUpdate - PhonebookNavigation - PhonebookResult Set - PhonebookResults - PhonebookSearch - PhonebookSelected - PhonebookUpdate

		- CallHistoryResultSet.
1_10_4_1	8/19/2019	Fixed DesignName status.
1_10_4_0	10/9/2018	Updated module to revision B1.
1_10_3_0	8/21/2018	Added model: Q-Sys Cinema Core 110c, Added Status command: Level Meter.
1_10_0_1	5/10/2018	Fixed Call History navigation. Added missing command GetStatusString
1_10_0_0	4/17/2018	Updated Camera router command to Router command, to account for audio routing functionality. Added status to Router command. Added Camera Control and Call History commands. Fixed Snapshot Load Command. Added Serial Control.
1_6_5_0	4/4/2017	Added Snapshot Save, Snapshot Load, updated module, added model Q-Sys Core 510i
1_3_0_0	9/12/2016	Update to standard. Fixed Mute.
1_1_0_0	8/24/2016	Initial Version

Module Notes

- Unidirectional variable must be set to 'True' if status is not required. Default value is 'False'.
  `Example: InterfaceName.Unidirectional = 'True'`

- connectionCounter variable must be set to the number of queries that will be sent to the device before displaying 'Disconnected' if no response is received. Default value is 15.
  `Example: InterfaceName.connectionCounter = 5`

- If login credentials are required, devicePassword and deviceUsername must be set accordingly.
  `Example: InterfaceName.devicePassword = 'extron'`

- NumberofCallHistoryResults variable must be set accordingly. Default value is '5'. NumberofCallHistoryResults ranges from 1 to 10.
  `Example: InterfaceName.NumberofCallHistoryResults = '5'`

- NumberofPhonebookResults variable must be set accordingly. Default value is '5'. NumberofPhonebookResults ranges from 1 to 10.
  `Example: InterfaceName.NumberofPhonebookResults = '5'`

- For Serial control, the RS232 port must be enabled through Q-SYS Designer software.

- Please contact QSC for additional information on RS232 setup and obtain the Control ID.

Supported Classes and Examples

SerialClass

```
    InterfaceName = ModuleName.SerialClass(ProcessorName, 'COM1', Model='Q-Sys Core 250i')
```

SerialOverEthernetClass

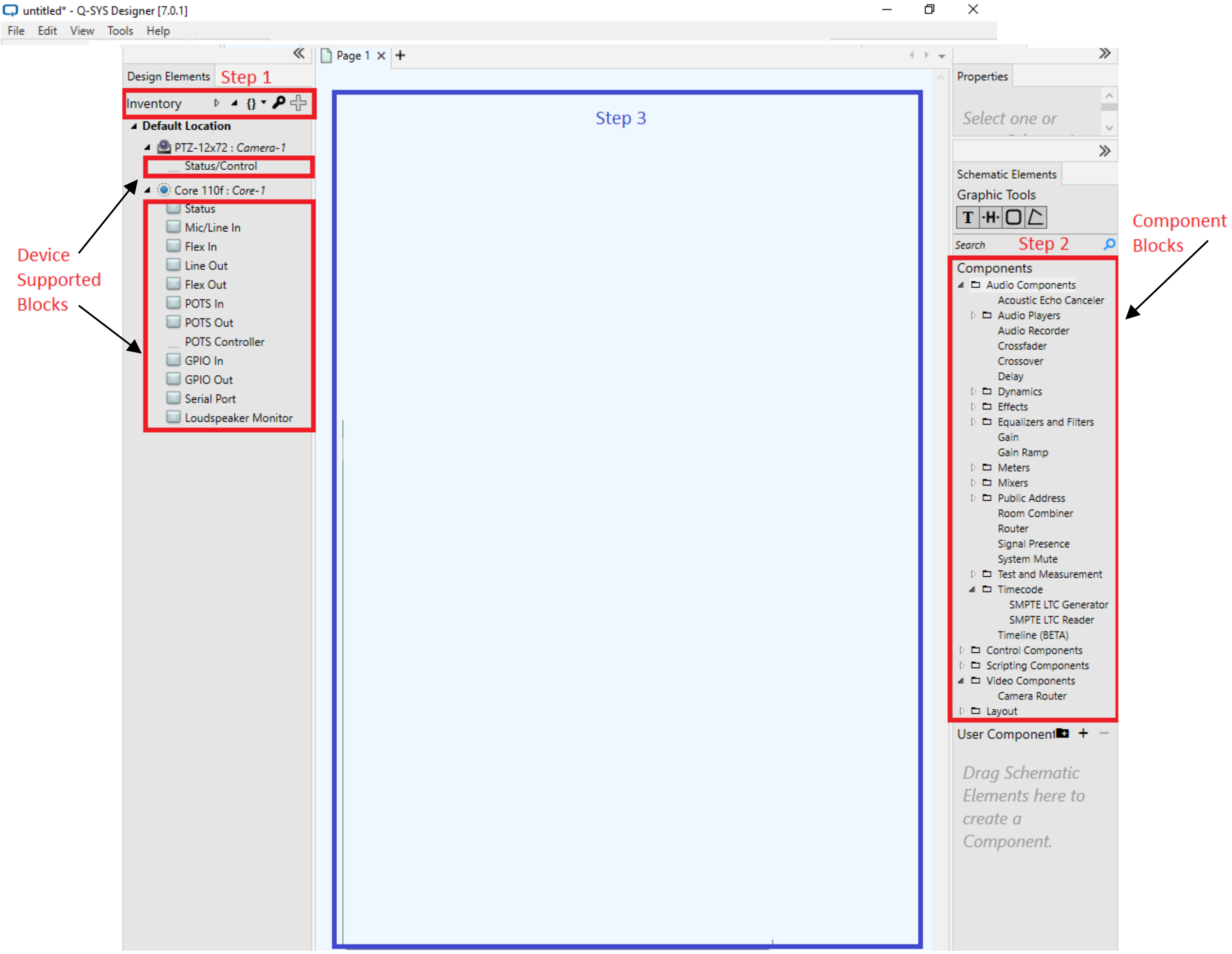
```
    InterfaceName = ModuleName.SerialOverEthernetClass('192.168.254.254', 2001, Model='Q-Sys Core 250i')
```

EthernetClass

```
    InterfaceName = ModuleName.EthernetClass('192.168.254.254', 1702, Model='Q-Sys Core 250i')
```

Q-SYS Designer Software

The Q-SYS Designer Software is used to design configurations for QSC devices. This section goes over the first steps to take to design a configuration using the software. (If you have already designed a Q-SYS configuration, you can skip this section.)



Steps:

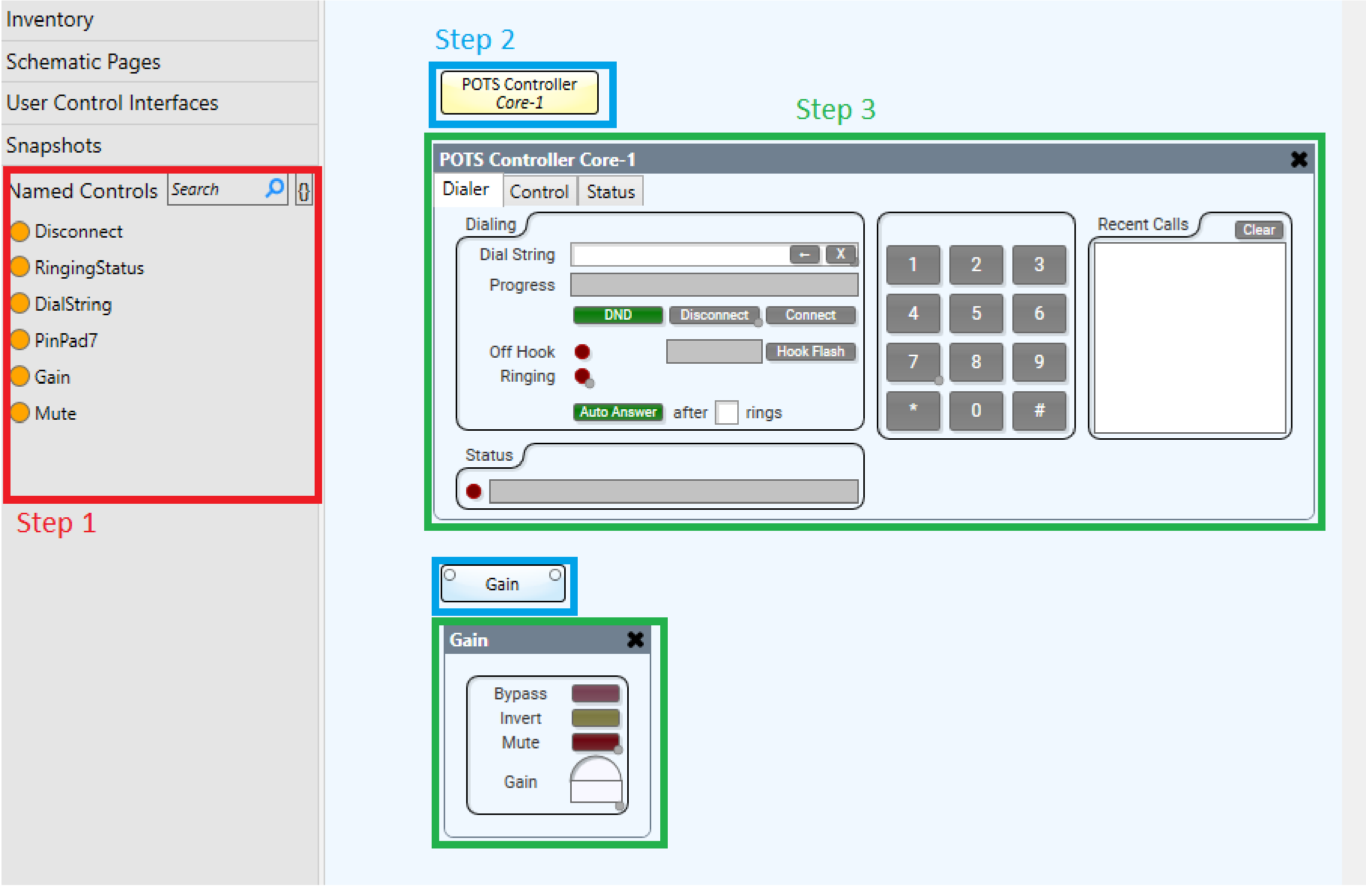
1. Open the Q-SYS Designer software and locate the **Inventory** section. This section is used to add and view the device(s) to be used in the Q-SYS configuration. It also displays the blocks the device(s) supports.

2. Locate the **Components** section. This section is used to add and view the component blocks to be used in the Q-SYS configuration.

3. Once these sections have been located, you can start designing the Q-SYS configuration by dragging Device supported blocks and/or Component blocks into the middle section of the screen, boxed in blue.

Obtaining the Control ID

Q-SYS Software configurations consist of individual blocks that correspond to a specific assigned Control ID. These Control IDs are to be used in this module to properly obtain control functionality. Most commands in this module require a 'Control ID' parameter to be configured.

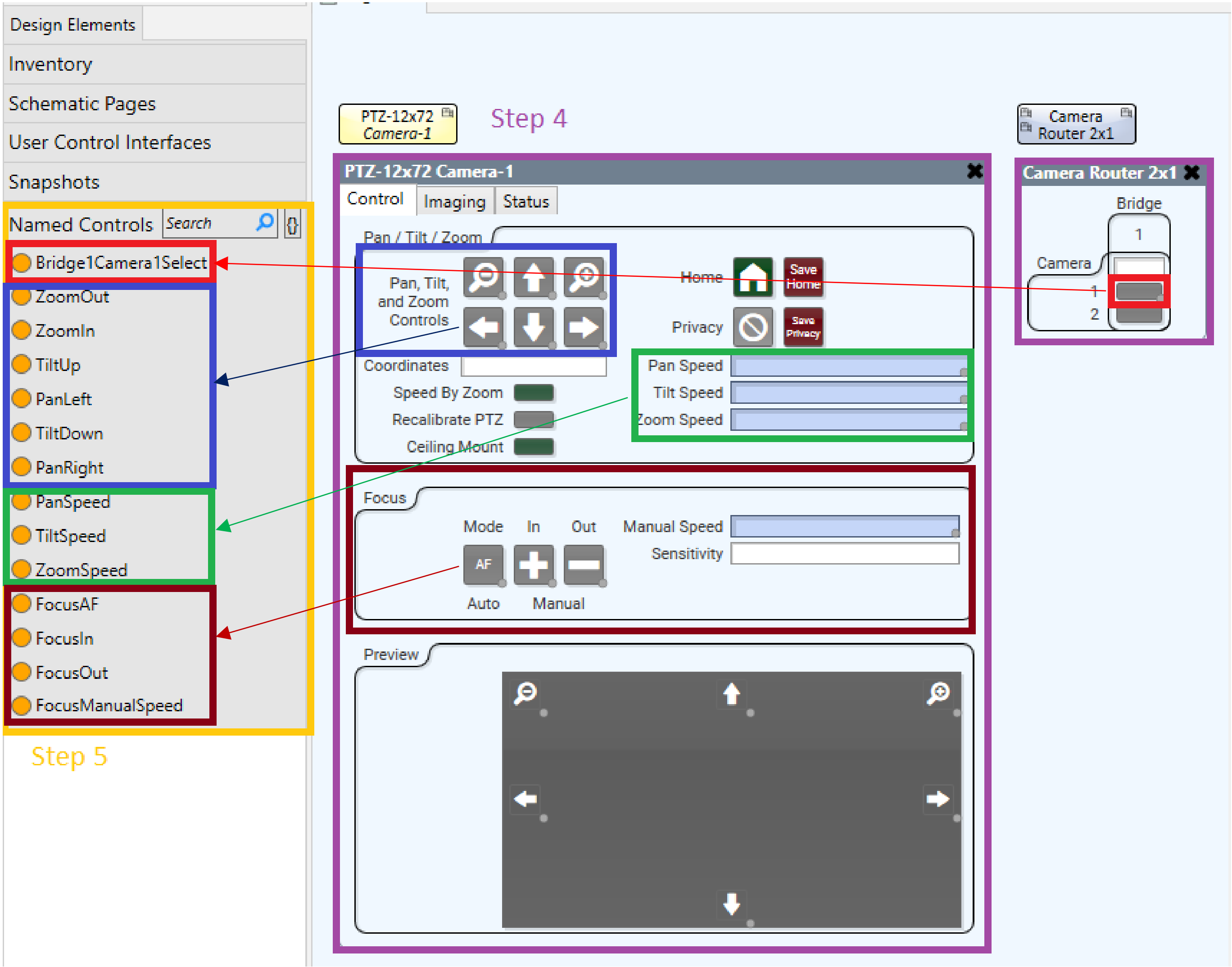


Steps:

1. Locate the **Named Controls** section. This is where the list of Control IDs will be displayed.
2. Select or click on one of the Device supported blocks or Component blocks in your Q-SYS configuration.
3. A window will pop up once you select or click a block. The window will display the control and status functions of the selected block.

Obtaining the Control ID (continued)

Example Q-SYS Configuration:

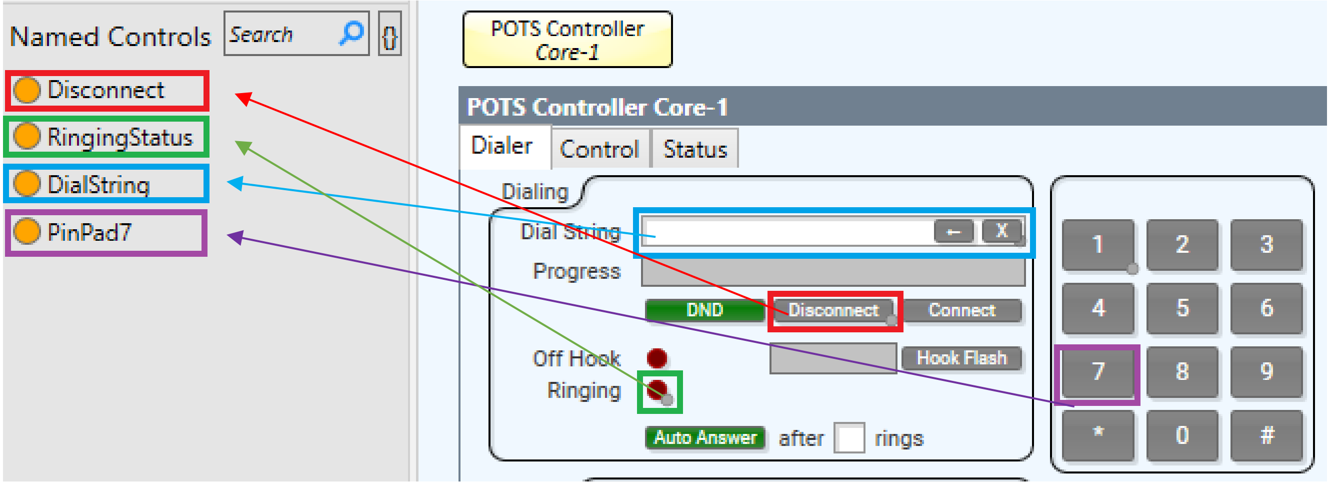


4. Within each window you can click on each of the control and status functions.

5. Select one of the displayed functions and drag it to the **Named Controls** section. The Control ID will then be displayed in this section. (You can customize the Control ID by double clicking on it.)

Configuring Commands

Once you have obtained the Control ID of a function from a specific block, keep in mind what type of function it is. There are different types of functions, which this module defines them as:



- **Control Set Value**

  Single Press functions can be configured by using Control Set Value with Control Value of 1.
  For example, Disconnect is given the Control ID of 'Disconnect' as a single press configure with control value 1.
- **Get Status Value**

  Status functions with 'On' and 'Off' States can be configured as follows using Get Status Value.
  For example, Ringing Status is given Control ID of 'RingingStatus' that will update with the corresponding status 'On' or 'Off'.
- **Get Status String**

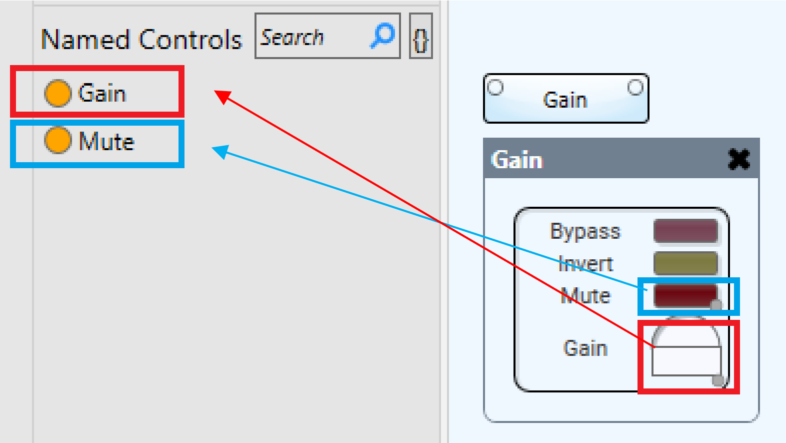
  Status functions with a string can be configured as followed using the Get Status String command.
  For example, Dial String is given Control ID of 'DialString' that will update with the corresponding string.
- **Dial String**

  Keypad functions can be configured by using Dial String with Control Value of 1. Also Dial String ID must be configured to the Control ID that is being used for Dial String status.
  For example, Digit 7 is given the Control ID of 'PinPad7 as a single press configure with control value 1.

Configuring Commands (continued)

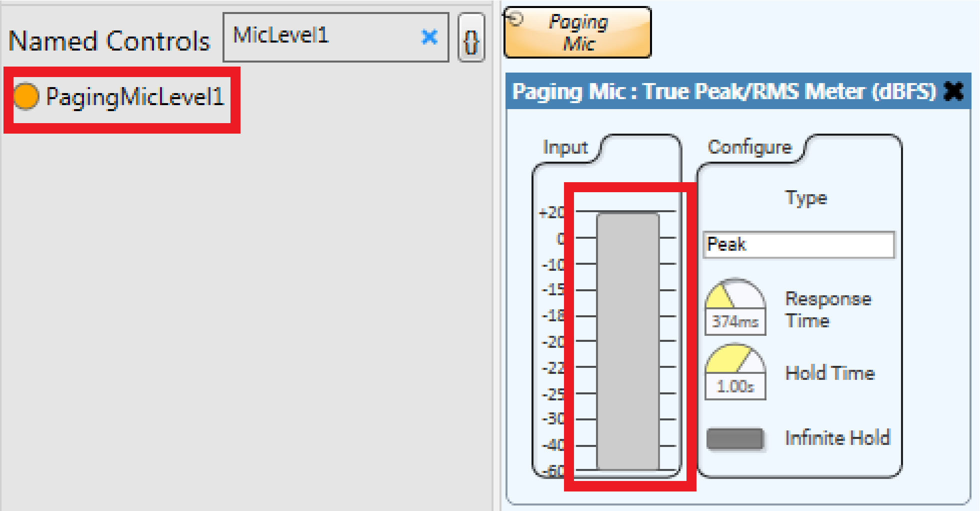
- **Gain/Mute**

Gain/Mute functions like the ones seen below can be configured using the Gain/Mute command with their corresponding Control ID.
For example, in this Gain block the Gain function was given the Control ID of 'Gain' and the Mute function was given the Control ID of 'Mute'.



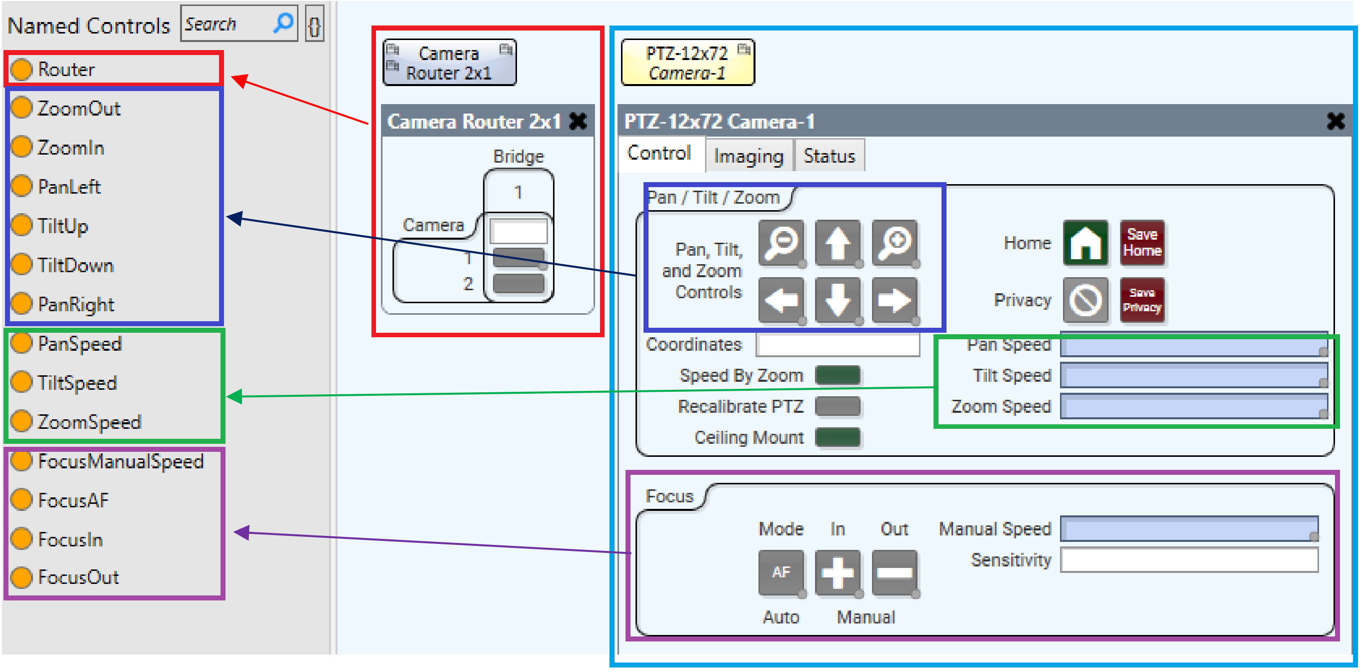
- **Level Meter**

Level Meter function from Meters Component Block like the one seen below can be configured using the Level Meter command with their corresponding Control ID. The Meter Type can be configured as "Peak" or "RMS" and the module will return the Level Meter reading accordingly.
For example, in this "Paging Mic" Level Meter block the Level Meter function was given the Control ID of 'PagingMicLevel1' and configured for "Peak" Type.



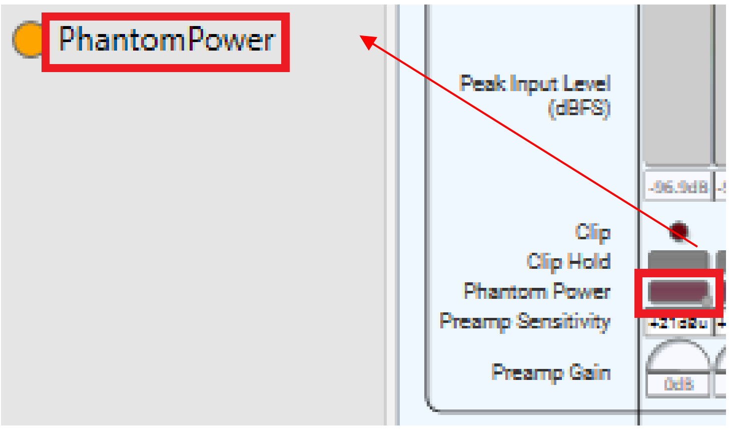
- **Focus/PTZ/Router**

Focus and PTZ commands correspond to the functions in the PTZ Camera block seen below.
Router command corresponds to the functions in the Camera Router block seen below.



- **Function**

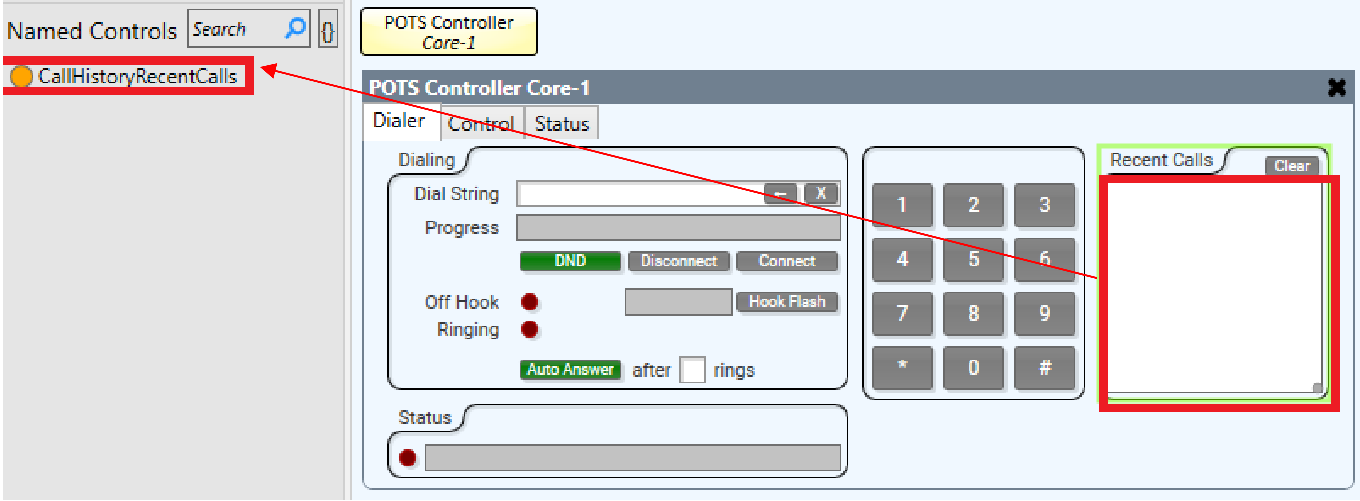
Function commands correspond to any block that functions as an enable / disable.
Example: Phantom Power



Configuring Commands (continued)

- **Call History**

  Call History commands correspond to functions like the Recent Calls function seen below.
  In the image below, the Recent Calls function is given the Control ID of ‘CallHistoryRecentCalls’ to be used as the module parameter for Call History commands.



Call History Commands

- Call History Update
  - This command is used to query the Call History, in which the device will respond with a list of results, which will be populated in the command below.

- Call History Results
  - This is where the results will be saved to when call history is updated. The number of results are set by a module parameter Number of Call History Results, this is initialized to 5 results by default. Simply drag this onto a Text Feedback object for displaying the search result.

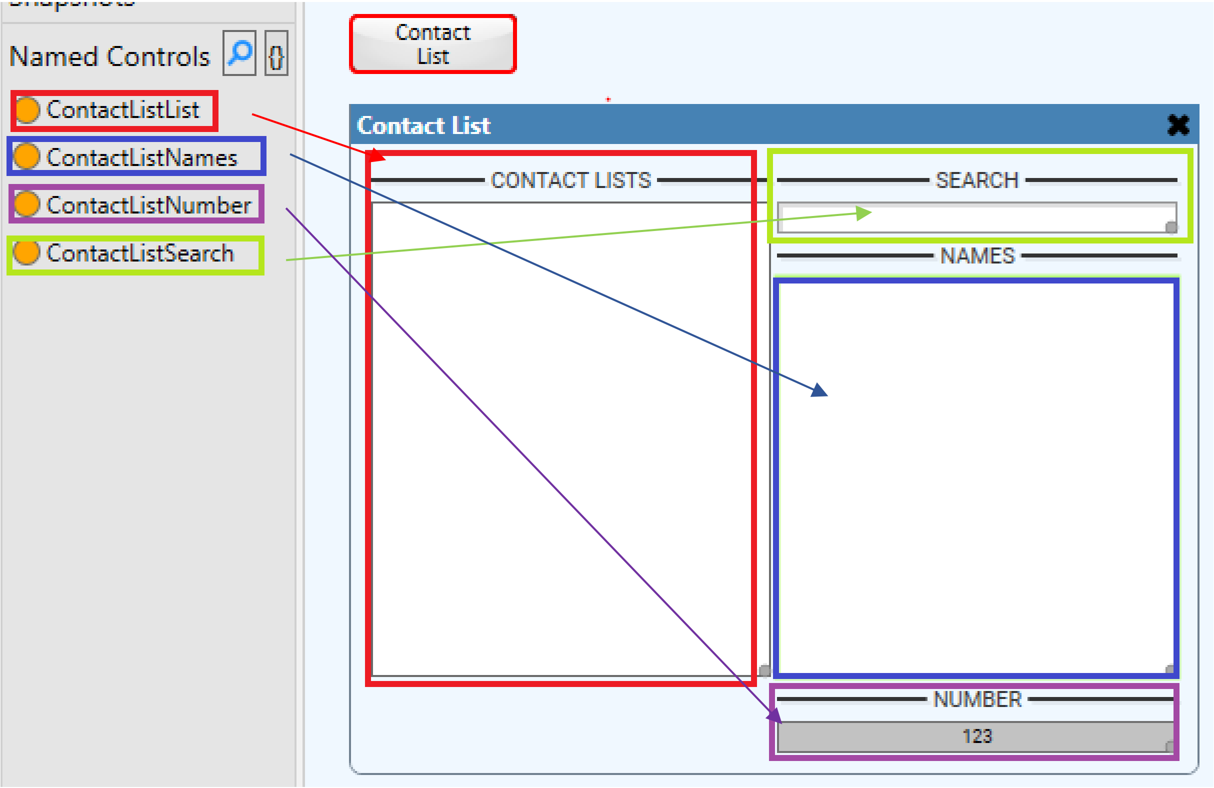
- Call History Result Set
  - This command obtains the result from Call History Results and sends to the Dial String status, to be used for dialing functionality.

- Call History Navigation
  - This command is used for navigating through the search results, it is like scrolling through the results when there are more than the Number of Call History Results that came back from the Call History Update.

Configuring Commands (continued)

- **Phonebook**

Phonebook commands correspond to functions like the Contact Lists, Names, Number, or Search function seen below.
In the image below, the Contact Lists or Names function is given the Control ID of 'Contact Lists' or 'Names' to be used as the module parameter for Phonebook Control, Phonebook List Update, Phonebook Navigation, Phonebook Results, Phonebook Result Set, and Phonebook Update. The Number function is given the Control ID of Number to be used as the Phonebook ID parameter for Phonebook Control. The Search function is given the Control ID of Search to be used as the module parameter for Phonebook Search.



Phonebook Commands

- Phonebook Control
    - This command will perform an action on the Phonebook Selected based on the Control ID assigned to this command.

- Phonebook List Update
    - This command is used to query the Contact Lists, in which the device will respond with a list of results, which will be populated in the Phonebook Results command.

- Phonebook Navigation
    - This command is used for navigating through the search results, it is like scrolling through the results when there are more than the Number of Phonebook Results that came back from the Phonebook Update.

- Phonebook Results
    - This is where the results will be saved to when Phonebook is updated. The number of results are set by a driver parameter Number of Phonebook Results, this is initialized to 5 results by default. Simply drag this onto a Text Feedback object for displaying the search result.

- Phonebook Result Set
    - This command will copy the name in the corresponding position from Phonebook Results over to Phonebook Selected.

- Phonebook Search
    - This command will use the search string provided from Phonebook Search String command to search the contact names that would match the string from Phonebook Results.

- Phonebook Selected
    - This command will display the Phonebook result selected by the Phonebook Result Set command.

- Phonebook Update
    - This command is used to query the names of the Contact List, in which the device will respond with a list of results, which will be populated in the Phonebook Results command.

Control Commands

Format with Qualifier:

```
InterfaceName.Set(Command, Value, {'Qualifier Key': 'Qualifier Value'})
```

Format with Qualifier:

```
InterfaceName.Set(Command, Value)
```

Command **CallHistoryNavigation**	Value 'Up' 'Page Down'	Value 'Down'	Value 'Page Up'
Qualifier Key **'Control ID'**	Qualifier Value 'String'		

`# CallHistoryNavigation example`
`InterfaceName.Set('CallHistoryNavigation', 'Up', {'Control ID': 'String'})`			

Command **CallHistoryResultSet**	Value None		
Qualifier Key **'Dial String ID'**	Qualifier Value 'String'		
Qualifier Key **'Control ID'**	Qualifier Value 'String'		
Qualifier Key **'Position'**	Qualifier Value '1' – '10'		

`# CallHistoryResultSet example`
`InterfaceName.Set('CallHistoryResultSet', None, {'Dial String ID': 'String', 'Control ID': 'String', 'Position': '1'})`			

Command **CallHistoryUpdate**	Value None		
Qualifier Key **'Control ID'**	Qualifier Value 'String'		

`# CallHistoryUpdate example`
`InterfaceName.Set('CallHistoryUpdate', None, {'Control ID': 'String'})`			

Command **ControlSetValue**	Value None		
Qualifier Key **'Control ID'**	Qualifier Value 'String'		
Qualifier Key **'Control Value'**	Qualifier Value 0 to 100 in steps of 1		

`# ControlSetValue example`
`InterfaceName.Set('ControlSetValue', None, {'Control ID': 'String', 'Control Value': 100})`			

Command **FocusControl**	Value 'Enable'	Value 'Disable'	
Qualifier Key **'Control ID'**	Qualifier Value 'String'		

`# FocusControl example`
`InterfaceName.Set('FocusControl', 'Enable', {'Control ID': 'String'})`			

Command	Value
FocusSpeed	0.001 to 0 in steps of 0.001
Qualifier Key	Qualifier Value
'Control ID'	'String'

```
    # FocusSpeed example
    InterfaceName.Set('FocusSpeed', 0, {'Control ID': 'String'})
```

Command	Value	Value
Function	'Enable'	'Disable'
Qualifier Key	Qualifier Value	
'Control ID'	'String'	

```
    # Function example
    InterfaceName.Set('Function', 'Enable', {'Control ID': 'String'})
```

Command	Value
Gain	-100 to 83 in steps of 0.1
Qualifier Key	Qualifier Value
'Control ID'	'String'

```
    # Gain example
    InterfaceName.Set('Gain', 83, {'Control ID': 'String'})
```

Command	Value	Value
Mute	'On'	'Off'
Qualifier Key	Qualifier Value	
'Control ID'	'String'	

```
    # Mute example
    InterfaceName.Set('Mute', 'On', {'Control ID': 'String'})
```

Command	Value
PhonebookControl	None
Qualifier Key	Qualifier Value
'Control ID'	'String'
Qualifier Key	Qualifier Value
'Phonebook ID'	'String'

```
    # PhonebookControl example
    InterfaceName.Set('PhonebookControl', None, {'Control ID': 'String', 'Phonebook ID': 'String'})
```

Command	Value
PhonebookListUpdate	None
Qualifier Key	Qualifier Value
'Control ID'	'String'

```
    # PhonebookListUpdate example
    InterfaceName.Set('PhonebookListUpdate', None, {'Control ID': 'String'})
```

Command	Value	Value	Value
PhonebookNavigation	'Up'	'Down'	'Page Up'
	'Page Down'		
Qualifier Key	Qualifier Value		
'Control ID'	'String'		

```
    # PhonebookNavigation example
    InterfaceName.Set('PhonebookNavigation', 'Up', {'Control ID': 'String'})
```

Command	Value
PhonebookResultSet	None
Qualifier Key	Qualifier Value
'Control ID'	'String'
Qualifier Key	Qualifier Value
'Position'	'1' – '10'

```
    # PhonebookResultSet example
    InterfaceName.Set('PhonebookResultSet', None, {'Control ID': 'String', 'Position': '1'})
```

Command **PhonebookSearch**	Value None	
Qualifier Key **'Control ID'**	Qualifier Value 'String'	
Qualifier Key **'Contact'**	Qualifier Value 'String'	

```
    # PhonebookSearch example
    InterfaceName.Set('PhonebookSearch', None, {'Control ID': 'String', 'Contact': 'String'})
```

Command **PhonebookSelected**	Value 'String'	
Qualifier Key **'Control ID'**	Qualifier Value 'String'	

```
    # PhonebookSelected example
    InterfaceName.Set('PhonebookSelected', 'String', {'Control ID': 'String'})
```

Command **PhonebookUpdate**	Value None	
Qualifier Key **'Control ID'**	Qualifier Value 'String'	

```
    # PhonebookUpdate example
    InterfaceName.Set('PhonebookUpdate', None, {'Control ID': 'String'})
```

Command **PTZControl**	Value 'Enable'	Value 'Disable'
Qualifier Key **'Control ID'**	Qualifier Value 'String'	

```
    # PTZControl example
    InterfaceName.Set('PTZControl', 'Enable', {'Control ID': 'String'})
```

Command **PTZSpeed**	Value 0.01 to 1 in steps of 0.01	
Qualifier Key **'Control ID'**	Qualifier Value 'String'	

```
    # PTZSpeed example
    InterfaceName.Set('PTZSpeed', 1, {'Control ID': 'String'})
```

Command **Router**	Value 'Enable'	Value 'Disable'
Qualifier Key **'Control ID'**	Qualifier Value 'String'	

```
    # Router example
    InterfaceName.Set('Router', 'Enable', {'Control ID': 'String'})
```

Command **SnapshotLoad**	Value '1' – '24'	
Qualifier Key **'Load Time'**	Qualifier Value 0 to 60 in steps of 1	
Qualifier Key **'Bank'**	Qualifier Value 'String'	

```
    # SnapshotLoad example
    InterfaceName.Set('SnapshotLoad', '1', {'Load Time': 60, 'Bank': 'String'})
```

Command **SnapshotSave**	Value '1' – '24'	
Qualifier Key **'Bank'**	Qualifier Value 'String'	

```
    # SnapshotSave example
    InterfaceName.Set('SnapshotSave', '1', {'Bank': 'String'})
```

Status Available

For all commands, call Update to receive the latest status. ConnectionStatus, CallHistoryResults, and PhonebookResults do not support the Update function. ConnectionStatus is triggered by the device providing a successful response to other Update function calls.

Format with Qualifier:

```
        InterfaceName.Update(Command, {'Qualifier Key': 'Qualifier Value'})
        Value = InterfaceName.ReadStatus(Command, {'Qualifier Key': 'Qualifier Value'})
        InterfaceName.SubscribeStatus(Command, {'Qualifier Key': 'Qualifier Value'}, 
FeedbackHandler)
        FeedbackHandler will be called only when the specified qualifier gets a new status.
```

Format without Qualifier:

```
        InterfaceName.Update(Command)
        Value = InterfaceName.ReadStatus(Command)
        InterfaceName.SubscribeStatus(Command, None, FeedbackHandler)
        FeedbackHandler will be called when any qualifier gets a new status.
```

Command	Value
CallHistoryResults	'String'
Qualifier Key	Qualifier Value
'Control ID'	'String'
Qualifier Key	Qualifier Value
'Position'	'1' – '10'

```
    # CallHistoryResults example
    Value = InterfaceName.ReadStatus('CallHistoryResults', {'Control ID': 'String', 'Position': 
'1'})
    InterfaceName.SubscribeStatus('CallHistoryResults', None, FeedbackHandler)
```

Command	Value
CallStatus	'String'
Qualifier Key	Qualifier Value
'Control ID'	'String'

```
    # CallStatus example
    InterfaceName.Update('CallStatus', {'Control ID': 'String'})
    Value = InterfaceName.ReadStatus('CallStatus', {'Control ID': 'String'})
    InterfaceName.SubscribeStatus('CallStatus', None, FeedbackHandler)
```

Command	Value
CallerID	'String'
Qualifier Key	Qualifier Value
'Control ID'	'String'

```
    # CallerID example
    InterfaceName.Update('CallerID', {'Control ID': 'String'})
    Value = InterfaceName.ReadStatus('CallerID', {'Control ID': 'String'})
    InterfaceName.SubscribeStatus('CallerID', None, FeedbackHandler)
```

Command	Value
CallerName	'String'
Qualifier Key	Qualifier Value
'Control ID'	'String'

```
    # CallerName example
    InterfaceName.Update('CallerName', {'Control ID': 'String'})
    Value = InterfaceName.ReadStatus('CallerName', {'Control ID': 'String'})
    InterfaceName.SubscribeStatus('CallerName', None, FeedbackHandler)
```

Command **ConnectionStatus**	Value 'Connected'	Value 'Disconnected'

```
    # ConnectionStatus example
    Value = InterfaceName.ReadStatus('ConnectionStatus')
    InterfaceName.SubscribeStatus('ConnectionStatus', None, FeedbackHandler)
```

Command **DesignName**	Value 'String'

```
    # DesignName example
    InterfaceName.Update('DesignName')
    Value = InterfaceName.ReadStatus('DesignName')
    InterfaceName.SubscribeStatus('DesignName', None, FeedbackHandler)
```

Command **FocusMode**	Value 'Auto'	Value 'Manual'
Qualifier Key **'Control ID'**	Qualifier Value 'String'	

```
    # FocusMode example
    InterfaceName.Update('FocusMode', {'Control ID': 'String'})
    Value = InterfaceName.ReadStatus('FocusMode', {'Control ID': 'String'})
    InterfaceName.SubscribeStatus('FocusMode', None, FeedbackHandler)
```

Command **FocusSpeed**	Value 0.001 to 0 in steps of 0.001
Qualifier Key **'Control ID'**	Qualifier Value 'String'

```
    # FocusSpeed example
    InterfaceName.Update('FocusSpeed', {'Control ID': 'String'})
    Value = InterfaceName.ReadStatus('FocusSpeed', {'Control ID': 'String'})
    InterfaceName.SubscribeStatus('FocusSpeed', None, FeedbackHandler)
```

Command **Function**	Value 'Enable'	Value 'Disable'
Qualifier Key **'Control ID'**	Qualifier Value 'String'	

```
    # Function example
    InterfaceName.Update('Function', {'Control ID': 'String'})
    Value = InterfaceName.ReadStatus('Function', {'Control ID': 'String'})
    InterfaceName.SubscribeStatus('Function', None, FeedbackHandler)
```

Command **Gain**	Value -100 to 83 in steps of 0.1
Qualifier Key **'Control ID'**	Qualifier Value 'String'

```
    # Gain example
    InterfaceName.Update('Gain', {'Control ID': 'String'})
    Value = InterfaceName.ReadStatus('Gain', {'Control ID': 'String'})
    InterfaceName.SubscribeStatus('Gain', None, FeedbackHandler)
```

Command **GetStatusValue**	Value 'On'	Value 'Off'
Qualifier Key **'Control ID'**	Qualifier Value 'String'	

```
    # GetStatusValue example
    InterfaceName.Update('GetStatusValue', {'Control ID': 'String'})
    Value = InterfaceName.ReadStatus('GetStatusValue', {'Control ID': 'String'})
    InterfaceName.SubscribeStatus('GetStatusValue', None, FeedbackHandler)
```

Command	Value	
LevelMeter	-120 – 20	
Qualifier Key	Qualifier Value	
'Control ID'	'String'	

```
    # LevelMeter example
    InterfaceName.Update('LevelMeter', {'Control ID': 'String'})
    Value = InterfaceName.ReadStatus('LevelMeter', {'Control ID': 'String'})
    InterfaceName.SubscribeStatus('LevelMeter', None, FeedbackHandler)
```

Command	Value	Value
Mute	'On'	'Off'
Qualifier Key	Qualifier Value	
'Control ID'	'String'	

```
    # Mute example
    InterfaceName.Update('Mute', {'Control ID': 'String'})
    Value = InterfaceName.ReadStatus('Mute', {'Control ID': 'String'})
    InterfaceName.SubscribeStatus('Mute', None, FeedbackHandler)
```

Command	Value	
PhonebookResults	'String'	
Qualifier Key	Qualifier Value	
'Control ID'	'String'	
Qualifier Key	Qualifier Value	
'Position'	'1' – '10'	

```
    # PhonebookResults example
    Value = InterfaceName.ReadStatus('PhonebookResults', {'Control ID': 'String', 'Position': 
'1'})
    InterfaceName.SubscribeStatus('PhonebookResults', None, FeedbackHandler)
```

Command	Value	
PTZSpeed	0.01 to 1 in steps of 0.01	
Qualifier Key	Qualifier Value	
'Control ID'	'String'	

```
    # PTZSpeed example
    InterfaceName.Update('PTZSpeed', {'Control ID': 'String'})
    Value = InterfaceName.ReadStatus('PTZSpeed', {'Control ID': 'String'})
    InterfaceName.SubscribeStatus('PTZSpeed', None, FeedbackHandler)
```

Command	Value	Value
Router	'Enable'	'Disable'
Qualifier Key	Qualifier Value	
'Control ID'	'String'	

```
    # Router example
    InterfaceName.Update('Router', {'Control ID': 'String'})
    Value = InterfaceName.ReadStatus('Router', {'Control ID': 'String'})
    InterfaceName.SubscribeStatus('Router', None, FeedbackHandler)
```

Cable and Adapter Requirements

For models Q-Sys Core 110f and Cinema Core 110c: Captive Screw to 3-pin 5mm Euro connector Serial Cable.
For all other models: Captive Screw to Female DB9 RS-232 Serial Cable.

Notes for the Device

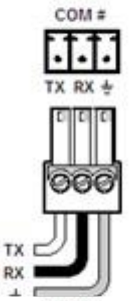
Configure RS-232 settings on the device using the Q-SYS Designer software.

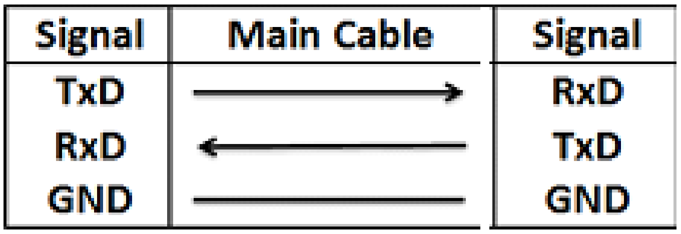
Serial communication

Port Type:	RS-232	**Parity:**	None
Baud Rate:	115200	**Stop Bits:**	One
Data Bits:	8	**Flow Control:**	None

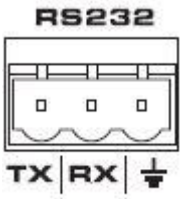
Pin Assignments Diagram

For model Q-Sys Core 110f and Cinema Core 110c:

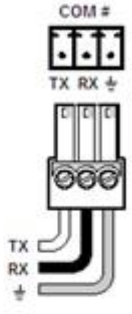


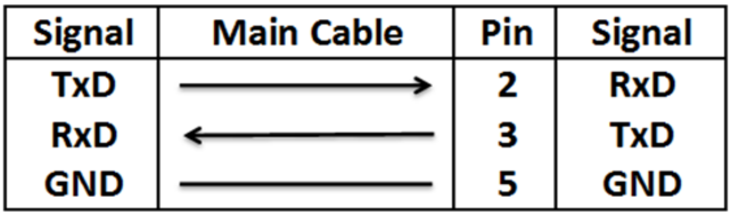


Signal	Main Cable	Signal
TxD	→	RxD
RxD	←	TxD
GND	—	GND

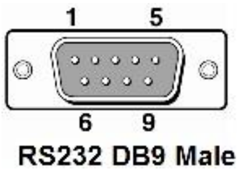


For a dels:





Signal	Main Cable	Pin	Signal
TxD	→	2	RxD
RxD	←	3	TxD
GND	—	5	GND



Network communication

When configuring the Ethernet module, be sure device settings match those of the Global Scripter ethernet interface.

Port Type:	Ethernet
Default Port:	1702
Logon Credentials Supported:	Yes
Multi-Connection Capabilities:	Yes
Port Changeability:	Yes

Ethernet Module Configuration Description

Please refer to user manual for settings and changes to the network communication

Notes for the Device

Appendix A. Set Commands

Control Set Value None Control ID testString Control Value 0	`csv "testString" 0\x0A`
Control Set Value None Control ID testString Control Value 100	`csv "testString" 100\x0A`
Focus Control Disable Control ID testString	`csv "testString" 0\x0A`
Focus Control Enable Control ID testString	`csv "testString" 1\x0A`
Focus Speed 0 Control ID testString	`csv "testString" 0\x0A`
Function Disable Control ID testString	`csv "testString" 0\x0A`
Function Enable Control ID testString	`csv "testString" 1\x0A`
Gain -100 Control ID testString	`csv "testString" -100\x0A`
Gain 83 Control ID testString	`csv "testString" 83\x0A`
Mute Off Control ID testString	`csv "testString" 0\x0A`
Mute On Control ID testString	`csv "testString" 1\x0A`
Phonebook Control None Control ID testString Phonebook ID testString	`css "testString" "string"\x0A`
Phonebook Result Set None Control ID testString Position 1	`css "testString" "string"\x0A`
Phonebook Result Set None Control ID testString Position 10	`css "testString" "string"\x0A`
Phonebook Search None Control ID testString	`css "testString" "string"\x0A`
PTZ Control Disable Control ID testString	`csv "testString" 0\x0A`
PTZ Control Enable Control ID testString	`csv "testString" 1\x0A`
PTZ Speed 0 Control ID testString	`csv "testString" 0\x0A`
PTZ Speed 1 Control ID testString	`csv "testString" 1\x0A`
Router Disable Control ID testString	`csv "testString" 0\x0A`
Router Enable Control ID testString	`csv "testString" 1\x0A`
Snapshot Load 1 Load Time 0 Bank testString	`ssl "testString" 1 0\x0A`
Snapshot Load 1 Load Time 60 Bank testString	`ssl "testString" 1 60\x0A`
Snapshot Load 24 Load Time 0 Bank testString	`ssl "testString" 24 0\x0A`
Snapshot Load 24 Load Time 60 Bank testString	`ssl "testString" 24 60\x0A`
Snapshot Save 1 Bank testString	`sss "testString" 1\x0A`
Snapshot Save 24 Bank testString	`sss "testString" 24\x0A`

Appendix B. Update Commands

Call Status Control ID testString	`get "testString"\x0A`
Design Name	`sg\x0A`
Focus Mode Control ID testString	`get "testString"\x0A`
Focus Speed Control ID testString	`get "testString"\x0A`
Function Control ID testString	`get "testString"\x0A`
Gain Control ID testString	`get "testString"\x0A`
Get Status Value Control ID testString	`get "testString"\x0A`
Level Meter Control ID testString	`get "testString"\x0A`
Mute Control ID testString	`get "testString"\x0A`
PTZ Speed Control ID testString	`get "testString"\x0A`
Router Control ID testString	`get "testString"\x0A`